Neural Comput & Applic (2018) 29:317–328
https://doi.org/10.1007/s00521-017-3005-9

EANN 2016

# Application of feature selection methods for automated clustering analysis: a review on synthetic datasets

**Aliyu Usman Ahmad**[1] · **Andrew Starkey**[1]

Received: 28 November 2016 / Accepted: 6 April 2017 / Published online: 22 April 2017


**Abstract** The effective modelling of high-dimensional data with hundreds to thousands of features remains a challenging task in the field of machine learning. This process is a manually intensive task and requires skilled data scientists to apply exploratory data analysis techniques and statistical methods in pre-processing datasets for meaningful analysis with machine learning methods. However, the massive growth of data has brought about the need for fully automated data analysis methods. One of the key challenges is the accurate selection of a set of relevant features, which can be buried in high-dimensional data along with irrelevant noisy features, by choosing a subset of the complete set of input features that predicts the output with higher accuracy comparable to the performance of the complete input set. Kohonen's self-organising neural network map has been utilised in various ways for this task, such as with the weighted self-organising map (WSOM) approach and this method is reviewed for its efficacy. The study demonstrates that the WSOM approach can result in different results on different runs on a given dataset due to the inappropriate use of the steepest descent optimisation method to minimise the weighted SOM's cost function. An alternative feature weighting approach based on analysis of the SOM after training is presented; the proposed approach allows the SOM to converge before analysing the input relevance, unlike the WSOM that aims to apply weighting to the inputs during the training which distorts the SOM's cost function, resulting in multiple local minimums meaning the SOM does not consistently converge to the same state. We demonstrate the superiority of the proposed method over the WSOM and a standard SOM in feature selection with improved clustering analysis.

**Keywords** Clustering · Self-organising neural network map · Feature selection · Automation

✉ Aliyu Usman Ahmad
r01aua14@abdn.ac.uk

Andrew Starkey
a.starkey@abdn.ac.uk

[1] School of Engineering, University of Aberdeen, Aberdeen, UK

## 1 Introduction

Clustering is one of the most widely used data analysis methods for numerous practical applications in emerging areas [1]. Clustering entails the process of organising objects into natural groups by finding the class of objects such that the objects in a class are similar to one another and dissimilar from the objects in another class [2]. A clustering algorithm usually considers all input parameters in an attempt to learn as much as possible about the given objects.

The self-organising neural network map (SOM) by Kohonen [3] has been widely used as one of the most successful clustering methods with strong data exploration and visualisation capabilities [4]. The SOM's mapping preserves a topological relation by maintaining neighbourhood relations such that patterns that are close in the input space are mapped to neurons that are close in the output space and vice-versa.

One of the biggest drawbacks of the SOM algorithm is its inability to automatically identify the features that are relevant for analysis and discard the irrelevant inputs that negatively distort the analysis result [5]. In an attempt to resolve this, researchers [6–8] have worked on the improvement of the algorithm by a feature weighting method during training with the application of the steepest descent optimisation method for the identification of

important inputs for clustering (WSOM, weighted self-organising map). The core of the weighted method lies in attempting to describe the contribution of each feature in the clustering algorithm in order to improve the clustering result.

This paper investigates the application of an existing weighted method approach (WSOM) and proposes an alternative approach for identifying the key features in a number of artificially produced datasets and the real world dataset used in the original WSOM study [6–8]. The study demonstrates how information on what the learning algorithm has learnt can be used to identify what is important for the learning, and therefore applied to improve the algorithm's ability to correctly classify and identify patterns in the data.

## 2 Neural network clustering methods

### 2.1 Self-organising neural network map

The self-organising neural network map (SOM) is an unsupervised artificial neural network learning method trained to produce a low-dimensional representation of high-dimensional input samples [4].

A typical SOM consists of the computational layer (map) and the input layers as shown in Fig. 1.

The input layer comprises of the source nodes representing the sample's features/attributes. There are as many weights for each node as there are number of features (dimensions) in the input layer, represented in the form of an input vector, i.e. $x = [x_i^1, x_i^2, \ldots, x_i^d]$ for an input sample where $d$ denotes sample dimensions and $i$ the sample number and $n$ denoting the total number of samples.

The computational layer (map) consists of neurons placed in nodes of a 2-dimensional grid (lattice); each neuron is identified by its index position, i.e. $j$, on the map and associated with a weight vector, i.e. $W_j = \{w_{ji} : j = 1, \ldots, n; i = 1, \ldots, d\}$ , the size of which is equal to the dimension of the input vector. The set of weights $W$ parameters are determined by iteratively minimising the cost function below;

$$R(C, W) = \sum_{i=1}^{N} \sum_{j=1}^{|W|} \kappa_{j,c(x_i)} \left\| x_i - w_j \right\|^2 \quad (1)$$

At every $n$th training step, the Gaussian neighbourhood function is calculated for the map; this is expressed as:

$$K_{j,\mathrm{c}(x_i)\ (n)} = \alpha(n) \cdot \mathrm{e}\left( -\frac{\delta^2{}_{j,\mathrm{c}\ (x_i)}}{2\sigma(n)^2} \right) \quad (2)$$

Where

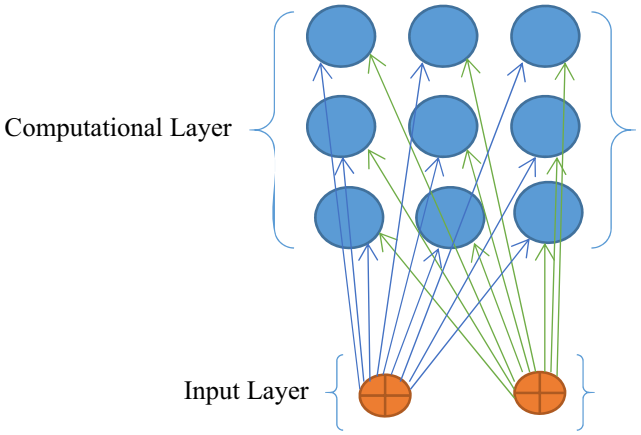


**Fig. 1** A 2-dimensional self-organising map architecture

- $K_{j,\mathrm{c}\ (x_i)\ (n)}$ is the neighbourhood function between each unit ($j$) on the map and the winning unit c ($x_i$) at the $n$th training step
- $\delta_{j,\mathrm{c}\ (x_i)}$ is the distance (Euclidean) from the position of unit ($j$) to the winning unit c ($x_i$) on the map.
- $\sigma(n)$ is the effective width of the topological neighbourhood at the $n$th training step; this serves as the moderator of the learning step during training iterations. The size of the effective width shrinks with time to facilitate the convergence of the map.
- $\alpha(n)$ is the learning rate that depends on the number of iterations ($n$); this is initialised to a value of around 0.1 which decreases from $\alpha_{\max}$ to $\alpha_{\min}$.

It is possible to use the results of a trained SOM in order to estimate the relevance of feature variables (weights). This is achieved by the use of the quantization error method [9] which is used to analyse the final result of the standard SOM for the identification of the features that were relevant during training.

## 3 Related work

Irrelevant input features are one of the major factors that distort the ability of learning algorithms for pattern recognition in data, as investigated by [10–13]. Numerous researchers have proposed feature selection methods for identifying and selecting the most important inputs in a data that best maximises the performance of learning algorithms [14–16] and can be categorised into filter, wrapper and embedded methods.

Filter methods aim to use statistical approaches to identify what inputs are important and are independent of the classifier, usually applied in the data pre-processing stage prior to training. Some of the commonly used filter methods include the RELIEF algorithm [17], correlation-based feature selection (CFS) [18], fast correlated-based filter (FCBF) [19] and the INTERACT [20] methods.

These are further discussed and compared by [21]. Other filter methods include mutual information [22] and Pearson's correlation coefficient scores [23]. Recently, hybrid methods are introduced; these combine the filter methods with a wrapper method for feature selection; some examples of such methods are presented by [24, 25].

Wrapper methods unlike filter methods aim to identify important inputs by searching for the best subset of features that produce the highest model classification accuracy. Commonly used wrapper methods include the recursive feature elimination method (RFE) [26] and exhaustive search and greedy forward search methods discussed in [27, 28]. Improvements to the above-discussed methods are reviewed and discussed by [29]. Recently emerged wrapper methods include polygon-based cross-validation (CV) with SVM method [30] and competitive swarm optimizer (CSO) [31].

The embedded methods learn about the importance of inputs from the model's training process and one of the examples of this method is the weighted self-organising map (WSOM) [8] that is investigated in this study. Other methods include hold out SVM (HO-SVM) and kernel-penalized SVM (KP-SVM) reviewed and discussed in [32].

## 4 Methodology

Most of the above methods require manual experimentation that consumes the bulk of the effort invested in the entire clustering analysis, with the exception of the embedded methods that identify important inputs from what the model has learnt. These latter methods are the area of focus in this study which aims to achieve the goal of a fully automated clustering process. We are particularly interested in the use of the self-organising map for automated feature selection due to its powerful topology preservation property, with the neighbourhood function (Eq. 2) that enables the SOM to not only group the data but also illustrates the underlying structure of the data for visualisation. As discussed in [33], the SOM has been widely applied especially for complex and high-dimensional datasets where traditional clustering methods are insufficient.

### 4.1 SOM weights analysis with quantization error method

On completion of SOM training which is achieved using the batch training method [3], the node weights values are expected to be the representation of their matching input samples, and also relatively close to the input samples mapped to their neighbouring nodes and relatively far from the input samples mapped to distant nodes.

Let $M_j$ be set of the training samples $x_i$ mapped to node $j$, and the quantization error for node $j$ is calculated after SOM training as follows:

$$E_j = \sum_{M_j} \left\| x_i - w_j \right\|^2 \tag{3}$$

The weight features with the lowest quantization error are expected to be the features whose corresponding input sample features are most relevant when comparing the samples against their winning nodes. A further analysis was carried out on the quantization error values for all the node weights in order to automatically separate the group of the relevant inputs from the irrelevant inputs, a parametric statistical test with median split was carried out on the quantization error values to differentiate the group of high values (as irrelevant features) from the group of low values (as relevant features). Since there is no reliance on a hard-coded threshold value to determine irrelevant and relevant features, this means that this step could be used for any amount of data features and results in a fully automated step for this aspect of the process.

### 4.2 Weighted self-organising neural network map

The weighted SOM (WSOM) function proposed by [8] is another method designed to compute the relevance of feature variables (weights) automatically during the training process. This approach entails the use of additional random weights that are multiplied by the input weights as a metric for measuring the relevance of the observations during training, and since the comparison is done one sample at a time, the updating method for the WSOM is incremental rather than batch as in the standard SOM.

Let $\Re^{\mathrm{d}}$ be the Euclidean data space and $E = \{ x_i; i = 1, \ldots, N\}$ a set of observations, where each observation $x_i = (x_i^1, x_i^2, \ldots, x_i^d)$ is a vector in $\Re^{\mathrm{d}}$.

Each node $j$ has prototype weights $w_j = (w_j^1, w_j^2, \ldots, w_j^d)$, and a single random weight is assigned to for each input attribute such that; $\pi_d = (\pi_1, \pi_2, \ldots \pi_d)$.

This method attempts to find the relevance of all the weights of a single vector which are applied against the whole set of input weights but is not able to determine the relevance of an individual weight of each node $j$ in a trained SOM.

The set of weights $W$ and $\pi$ parameters are determined by iteratively minimising the cost function as follows:

$$R_{gvw}(C, W, \pi) = \sum_{i=1}^{|E|} \sum_{j=1}^{|W|} \kappa_{j,c(x_i)} \left\| \pi_d \otimes x_i - w_j \right\|^2 \tag{4}$$

The cost function $R_{gvw}(W, \pi)$ is as described in Eq . 4. The algorithm is optimised by finding the min $W, \pi\ R_{gvw}(W, \pi)$.

The process begins by initially starting with some random values for $W, \pi$ then these values are modified in order to reduce $R_{gvw}$ ( $W, \pi$), until the minimum of the cost function is reached.

The method uses the steepest descent algorithm in order to optimise its cost function;

{

$$R_j := R_j - \alpha \frac{\partial}{\partial R_j} R_{gvw}(W, \pi) \quad (\text{For } j = W \text{ and } j = \pi) \quad (5)$$

}

The gradient descent minimization of the function can be implemented as;

$$w_j(n+1): = w_j(n) - (n) - \alpha(n)\kappa_{j,c(x_i)}(n)\kappa_{j,c(x_i)}\left(w_j - \pi_g \otimes x_i\right) \quad (6)$$

$$\pi_g(n+1): = \pi_g(n) - (n) - \alpha(n)\kappa_{j,c(x_i)}(n)\kappa_{j,c(x_i)}\ x_i\left(\pi_g \otimes x_i - w_j\right) \quad (7)$$

The steepest descent algorithm which is utilised by the WSOM method searches for the minimum of a function by computing the gradient of the function, starting at a random point $P_0$, and moving from $P_i$ to $P_{i+1}$ in the direction of the local downhill gradient $-\nabla f(P_i)$ for each iteration of line minimization.

The steepest descent method is guaranteed to find a solution for quadratic functions, which are convex-shaped functions with a single minimum that is equal to the global minimum [34] (as illustrated in Fig. 2). For problems beyond quadratic functions with multiple local minimums (such as Schwefel function, Fig. 3), the gradient descent finds the solution of a function based on the first identified local minimum and ignores other local minimums, and does not necessarily and cannot be guaranteed to find the global minimum of the given function. It is therefore important to confirm that the cost function for the WSOM method results in a single global minimum that can be found by the steepest descent approach.

For a full description of the WSOM process, the reader is directed to [7, 8]. In the WSOM approach, the relevance of an input vector is indicated by the global weights with irrelevant vectors having global weights close to 0 and relevant vectors having global weights different from 0. The relevance of an input vector can be measured by this method only if the given data features are normalised to the same scale.

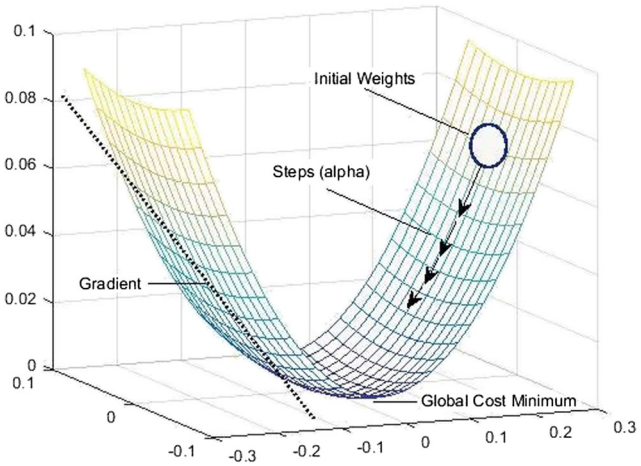


**Fig. 2** Steepest decent method for a quadratic functions

## 5 Experiment

### 5.1 Synthetic datasets

In order to assess the efficacy of the WSOM method against a standard SOM implementation, a number of synthetic datasets were developed which had different features, starting with a simple dataset with a small number of attributes and moving to datasets with a larger number of inputs and additional noise. All data sets had equal class distribution (i.e. same number of samples for each class), and normalised, with the exception of Synthetic_Data04 and Synthetic_Data05. These datasets are discussed more fully in the tables. The use of synthetic data rather than real data sets of this type is very important as it allows a full assessment of how well the techniques work and what type of problems they can solve, and where they may encounter difficulties if any.

In addition, the dataset used in the original WSOM paper was obtained which is a real world dataset of waveform data and is also described in Table 1.

### 5.2 Experiment design

As both methods rely on a random process, the performance of the algorithms was measured based on results of 10 runs for each of the methods on the synthetic datasets, and the results of these runs are shown in Tables 2, 3, 4, 5, 6 and 7.

However, in order to check whether the weighted SOM cost function (Eq. 4) for Synthetic_Data01 is a quadratic function, to be suitable for the steepest descent optimization approach, the cost function was optimised with the simulated annealing algorithm on Synthetic_Data01; a stochastic search method that aims to expose all possible minimums of the function by random search in space, with initial temperature $(T_o) = 10.0$ , cooling rate $(\alpha) = 0.99$ and maximum iteration (Maxtime) = 1000.

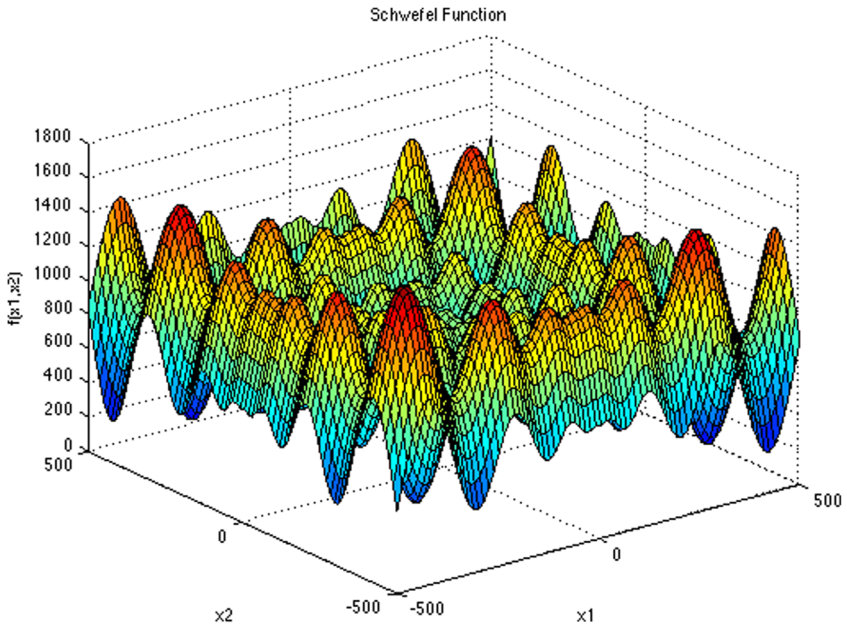


**Fig. 3** Problems beyond quadratic functions, Schwefel function [35]

If a cost function has a single global minimum, the best combination of weight values for different runs of the algorithm will be expected to be within the same region and to have a positive linear correlation when compared against each other. Otherwise, if the cost function has multiple local minimums, then the best combination of weight values would be in different regions for different runs of the algorithm and will not be correlated.

The normalised correlation matrix of the final weights from the simulated annealing algorithm was computed to show the similarity of the weights produced from six runs of the algorithm against each other; the same experiment was carried out on the standard SOM cost function for comparison.

When undertaking the correlation analysis for the WSOM method, the raw SOM weights cannot be used directly but must be divided by the corresponding global weights $\pi$. This step is required since the global weights and WSOM node weights are linked as described in the cost function, and it is possible due to the random nature of the process that different values could be arrived at for these variables whilst still mapping against similar input samples and the node weights could therefore be different from one run to the next.

In addition, the problem with direct comparison of weights at the same index for the six different simulated annealing runs is that nodes are not necessarily localised to a specific index. In a single run, a node might appear in the first index, whilst in a separate run, the same node might appear in a different index. As such, direct comparison of the weights infers the comparison of random unrelated nodes, which are most likely to be not correlated at all times.

To overcome this problem, the indexes of all the nodes weights was re-arranged to correspond to the best matching positions for all the nodes from the six different simulated annealing runs before carrying out the correlation test on the weights.

Let $E$ be set of weight values for a given SOM run $(w_n^i; n = 1, \ldots N)$, were $N$ is the total number of weights and $W$ is set of weight values for other SOM runs to total a number of SOM runs $R$. The index position $I$ of a node in a given SOM when compared against the SOM with weight values $E$ is computed as Eq. 8. For completeness, this was executed for all the SOM runs.

$$I = \sum_{n=1}^{\left|E\right|} \sum_{m=1}^{\left|W \ldots R\right|} min \ \left\| w_n^i - w_m^j \right\|^2 \tag{8}$$

A null hypothesis test $Ho : \rho = 0$ was conducted with 0.5 significance level to investigate the relationship between the final node weights (i.e. to see if they are correlated or not). Nodes are correlated if their correlation coefficient is different from zero, and therefore means there is linear relationship between the nodes. There is no correlation for nodes with correlation coefficients close to zero.

## 6 Results

The analysis of the correlation results can be seen in Figs. 4 and 5 where the bars in the plot represent the correlation coefficients values $\rho$ for a given run of the self-organising process compared against another on Synthetic_Data01. The red line on the plots at 0.5 and −0.5 indicate the respective

**Table 1** Synthetic and real datasets definition

| Dataset name | Samples | Input features | Classes |
|---|---|---|---|
| Synthetic_Data01 (Normalised) | 100 | 4 | 5 |
| | All classes defined by first 4 related features. | | |
| | This is a simple dataset with no irrelevant inputs and outliers, created mainly for exploring the cost functions of the two self-organising algorithms. | | |
| Synthetic_Data02 (Normalised) | 1220 | 7 | 5 |
| | All classes defined by first 4 related features. Irrelevant features: 5, 6, 7 | | |
| | Irrelevant inputs are clearly separated from the relevant inputs for easy identification by the algorithms. | | |
| Synthetic_Data03 (Normalised) | 1220 | 10 | 5 |
| | Classes defined by features independently with equal distribution. Class1 = 1, 2, & 3, Class2 = 4, 5, & 6, Class3 = 2, 3, 4 & 5, Class4 = 6, 7, & 8, Class5 = 1, 4, & 8. Noise features; features 9 & 10 | | |
| | In addition to Synthetic_Data02, the definition of classes was distributed among variables, to identify the self-organising method's ability to identify the degree of relevance of the input features for classification. | | |
| Synthetic_Data04 (Normalised) | 1220 | 9 | 5 |
| | Classes defined by features independently with unequal distribution. Class1 (550 samples) = 1,2, & 3, Class2 (300 samples) = 1, 2, & 3, Class3 (200 samples) = 2, 4, & 5, Class4 (100 samples) = 1, 3, 5, & 6, Class5 (70 samples) = 1,3,4, & 7, Noise features; features 8 & 9 | | |
| Synthetic_Data05 (Unnormalised) | 1220 | 7 | 5 |
| | All classes defined by first 4 related features. Irrelevant features: 5, 6, 7 | | |
| | This dataset was created to evaluate the self-organising system's performance in identifying irrelevant inputs from unnormalised datasets having features of unequal variance. | | |
| WaveForm dataset (Normalised) | 5000 | 40 | 3 |
| | As described by [36] the first 21 inputs of the waveform data describe the classes, the latter 19 are completely irrelevant noise features with mean 0 and variance 1. More details can be found from the UCI repository online. No information is provided on which inputs out of the first 21 describe each class. | | |

boundaries for positive and negative correlations, with no correlation shown at or around 0.

The bars above the red line indicate the pairs of node weights with correlations significantly different from 0, which implies that there is a significant linear relationship between the weights of the various runs (i.e. the SOM weights are broadly equivalent and the values can be said to be the same). On the other hand, the bars below the line indicate the pairs of node weights with a correlation coefficient that is not significantly different from 0, which implies that there is no significant linear relationship between the weights (i.e. the SOM weights are not the same and contain different values).

The correlation matrix for the standard SOM (Fig. 4) weights shows that almost all pairs of weights (4 out of 6) have correlations significantly different from zero which proves positive correlation among weights. On the other hand, the correlation matrix for the weighted SOM (Fig. 5) shows that only two out of the six weights are correlated, which indicates that this method has resulted in different solutions being found.

In summary, it can be concluded that the simulated annealing algorithm with the standard SOM cost function finds similar solutions in the majority of the different runs. On the other hand, the algorithm with the weighted SOM cost function finds different solutions for most of the runs, which is most likely to be as a result of multiple local minimums in the cost function.

The results given in Tables 2, 3, 4, 5 and 6 for the five synthetic datasets and the real world waveform dataset

**Table 2** Performance of clustering methods on Synthetic_Data01

| Clustering Synthetic_Data01 | | | | | | |
|---|---|---|---|---|---|---|
| Training parameters | Map dimension, 3 × 3 rectangular grid topology<br>Training epochs, 1000<br>Learning rate, 0.1 | | | | | |
| | Weighted SOM | | | Standard SOM | | |
| RUNS | Identified important inputs | Correct classes found (all inputs) | Correct classes found (selected inputs) | Identified important attributes | Correct classes found (all inputs) | Correct classes found (selected inputs) |
| Run 1 | 1/4 | 1/5 | 0/5 | 4/4 | 5/5 | 5/5 |
| Run 2 | 1/4 | 1/5 | 0/5 | 4/4 | 5/5 | 4/5 |
| Run 3 | 1/4 | 1/5 | 1/5 | 4/4 | 5/5 | 5/5 |
| Run 4 | 1/4 | 2/5 | 1/5 | 4/4 | 5/5 | 5/5 |
| Run 5 | 2/4 | 1/5 | 2/5 | 4/4 | 4/5 | 5/5 |
| Run 6 | 1/4 | 0/5 | 0/5 | 4/4 | 5/5 | 5/5 |
| Run 7 | 1/4 | 1/5 | 0/5 | 4/4 | 5/5 | 5/5 |
| Run 8 | 1/4 | 0/5 | 1/5 | 4/4 | 5/5 | 5/5 |
| Run 9 | 1/4 | 2/5 | 1/5 | 4/4 | 5/5 | 5/5 |
| Run 10 | 1/4 | 0/5 | 0/5 | 4/4 | 4/5 | 5/5 |

show the performance of the two methods in identifying the important attributes and whether classes were correctly classified. The tables also give details of the training parameters used. On completion of the classification after training with all inputs, the inputs identified by the respective feature selection method was applied to the

**Table 3** Performance of clustering methods on Synthetic_Data02

| Clustering Synthetic_Data02 | | | | | | |
|---|---|---|---|---|---|---|
| Training parameters | Map dimension, 3 × 3 rectangular grid topology<br>Training epochs, 1000<br>Learning rate, 0.1 | | | | | |
| | Weighted SOM | | | Standard SOM | | |
| RUNS | Identified important inputs | Correct classes found (all inputs) | Correct classes found (selected inputs) | Identified important attributes | Correct classes found (all inputs) | Correct classes found (selected inputs) |
| Run 1 | 1/4 | 0/5 | 0/5 | 4/4 | 2/5 | 5/5 |
| Run 2 | 0/4 | 0/5 | – | 4/4 | 1/5 | 5/5 |
| Run 3 | 2/4 | 1/5 | 2/5 | 4/4 | 2/5 | 5/5 |
| Run 4 | 1/4 | 0/5 | 1/5 | 3/4 | 1/5 | 4/5 |
| Run 5 | 2/4 | 0/5 | 1/5 | 4/4 | 1/5 | 5/5 |
| Run 6 | 3/4 | 1/5 | 1/5 | 4/4 | 2/5 | 5/5 |
| Run 7 | 1/4 | 1/5 | 1/5 | 4/4 | 3/5 | 5/5 |
| Run 8 | 2/4 | 0/5 | 2/5 | 4/4 | 2/5 | 5/5 |
| Run 9 | 0/4 | 0/5 | – | 4/4 | 3/5 | 5/5 |
| Run 10 | 1/4 | 0/5 | 1/5 | 4/4 | 2/5 | 5/5 |

**Table 4** Performance of clustering methods on Synthetic_Data03

| Clustering Synthetic_Data03 | | | | | | |
|---|---|---|---|---|---|---|
| Training parameters | Map dimension, 3 × 3 rectangular grid topology<br>Training epochs, 1000<br>Learning rate, 0.1 | | | | | |
| | Weighted SOM | | | Standard SOM | | |
| RUNS | Identified important inputs | Correct classes found (all inputs) | Correct classes found (selected inputs) | Identified important attributes | Correct classes found (all inputs) | Correct classes found (selected inputs) |
| Run 1 | 0/8 | 0/5 | – | 2/8 | 3/5 | 2/5 |
| Run 2 | 0/8 | 0/5 | – | 4/8 | 1/5 | 3/5 |
| Run 3 | 1/8 | 1/5 | 0/5 | 2/8 | 1/5 | 2/5 |
| Run 4 | 0/8 | 0/5 | – | 3/8 | 0/5 | 2/5 |
| Run 5 | 1/8 | 0/5 | 0/5 | 1/8 | 1/5 | 1/5 |
| Run 6 | 2/8 | 0/5 | 1/5 | 1/8 | 0/5 | 1/5 |
| Run 7 | 1/8 | 1/5 | 1/5 | 5/8 | 1/5 | 4/5 |
| Run 8 | 1/8 | 0/5 | 0/5 | 1/8 | 2/5 | 2/5 |
| Run 9 | 0/8 | 0/5 | – | 2/8 | 1/5 | 2/5 |
| Run 10 | 1/8 | 0/5 | 0/5 | 2/8 | 1/5 | 2/5 |

dataset so that input samples were remapped to the trained weights with only the selected inputs. If the features have been identified correctly, then it is assumed that the classification of the samples would remain the same. The

**Table 5** Performance of clustering methods on Synthetic_Data04

| Clustering Synthetic_Data04 | | | | | | |
|---|---|---|---|---|---|---|
| Training parameters | Map dimension, 3 × 3 rectangular grid topology<br>Training epochs, 1000<br>Learning rate, 0.1 | | | | | |
| | Weighted SOM | | | Standard SOM | | |
| RUNS | Identified important inputs | Correct classes found (all inputs) | Correct classes found (selected inputs) | Identified important attributes | Correct classes found (all inputs) | Correct classes found (selected inputs) |
| Run 1 | 2/7 | 1/5 | 1/5 | 4/7 | 3/5 | 2/5 |
| Run 2 | 1/7 | 1/5 | 0/5 | 4/7 | 2/5 | 3/5 |
| Run 3 | 1/7 | 1/5 | 0/5 | 2/7 | 1/5 | 1/5 |
| Run 4 | 1/7 | 1/5 | 0/5 | 4/7 | 4/5 | 2/5 |
| Run 5 | 1/7 | 1/5 | 0/5 | 3/7 | 2/5 | 2/5 |
| Run 6 | 0/7 | 0/5 | – | 4/7 | 2/5 | 3/5 |
| Run 7 | 2/7 | 1/5 | 1/5 | 4/7 | 2/5 | 3/5 |
| Run 8 | 1/7 | 1/5 | 0/5 | 2/7 | 1/5 | 1/5 |
| Run 9 | 1/7 | 1/5 | 0/5 | 2/7 | 1/5 | 1/5 |
| Run 10 | 1/7 | 1/5 | 0/5 | 4/7 | 2/5 | 3/5 |

**Table 6** Performance of clustering methods on Synthetic_Data05

| Clustering Synthetic_Data05 | | | | | | |
|---|---|---|---|---|---|---|
| Training parameters | Map dimension, 3 × 3 rectangular grid topology<br>Training epochs, 1000<br>Learning rate, 0.1 | | | | | |
| | Weighted SOM | | | Standard SOM | | |
| RUNS | Identified important inputs | Correct classes found (all inputs) | Correct classes found (selected inputs) | Identified important attributes | Correct classes found (all inputs) | Correct classes found (selected inputs) |
| Run 1 | 2/4 | 0/5 | 1/5 | 3/4 | 3/5 | 5/5 |
| Run 2 | 0/4 | 0/5 | – | 4/4 | 2/5 | 5/5 |
| Run 3 | 1/4 | 1/5 | 2/5 | 4/4 | 2/5 | 5/5 |
| Run 4 | 1/4 | 0/5 | 1/5 | 4/4 | 2/5 | 5/5 |
| Run 5 | 1/4 | 0/5 | 2/5 | 4/4 | 2/5 | 5/5 |
| Run 6 | 0/4 | 1/5 | – | 4/4 | 2/5 | 5/5 |
| Run 7 | 2/4 | 1/5 | 0/5 | 4/4 | 2/5 | 5/5 |
| Run 8 | 1/4 | 1/5 | 1/5 | 4/4 | 2/5 | 5/5 |
| Run 9 | 0/4 | 1/5 | – | 4/4 | 3/5 | 5/5 |
| Run 10 | 1/4 | 1/5 | 0/5 | 3/4 | 2/5 | 5/5 |

remapping of input samples was achieved by only recalculating the best matching units (BMUs) [3] for the selected features against their corresponding node weights. Classes are identified if at least 60% of class samples from the input samples belonging to the same class are mapped to the same node.

**Table 7** Performance of clustering methods for waveform data

| Clustering waveform data | | | | | |
|---|---|---|---|---|---|
| Training parameters | Map dimension, 26 × 14 rectangular grid topology<br>Training epochs, 1000<br>Learning rate, 0.1 | | | | |
| | Weighted SOM | | Standard SOM | | |
| RUNS | Identified important inputs | Correct classes found (selected inputs) | Identified important attributes | Correct classes found (all inputs) | Correct classes found (selected inputs) |
| Run 1 | 2/20 | 1/3 | 18/20 | 1/3 | 2/3 |
| Run 2 | 9/20 | 1/3 | 18/20 | 1/3 | 3/3 |
| Run 3 | 19/20 | 2/3 | 15/20 | 2/3 | 2/3 |
| Run 4 | 11/20 | 1/3 | 18/20 | 1/3 | 3/3 |
| Run 5 | 5/20 | 1/3 | 18/20 | 1/3 | 3/3 |
| Run 6 | 19/20 | 2/3 | 18/20 | 2/3 | 2/3 |
| Run 7 | 15/20 | 2/3 | 19/20 | 1/3 | 3/3 |
| Run 8 | 2/20 | 1/3 | 18/20 | 1/3 | 2/3 |
| Run 9 | 16/20 | 2/3 | 17/20 | 1/3 | 2/3 |
| Run 10 | 11/20 | 1/3 | 18/20 | 1/3 | 3/3 |

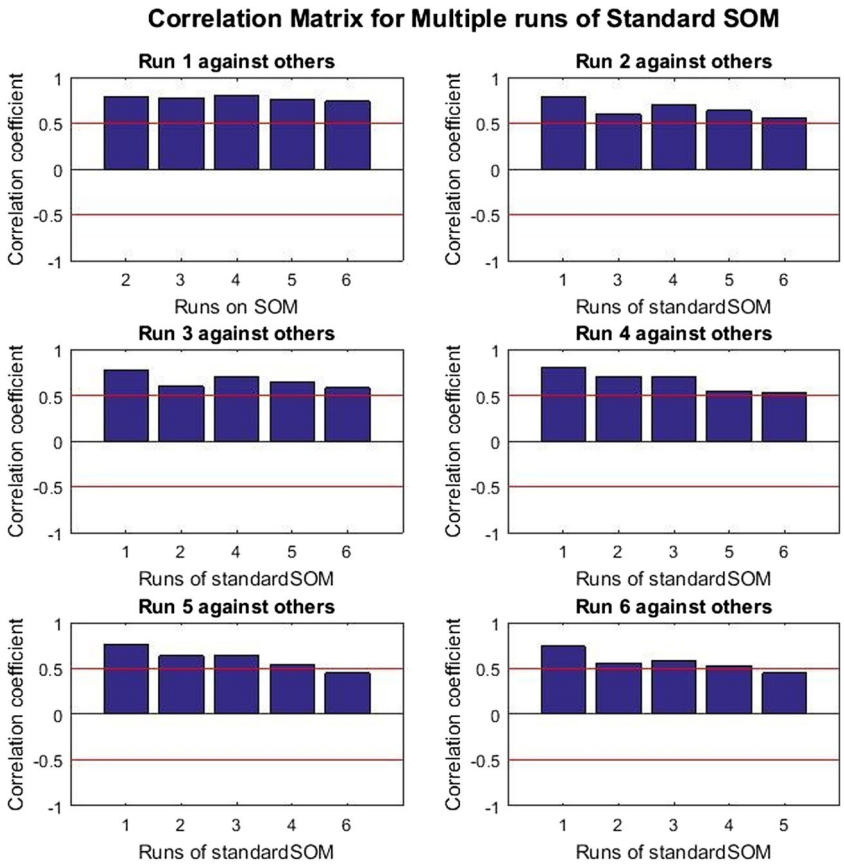


**Fig. 4** Correlation matrix for multiple runs of standard SOM on Synthetic_Data01

## 7 Discussion and conclusions

As seen in Table 2, the standard SOM was able to correctly identify all the classes in most of the runs for simple data with no irrelevant inputs and was also able to identify all inputs as important due to low quantization error between weights to their mapped input samples. Unlike the standard SOM, the weighted SOM failed to identify the classes for the same simple data set with no irrelevant inputs. The weighted SOM method also performed poorly by failing to correctly identify clusters and differentiate the relevant input vectors from the irrelevant input vectors on the Synthetic_Data02. However, the standard SOM with quantization error method after training clearly identified the relevant vectors for the training on this dataset. Both methods performed poorly in correctly identifying the clusters in the data as the result of the influence of the irrelevant inputs during the training.

For the more complicated dataset with overlapping class definition (Synthetic_Data03 and Synthetic_Data04), the analysis of the standard SOM's training result with the quantization error also failed to identify what was important for the training, as presented in Tables 4 and 5.

As discussed in Section 4.2, the steepest descent algorithm is guaranteed to find the local minimum for quadratic functions with a single global minimum, whereas for functions with multiple local minimums, the gradient descent finds the solution of the function based on the first identified local minimum ignoring other local minimums, and therefore is not suitable for the proposed WSOM cost function as our results clearly demonstrate that multiple minimums exist in the solution space defined by the WSOM cost function.

As seen from the performance of methods on Synthetic_Data03 and Synthetic_Data04, identifying feature relevance based on classification results of the SOM does not provide good results when the SOM has not classified the samples correctly. The results also show that complete removal of the identified irrelevant input features produces worse class identification as shown in runs 1 and 4 in Table 5. This demonstrates the importance of identifying feature relevance based on class or at node level rather than a global SOM analysis and suggests the use of techniques that reduce the importance of identified irrelevant inputs rather than completely removing them.

A further experiment on the WSOM method with the WaveForm data that was used in the original WSOM paper has revealed that the method is able to group the dataset occasionally (i.e. at some random iteration) during multiple runs, but that other runs show a different set of weightings with a different solution. This provides further evidence to support the conclusion reached of multiple local minimums in the

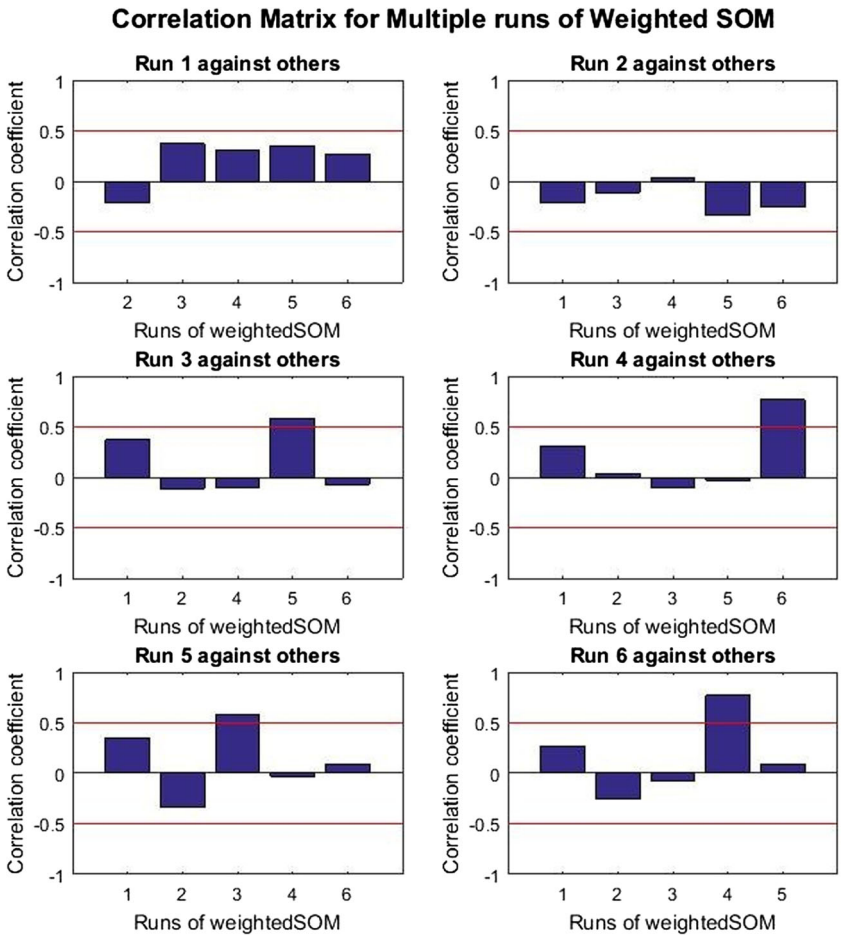


**Fig. 5** Correlation matrix for multiple runs of weighted SOM on Synthetic_Data01

method's cost function. The same SOM size of 26 × 14 as used by [8] was used for comparison. These results indicate that the WSOM method should be used with caution and that multiple runs may be required depending on the underlying data set in order to ensure that the optimal results are found. In practice, it may be difficult to know when these have been obtained making the use of this method problematic.

The quantization error between the weight values and their matched classified input samples shows more potential for identifying important features; however, these results show that this approach will only work for certain types of data. One of the limitations of the proposed method is its inability to correctly identify irrelevant inputs for a SOM with inappropriate topology size and having highly misdiagnosed classes (i.e. multiple classes mapped to a single node). This implies the requirement for an incremental system with the ability to automatically adjust the SOM's topology size during training, to allow the spread of class samples across nodes for more accurate input relevance analysis. It is also interesting to note for some of the synthetic datasets that the quantization method correctly identifies the important features despite not being able to correctly classify the groupings in the data. This suggests the need for an additional layer that uses the relevance information from the feature selection method to prune or suppress the irrelevant features and guide the remapping of a self-organising system with the relevant features for a higher clustering performance to achieve a fully automated clustering process, and this will be the subject of future work.

#### Compliance with ethical standards

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



## References

1. Tajunisha S, Saravanan V (2010) Performance analysis of k-means with different initialization methods for high dimensional data 1:44–52
2. Aggarwal CC, Reddy CK (2013) Data clustering: algorithms and applications. CRC Press

3. Kohonen T (1990) The self-organizing map. Proc IEEE 78:1464–1480
4. Yin H (2008) The self-organizing maps: background, theories, extensions and applications. In: Anonymous Computational intelligence: A compendium, Springer, pp 715–762
5. Shafreen Banu A, Ganesh SH (2015) A study of feature selection approaches for classification: 1–4
6. De Carvalho, FAT, Bertrand P, Simões EC (2015) Batch SOM algorithms for interval-valued data with automatic weighting of the variables. Neurocomputing
7. Mesghouni N, Temanni M (2011) Unsupervised double local weighting for feature selection 1:413–417
8. Grozavu N, Bennani Y, Lebbah M (2009) From variable weighting to cluster characterization in topographic unsupervised learning 1005–1010
9. De Bodt E, Cottrell M, Verleysen M (2002) Statistical tools to assess the reliability of self-organizing maps. Neural Netw 15: 967–978
10. Liu H, Yu L (2005) Toward integrating feature selection algorithms for classification and clustering. IEEE Trans Knowled Data Eng 17: 491–502
11. Dash M, Liu H (1997) Feature selection for classification 1:131–156
12. Hua J, Tembe WD, Dougherty ER (2009) Performance of feature-selection methods in the classification of high-dimension data. Pattern Recogn 42:409–424
13. Kaushik A, Gupta H, Latwal DS (2016) Impact of feature selection and engineering in the classification of handwritten text 2598–2601
14. Vega R, Sajed T, Mathewson KW, Khare K, Pilarski PM, Greiner R, Sanchez-Ante G, Antelis JM (2016) Assessment of feature selection and classification methods for recognizing motor imagery tasks from electroencephalographic signals 6:p37
15. Jiménez F, Jódar R, Martín MDP, Sánchez G, Sciavicco G (2016) Unsupervised feature selection for interpretable classification in behavioral assessment of children. Exp Syst
16. Kwak N, Choi C (2002) Input feature selection for classification problems. IEEE Trans Neural Netw 13:143–159
17. Kira K, Rendell LA (1992) A practical approach to feature selection 249–256
18. Hall MA (1999) Correlation-based feature selection for machine learning. Dissertation, The University of Waikato
19. Yu L, Liu H (2003) Feature selection for high-dimensional data: a fast correlation-based filter solution 3:856–863
20. Zhao Z, Liu H (2007) Searching for interacting features. 7:1156–1161
21. Sánchez-Maroño N, Alonso-Betanzos A, Tombilla-Sanromán M (2007) Filter methods for feature selection–a comparative study 178–187
22. Amiri F, Yousefi MR, Lucas C, Shakery A, Yazdani N (2011) Mutual information-based feature selection for intrusion detection systems 34:1184–1199
23. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection 3:1157–1182
24. Chuang LY, Ke CH, Yang CH (2016) A hybrid both filter and wrapper feature selection method for microarray classification. arXiv preprint arXiv:1612.08669. Dec 27
25. Apolloni J, Leguizamón G, Alba E (2016) Two hybrid wrapper-filter feature selection algorithms applied to high-dimensional microarray experiments. Appl Soft Comput 38:922–932
26. Granitto PM, Furlanello C, Biasioli F, Gasperi F (2006) Recursive feature elimination with random forest for PTR-MS analysis of agroindustrial products. Chemometrics Intellig Lab Syst 83:83–90
27. Chandrashekar G, Sahin F (2014) A survey on feature selection methods. Comput Electr Eng 40:16–28
28. Tang J, Alelyani S, Liu H (2014) Feature selection for classification: a review 37
29. Li J, Cheng K, Wang S, Morstatter F, Trevino RP, Tang J, Liu H (2016) Feature selection: a data perspective
30. Ma L, Li M, Gao Y, Chen T, Ma X, Qu L (2017) A novel wrapper approach for feature selection in object-based image classification using polygon-based cross-validation. IEEE Geosci Remote Sens Lett 14(3):409–413
31. Gu S, Cheng R, Jin Y (2016) Feature selection for high-dimensional classification using a competitive swarm optimizer. Soft Computing pp 1–12
32. Maldonado S, Weber R (2011) Embedded feature selection for support vector machines: state-of-the-art and future challenges 304–311
33. Tasdemir K, Merényi E (2009) Exploiting data topology in visualization and clustering of self-organizing maps. IEEE Trans Neural Netw 20:549–562
34. Gonzaga CC, Schneider RM (2015) On the steepest descent algorithm for quadratic functions 1–20
35. Schwefel H (1977) Numerische optimierung von computer-modellen mittels der evolutionsstrategie. Birkhäuser, Basel Switzerland
36. Breiman L, Friedman JH, Olshen RA, Stone CJ (1984) Classification and regression trees, the Wadsworth statistics and probability series, Wadsworth international group, Belmont California (pp 356)