Endless Science Open Journal of Machine Learning

Clustering: Finding Patterns in the Darkness

Héctor D. Menéndez*

Middlesex University London and Endless Science

h.menendez@mdx.ac.uk

Abstract

Machine learning is changing the world and fuelling Industry 4.0. These statistical methods focused on identifying patterns in data to provide an intelligent response to specific requests. Although understanding data tends to require expert knowledge to supervise the decision-making process, some techniques need no supervision. These unsupervised techniques can work blindly but they are based on data similarity. One of the most popular areas in this field is clustering. Clustering groups data to guarantee that the clusters' elements have a strong similarity while the clusters are distinct among them. This field started with the K-means algorithm, one of the most popular algorithms in machine learning with extensive applications. Currently, there are multiple strategies to deal with the clustering problem. This review introduces some of the classical algorithms, focusing significantly on algorithms based on evolutionary computation, and explains some current applications of clustering to large datasets.

Keywords: Clustering; K-means; Expectation-Maximization; Spectral Clustering; Evolutionary Computation

1 Introduction

Machine learning is one of the main areas of artificial intelligence. This area reflects how statistics understands the behaviour of a system based on previous data. This ability allows the creation of forecasting models that during the last years have predicted several aspects of human lives.

Normally, these techniques are divided into those based on supervised knowledge and those based on unsupervised knowledge. There are also several sub-fields in this area (Section 2), but the one bringing this work to life is clustering, unsupervised

*Corresponding Author

algorithms that extract patterns from data with no human intervention. Unsupervised learning has the capabilities to find patterns in data based on the way it is distributed in a feature space. Clustering focuses on grouping the data.

Given a dataset, a clustering algorithm groups the data based on their similarities blindly [55]. This simple idea has been extended to multiple different data representations, providing clustering with the ability to adapt to multiple different contexts, for instance, we can find clustering algorithms applied to automatic summarization [69], to identify influencers in social networks [78], and to recognise elements inside of photos [61], among others.

The clustering process starts with a dataset representing objects, a notion of similarity among the objects, and an algorithm that will perform the grouping [55]. The main goal of the algorithm is to optimise a cost function that will aim to reduce the similarities between different clusters and to increase the similarity among the elements of a cluster. During my PhD, I worked on how to improve this cost function reduction process. During that period, I discovered bio-inspired algorithms, which became one of the main research fields of my career. One of the main areas that can be applied for optimization within bio-inspired computation is evolutionary algorithms [22].

There are also some problems that clustering algorithms need to deal with, especially when the data follows specific patterns in their feature space. These problems are normally categorised as manifold detection problems [65], and one of the classical algorithms dealing with them is spectral clustering [77]. However, this algorithm lacks some features such as stability and scalability [63]. Here is where I started to use the strength of evolutionary computation to improve the accuracy and stability of spectral clustering algorithms. Spectral clustering leverages a similarity graph to represent the data space. The aim of the algorithm is to find the best cut for separating the data into clusters. This is where the problems start. The graph representation is very difficult to scale, the cut decision is sensitive to the similarity metric parameters, and the optimization process is slow [66]. My work was based initially on improving the stability and accuracy of the algorithm using evolutionary computation [66], after I started working on the scalability and online systems [65], and I also tested it with different bio-inspired algorithms [71].

This humble review aims to introduce some of the classical clustering algorithms and gives an overview of the application of evolutionary computation to this problem, giving also an overview of the data mining pipeline. It starts by providing some background on the topics of data mining and data representation (Section 2). Then, it deals with the clustering problem (Section 3) and different evolutionary algorithms applied to clustering (Section 4). After, it focuses on clustering algorithms for big data problems (Section 5). Then, I conclude with some examples of clustering applications (Section 6).

2 Data Mining and Machine Learning

Data Mining is "the process of discovering meaningful new correlations, patterns and trends by sifting through large amounts of data stored in repositories, using pattern recognition technologies as well as statistical and mathematical techniques" [51]. Data mining analysts apply these techniques following five steps:

1. **Data Extraction:** This step obtains the datasets that will be analysed. There are several public databases, for example, the UCI Machine Learning Repository

[33], widely used to test Data Mining algorithms.

2. **Data Preprocessing and Normalization:** This step prepares the data for analysis. It aims to [51]: avoid misclassification, reduce the data dimensions (through projections or feature selection techniques), and normalize.
3. **Model Generation:** This is the most important part of the data analysis process. An algorithm will create a model based on the data patterns. It is usual to use machine learning or other statistical techniques to generate the model [51].
4. **Model Validation:** Depending on the type of model, the validation process is different. This process evaluates the model parameters. It is usual to use validation with classifiers [51], however, for the clustering problem, validation is almost a blind process [97] (which is a consequence of the clustering nature).
5. **Model Application:** The model goal is to be applied in order to predict the behaviour of new inputs.

This review mainly focuses on the model generation process. These methods are based on statistical inference and machine learning techniques. Machine learning algorithms receive a sequence of inputs, called data, and look for patterns that predict or explain the behaviour of these data to extrapolate to new inputs. Some applications of these models can be found in [97]: business, biology, music, human behaviour, games, amongst others. These techniques can be divided into four main categories [1]:

- *Supervised Learning:* A sequence of desired outputs is also given with the inputs. The algorithm's goal is to learn how to produce the correct output given new inputs. The output could be a class label (classification) or a real number (regression). Some examples of classical supervised methods are [51]: decision trees, support vector machines, and neural networks.
- *Reinforcement Learning:* The algorithm produces a set of actions that affect the effectiveness of an environment. These effects generate a reward (or punishment) that the algorithm aims to maximize through its decisions.
- *Game Theory Learning:* It is a generalization of reinforcement learning. In this case, the environment can contain other entities with the same characteristics.
- *Unsupervised Learning:* The algorithm simply receives the input data. Its goal is to generate the labels based on the inputs' similarities. This review focuses on these algorithms.

The most common unsupervised data mining method is clustering. Another good example of an unsupervised learning method is dimensionality reduction [1] which is the process of reducing the number of random variables under consideration in a dataset analysis. These techniques are also known as feature selection methods.

2.1 Data Representation and Databases

Every data mining process starts by acquiring the data. Data have several representations and can be obtained from different sources. These techniques strongly depend on the data information and structure. On the one hand, the kind of information provided determines the algorithm needed for the analysis (e.g., clustering, classification, regression, among others). On the other hand, the structure defines the way the data is analysed (e.g., time series analysis, text mining, image segmentation, etc). Using this information, the analyser determines the best technique to extract patterns from the

data. Data structures can be divided into several categories. The most relevant are the following [51]:

- **Static Data** [33]: This kind of data is the most frequent. It is usually divided into numerical and categorical data and can be represented as a matrix. When it is categorical, it is usually converted to a numerical representation.
- **Time Series** [95]: Time series data are similar to static data but it pays attention to the temporal component. These data are usually used to predict the future behaviour of a set of events in one -or several, variables (e.g., stock market).
- **Texts** [17]: Text documents usually provide information that is analysed in a semantic context. Different techniques, such as Topic Detection and Tracking (TDT) or document summarization, are good examples of how these data need to be interpreted in a different way.
- **Images** [100]: Images have two main kinds of analysis: detect different parts of an image (e.g. segmentation), or group images by similarity (e.g., discriminate images that have examples of cars, trains, etc. against other images). An image representation is usually a set of pixels.
- **Stream data** [4]: Stream data is a concept which describes those data sources that provide data continually. The algorithms used to analyse these kinds of data are usually focused on large data analysis or online analysis.

In addition to the data structure identification process, it is important to consider different databases where these data types can be found. Some examples are:

- **UCI Machine Learning Repository** [33]: the Center of Machine Learning and Intelligent Systems provides around 600 datasets used for data analysis. This repository is extremely useful to test new algorithms.
- **UCR Time Series Classification Archive** [26]: this repository is specific for time series classification. It provides around 125 datasets. This repository works as a strong benchmark on time series analysis using machine learning.
- **Kaggle** [1]: Kaggle is one of the most relevant machine learning competitions that provides multiple datasets to learn about techniques and tools under different data representations. It also provides a large database of datasets to test algorithms.
- **R** [83]: The R-project also provides different benchmarks and packages which can generate synthetic data to test algorithms.
- **Berkeley Segmentation Dataset** [57]: This dataset is a benchmark for image segmentation analysis. It provides different images and a few human-based segmentations to evaluate new techniques.
- **BioMed** [2]: This database contains several references to medical and biomedical literature. These documents are usually papers or books about medical research.

3 Classical Clustering Techniques

Clustering has become an important field in data mining. It is used to find hidden information or patterns in an unlabelled dataset and has several applications related

[1] https://www.kaggle.com/datasets

to biomedicine [98], marketing [41], image segmentation [80] and virtual worlds [15] amongst others.

Clustering techniques are frequently used in machine learning. A popular clustering technique, when the number of clusters is known, is K-means [55]. Other methods, such as Expectation-Maximization (EM) [30], are applied when the number of clusters is unknown. EM is an iterative optimization method that estimates some unknown parameters computing probabilities of cluster membership based on one or more probability distributions; its goal is to maximize the overall probability or likelihood of the data being in the final clusters [76].

Since these techniques fix the number of clusters a priori, there are validation techniques such as cross-validation [47] which are used to improve the number of clusters selection (through metrics such as the Minimum Sum-of-Squares [73]).

Other research lines have tried to improve these algorithms. For example, some *online* methods have been developed to avoid the K-means convergence problem to local solutions which depend on the initial values [12]. These methods create the clusters adding a new data instance at each step and modifying the cluster structure with this new information. Some other improvements of K-means are related to dealing with the different kinds of data representation, for example, mixed numerical data [7] or categorical data [85]. There are also some studies comparing methods with different datasets, for example, Wang et al. [93] compare self-organizing maps, hierarchical clustering and competitive learning where establishing molecular data models of large size sets.

Machine learning techniques have also been improved through the k-means algorithm, for example, reinforcement learning algorithms [11, 37]; or using topological features of the data set [36, 37], which can also be helpful for data visualization.

The following sections introduce three classical clustering algorithms: K-means, Expectation-Maximization, and Spectral Clustering.

3.1 K-means

K-means [54] is maybe the most popular and well-known partitional clustering algorithm. It is a straightforward clustering guided method (usually by a heuristic) to group data in a predefined number of clusters. Given a fixed number of clusters (k), K-means tries to find a division of the dataset based on a set of common features given by distances (or metrics) that are used to determine what elements belong to each cluster [55]. K-means has also been improved through different techniques, like genetic algorithms [13]. Algorithm 1 shows the pseudo-code for the K-means algorithm [51].

K-means initially sets the centroids (line 1) and the elements are added to the cluster whose centroid is closer to them (lines 5 to 7). After, the position of the new centroids of the clusters are calculated (line 9) and again the closest elements are added (lines 5 to 7). It continues until the centroid position converges to a fixed point (line 2).

3.2 Expectation Maximization

Expectation-Maximization (EM) [30] is used when the number of clusters is unknown. Initially, it takes the likelihood and tries to maximize it. The process consists of applying the two following steps iteratively until it converges:

- **Expectation step**: Fix a model (θ) and estimate missing labels ($\mathbf{y}$).

Algorithm 1 Pseudo-code for K-means algorithm

Input: A dataset of n elements $X = \{x_1, \ldots, x_n\}$ and a fix number of clusters k.
Output: A set of clusters $C = \{C_1, \ldots, C_k\}$ which partitionate X

1: Assign k records to be the initial cluster centroids. We define the set of centroids as $Y = \{y_1, \ldots, y_k\}$ and $Y' = \emptyset$
2: **while** $Y \neq Y'$ **do**
3: Set all $C_j = \emptyset$.
4: $Y' \leftarrow Y$.
5: **for all** $x_i \in X$ **do**
6: Calculate the minimal distance centroid to x_i. Let y_j be the minimal distance centroid to x_i.
7: Introduce x_i in C_j.
8: **end for**
9: Calculate the centroids of C and set $y_i \leftarrow centroid(C_i)$.
10: **end while**
11: **return** C

- **Maximization step**: Fix missing labels ($\mathbf{y}$) (or a distribution over missing labels) and find the model (θ) that maximizes the likelihood function ($L(\theta)$).

The likelihood function is defined by:

$$L(\theta) = p(\mathbf{X}, \mathbf{y}|\theta) = \prod_i p(\mathbf{x}_i, y_i|\theta)$$

Where θ is a model defining how each instance $\mathbf{x}_i$ is assigned to a label y_j. This algorithm begins with the definition of an initial model $\theta^{(0)}$ and constructs a sequence $\theta^{(0)}$,$\theta^{(1)}$,...,$\theta^{(t)}$,... of models with increasing values of likelihood. Normally, the logarithm of the likelihood function is used for simplifying the calculation process:

$$l(\theta) = \log L(\theta) = \sum_i \log p(\mathbf{x}_i, y_i|\theta)$$

EM also uses an auxiliary function, which depends on the model and the distribution of missing labels, defined by:

$$Q(\theta|\theta^{(m)}) = \sum_i Q_i(\theta|\theta^{(m)}) = \sum_i E_{y_i|\mathbf{x}_i,\theta^{(m)}}[\log p(\mathbf{x}_i, y_i|\theta)]$$

$$Q(\theta|\theta^{(m)}) = \sum_i \sum_{j=1}^{k} P(y_i = j|\mathbf{x}_i, \theta^{(m)}) \log p(\mathbf{x}_i, y_i|\theta) \quad (1)$$

This function is used to increment the likelihood of the estimator θ as a maximum. Algorithm 2 shows the pseudo-code for EM.

The m-th iteration of the E-step (line 5) produces a guess of the $n \times k$ membership-probabilities of the elements to the clusters $\{\gamma_{ij}\} = P(y_i = j|\mathbf{x}_i, \theta^{(m)})$, where γ_{ij} is the guessed probability that sample $\mathbf{x}_i$ came from the j-th cluster. The M-step (line 6) gives a closed-form solution to the new estimates of the estimator θ. It converges if the increment of the likelihood is lower than a given value (δ).

Algorithm 2 Expectation Maximization Pseudo-code [40]

Input: A dataset of n elements $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. A convergence error value δ.
Output: A set of clusters $C = \{C_1, \ldots, C_k\}$ that divides X

1: Fix a number of cluster k {this value is estimated applying cross-validation and repeating this algorithm with different values of k}
2: Choose the initial model $\theta^{(0)}$.
3: Compute the initial log-likelihood $l^{(0)}(\theta^{(0)})$.
4: **repeat**
5: **E-step**: Calculate $\{\gamma_{ij}^{(m)}\}$ where $\gamma_{ij} = P(y_i = j|\mathbf{x}_i, \theta^{(m)})$. {For the m iteration}
6: **M-step**: Calculate $\theta^{m+1} = \text{argmax}_\theta \left(Q(\theta|\theta^{(m)})\right)$, where $Q(\theta|\theta^{(m)}) = \sum_i \sum_{j=1}^{k} \gamma_{ij}^{(m)} \log p(\mathbf{x}_i, y_i|\theta)$. {Extracted from equation 1}
7: **Convergence check**: Calculate $l^{(m+1)}(\theta^{(m+1)})$.
8: **until** $|l^{(m+1)} - l^{(m)}| < \delta$
9: Put $\mathbf{x}_i$ in C_j if $\gamma_{ij} = max(\{\gamma_{iq}\}_{q=1}^{k})$
10: **return** C

3.3 Spectral Clustering

Spectral clustering methods are based on a straightforward interpretation of weighted undirected graphs as can be seen in [9, 75, 77, 91]. The Spectral Clustering approach is based on a similarity graph that can be formulated in three different ways:

1. **The ϵ-neighbourhood graph**: all the components whose pairwise distance is smaller than ϵ are connected.
2. **The k-nearest neighbour graphs**: the vertex v_i is connected with vertex v_j if v_j is among the k-nearest neighbours of v_i.
3. **The fully connected graph**: all points with positive similarity are connected with each other.

The main problem is how to compute the eigenvector and the eigenvalues of the Laplacian matrix of these similarity graphs. For example, when large datasets are analysed, the similarity graph of the Spectral Clustering algorithm takes a significant amount of memory, this makes it hard to compute the eigenvalues and eigenvectors. Some works are focused on this problem: von Luxburg et al. [91] present the problem, Ng et al.[77] apply an approximation to a specific case, and Nadler et al.[75] apply operators to get better results. The classical algorithms can be found in [91].

The theoretical analysis of the observed good behaviour of SC is justified using the perturbation theory [75, 91], random walks and graph cut [91]. The perturbation theory explains, through the eigengap, the behaviour of Spectral Clustering.

Spectral clustering methods were introduced by Ng et al. in [77]. These methods apply the knowledge extracted from graph spectral theory to clustering techniques. These algorithms are divided into three main steps:

1. First, the algorithm constructs a graph using the data instances as nodes and applies a similarity measure to define the edges' weights (see Algorithm 3, line 1). Normally, the similarity measure used is the Radial Basis Function (RBF) Kernel defined by:

$$s(x_i, x_j) = e^{-\sigma||x_i - x_j||^2} \tag{2}$$

where σ is used to controlling the width of the neighbourhood.

2. Second, it studies the graph spectrum calculating the Laplacian Matrix associated with the graph (see Algorithm 3, lines 2 and 3). There are different definitions of the Laplacian Matrix. These definitions obtain different results when they are applied to the Spectral Clustering algorithm. They can be used to categorize the spectral clustering techniques as follows [91]:

   - **Unnormalized Spectral Clustering**: It defines the Laplacian matrix as:

   $$L = D - W$$

   - **Normalized Spectral Clustering**: It defines the Laplacian matrix as:

   $$L_{sym} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$$

   - **Normalized Spectral Clustering (related to Random Walks)**: It defines the Laplacian matrix as:

   $$L_{rw} = D^{-1} L = I - D^{-1} W$$

   In these formulas I is the identity matrix, D represents the diagonal matrix whose (i, i)-element is the sum of the similarity matrix i-th row and W represents the similarity graph (see Algorithm 3, line 2). Once the Laplacian is calculated (in Algorithm 3 the Normalized Spectral Clustering algorithm is used, however, in this case, to simplify, the eigenvalues which are calculated are $1 - \lambda_i$ instead of λ_i, the eigenvectors do not change), its eigenvectors are extracted (see lines 4 and 5 of Algorithm 3). Some of the main problems of Spectral Clustering are related to the consistency of the two classical methods used in the analysis: normalized and un-normalized Spectral Clustering. A deep analysis about the theoretical effectiveness of normalized clustering over un-normalized can be found in [92].

3. And, finally, the eigenvectors of the Laplacian matrix are considered as points and a clustering algorithm, such as K-means, is applied over them to define the clusters (see Algorithm 3, lines 7 and 8).

4 Genetic Algorithms for Clustering

Over the last years, the application of evolutionary algorithms to clustering has attracted much research interest, yielding a large literature corpus. Evolutionary Computation is a vast field that includes many families of algorithms, all of them inspired by natural selection. Perhaps the most popular is genetic algorithms (GAs), where a population of candidate solutions is encoded in strings named chromosomes. Then, GAs apply a set of genetic operators (typically mutation and crossover) and a stochastic selection operator based on a fitness function to breed the next algorithm iteration. Hruschka et al. [44] present a complete survey on this topic.

Although genetic algorithms have been traditionally used in optimization problems, these algorithms have also been used for general data and information extraction [34]. The operators of the genetic algorithms can also be modified. Some examples of these modifications can be found in [82] where Poli and Langdon improved the algorithm using backwards-chaining, creating and evaluating individuals recursively reducing the computation time. Other applications of genetic clustering algorithms can be found

Algorithm 3 Normalized Spectral Clustering according to Ng et al. (2001)[77]

Input: A dataset of n elements $X = \{x_1, \ldots, x_n\}$ and a fix number of clusters k.
Output: A set of clusters $C = \{C_1, \ldots, C_k\}$ which partitionate X

1: Form the affinity matrix $W \in R^{n \times n}$ defined by $W_{ij} = e^{-||x_i - x_j||^2/2\sigma^2}$ if $i \neq j$, and $W_{ii} = 0$.
2: Define D to be the diagonal matrix whose (i,i)-element is the sum of the i-th row of W.
3: Construct the matrix $L = D^{-1/2} W D^{-1/2}$.
4: Find $v_1, \ldots, v_k$, the k largest eigenvectors of L (chosen to be orthogonal to each other in the case of repeated eigenvalues) and form the matrix $V = [v_1 v_2 \ldots v_k] \in R^{n \times k}$ by stacking the eigenvectors in columns.
5: Form the matrix Y from V by renormalizing each row of V to have unit length (i.e. $Y_{ij} = V_{ij}/(\sum_j V_{ij}^2)^{1/2}$).
6: Apply K-means (or any other algorithm) treating each row of Y as a point in R^k.
7: Assign the points x_i to cluster C_j if and only if the row i of the matrix Y was assigned to cluster j.
8: **return** C

in swarm systems [50], software systems [31], file clustering [32] and task optimization [81], amongst others.

The GA-based clustering guides the clustering algorithm using different fitness functions to tune up the cluster selection process. In [23], Cole developed different approaches, especially focused on codification and clustering operations. There is also a deep revision in [44] where Hruschka et al. provided a complete up-to-date state-of-the-art in evolutionary algorithms for clustering.

There are several methods using evolutionary approaches from different perspectives, for example, Aguilar [6] modifies the fitness considering cluster asymmetry, coverage and specific information of the studied case; Tseng and Yang [90] use a compact spherical cluster structure and a heuristic strategy to find the optimal number of clusters; Maulik and Bandyopadhyay [59] use the clustering algorithm for metric optimization trying to improve the cluster centre positions; Shi et al. [86] based the search of the genetic clustering algorithm in their Extend Classifier Systems which is a kind of Learning Classifier System, in which a fitness of the classifier is determined by the measure of its prediction's accuracy; Das and Abraham [25] use Differential Evolution, a method that optimizes a problem by iteratively trying to improve a candidate solution with regards to a given measure of quality.

Some of those previous methods are based on K-means, for example, Krishna and Murty [49] replace the crossover of the algorithm using K-means as a search operator, and Wojciech and Kwedlo [96] also use differential evolution combined with K-means, where it is used to tune up the individuals obtained from mutation and crossover operators. Finally, other general results of genetic algorithm approaches to clustering can be found in [3]. There are also other complete studies for multi-objective clustering developed by Handl et al. [42] and for Nearest Neighbour Networks developed by Huttenhower et al. [45]. A few of these methods apply to different clustering approaches, a good example is the genetic graph-based clustering algorithm [66], which is based on spectral clustering.

4.1 MOGA for Clustering

Clustering can have multiple objectives and, in consequence, genetic algorithms will need to adapt to balance them. This idea leads to multiple works applying Multi-Objective Genetic Algorithms (MOGA) [28]. These approaches are characterized by the capability to use opposite objectives in the same fitness function. The evolution of the individuals defines a Pareto Front where the best fitness values –or dominant individuals– are found, according to the optimization metrics. These solutions are called **dominant solutions** and define a set of possible solutions to the problem.

MOGAs have been applied to several clustering problems [44]. There are usually two main approximations: some works generate a new MOGA to create a new clustering algorithm [74], while others apply classical MOGAs to solve the problem of minimizing some cost functions which are the objectives of the fitness function [52]. The most classical MOGAs are SPEA2 (Second version of the Strength Pareto Evolutionary Algorithm [102]), NSGA-II (Nondominated Sorting Genetic Algorithm [29]) and PESA [24] amongst others. These algorithms have been applied in clustering problems with different results [52]. NSGA-II and SPEA2 have demonstrated to achieve good results applied to clustering problems, however, SPEA2 usually defines a better Pareto Front than NSGA-II [102].

Some authors claim the superiority of this approach, for instance, Ripon and Kwong [84] stated that traditional single-objective algorithms suffer premature convergence that multi-objective algorithms solve. It is clear that sometimes using a single criterion loses important pieces of information that would be exploited for benefit of the search.

There are some proposals of MOGAs with an adaptive number of clusters. Handl et al. proposed the Multi-Objective Clustering with automatic K-determination (MOCK) [43], a graph-oriented clustering algorithm on a MOGA. In this approach, the chromosomes represent non-weighted graphs with an integer representation. Each loci represents a data instance and the allele a link to another instance. With this representation, a chromosome may contain several subgraphs, i.e., graphs without links to other graphs. These isolated subgraphs represent the clusters. Despite the graph-based representation, this approximation cannot be considered spectral clustering because of the lack of spectral analysis. Matake et al. proposed an improvement of MOCK [58] to compute k more efficiently and make the algorithm well suited for large datasets.

Another example of an adaptive k multi-objective clustering algorithm is the Variable-Length Real Jumping Genes Genetic Algorithm (VRJGGA), proposed by Ripon et al. [84]. VRJGGA is an adaptive version of another algorithm named JGGA. It uses a Variable-Length Genetic Algorithm with a standard cluster centroid representation in a chromosome of floats. The variance of chromosome lengths is introduced with two custom genetic operators: cut-and-paste and copy-and-paste.

On the contrary to previous partitional clustering algorithms, Banerjee [10] used a MOGA to solve the fuzzy clustering problem with adaptive k and noisy data. This approach uses a quite complex representation scheme with each individual divided into two independent strings: one distinguishes between clean and noisy data while the other one keeps the result of the partition.

There are also examples of multi-objective spectral clustering. One example is Wang [94], who proposed an evolutionary multi-objective spectral algorithm clustering algorithm for datasets that contain different views of the same data, for instance, because data come from heterogeneous sources. As a consequence, the dataset is represented by means of several graphs. In this context, the algorithm is able to automatically determine k by means of Pareto optimization.

There are only a few attempts to address spectral clustering with multi-objective computational intelligence. An example is to use Harmony Search Algorithm (HSA). This is a search method inspired by musicians improvisation that has an increasing number of applications. Li et al. proposed the Spectral Clustering-based Adaptive Hybrid Multi-Objective Harmony Search Algorithm (SCAH-MOHSA) [53], which is a complex algorithm for community detection in graphs; it uses spectral clustering with a Multi-Objective HSA and local search. Another second example is the Multi-Objective Genetic Graph-based Clustering Algorithm (MOGGCA) [63] that extended the GGC [66] algorithm to multiple objectives, improving the convergence. This algorithm was also extended to identify the number of clusters automatically by using a co-evolutionary approach [67].

5 Big Data and Online Clustering Techniques

One of the current main challenges in machine learning is the analysis of massive data online. Due to classification requires a previous labelling process, these methods need high efforts for real-time analysis. However, due to unsupervised techniques do not need this previous manual effort, clustering methods are a promising field for real-time analysis.

When data streams are analysed, it is important to consider the analysis goal, in order to determine the best type of algorithm to be used. We could divide data stream analysis into two main categories:

- **Offline analysis**: we consider a portion of data (usually large data) and apply an offline clustering algorithm to analyse it.
- **Online analysis**: the data are analysed in real-time. These kinds of algorithms are constantly receiving new data instances and are not usually able to keep past information.

5.1 Offline Clustering Analysis

The classical clustering processes used to analyse datasets have been adapted in order to improve their scalability and parallelism during the whole analysis process. This philosophy gain relevance with the Map-Reduce [27] algorithm and the Hadoop [89] framework, which implements this system. Current tools are based on Spark [99], which improves several features of Map-Reduce.

Map-Reduce is a model used to process and analyse large datasets using a parallel and distributed algorithm over a computer cluster. It is divided into two main steps:

- **The "Map" step**: The input is divided into smaller sub-problems. These sub-problems are concurrently processed by each cluster node. They may also be divided by generating a tree structure. Once a node has completed the process, it sends the answer to its master node.
- **The "Reduce" step**: The master node combines all the answers from the sub-problems to generate the final solution.

There are several adaptations of clustering algorithms to the Map-Reduce model, the most relevant ones are related to K-means [101], EM [21] and Spectral Clustering [20] algorithms.

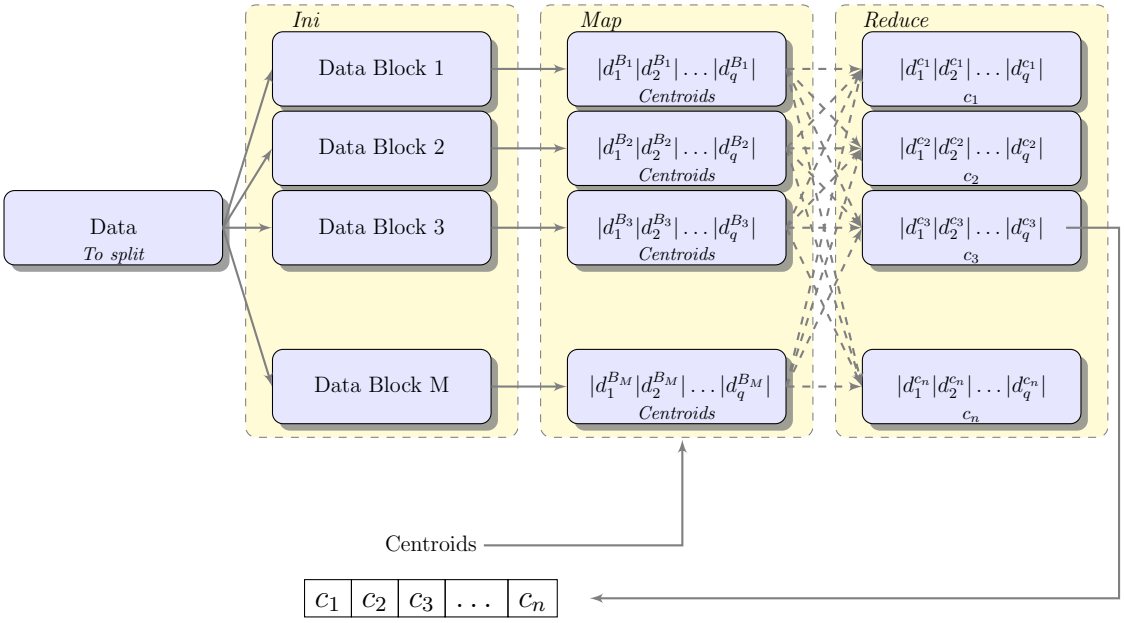

Figure 1: Map-Reduce Scheme for K-means.

5.1.1 K-means with Map-Reduce

K-means has been adapted to follow the Map-Reduce paradigm [101]. The algorithm focuses on the two main steps of K-means:

- Associate the data to the closest centroids.
- Calculate the cluster centroid.

The Map-Reduce version of the algorithm is divided into three main steps:

1. **Initialization**: The dataset is divided into blocks and the initial centroids are set.
2. **Data Association (Map)**: the data of each block is associated with the closest centroid. In this case, each node associates the data of one block and all nodes share the set of centroids.
3. **Centroids update (Reduce)**: Each node receives all the data,s which have been assigned to a centroid and updates the centroid position.

After step 3), the centroids set is updated and steps 2) and 3) are repeated until the algorithm converges or a maximum number of iterations is achieved (see Figure 1 for an example).

5.1.2 EM with Map-Reduce

Expectation-Maximization is divided into two steps [21], similarly to K-means. In this case, the algorithm works as follows:

1. **Initialization**: The data are divided into blocks.
2. **Expectation step (Map)**: Fixing the estimator (θ), the missing labels ($\boldsymbol{y}$) are estimated per block. In this case, all blocks share an estimator.

3. **Maximization step (Map and Reduce)**: Fixing the labels, the model will update the values according to the likelihood ($L(\theta)$) function. Part of this step might be calculated during the mapping process (e.g., partial operations which depend on the estimator). The updating process will be realized on the reduction process, producing new parameters for the Expectation-step.

A good example for the application is the Gaussian Mixture Model (GMM). In this case, we have the parameters μ (mean) and $\mathbf{\Sigma}$ (standard deviation) for the Gaussian sets, and we need to calculate the values of γ_{ij} (partial values of probability) for the M-step. Let $p^{(j)}(y) = \sum_{partial} \gamma_{ij}$, then the mapper will associate the data and calculate the partial sums for μ and $\mathbf{\Sigma}$ in the following way:

$$\mu^{(j)}_{partial} = \sum_{partial} (\gamma_{ij} \cdot x_i) \tag{3}$$

$$\mathbf{\Sigma}^{(j)}_{partial} = \sum_{partial} (\gamma_{ij} \cdot (x_i - \mu_j) \cdot (x_i - \mu_j)^t), \tag{4}$$

While the reducer will update the values as follows:

$$\mu_i = \frac{\sum_{blocks} \mu^{(j)}_{partial}}{\sum_{blocks} p^{(j)}(y)} \tag{5}$$

$$\mathbf{\Sigma}_i = \frac{\sum_{blocks} \mathbf{\Sigma}^{(j)}_{partial}}{\sum_{blocks} p^{(j)}(y)} \tag{6}$$

5.1.3 Spectral Clustering with Map-Reduce

The Spectral Clustering algorithm has also been successfully extended to the Map-Reduce paradigm [20]. The main motivation of this extension is the memory usage reduction (related to the similarity graph and the Laplacian matrix) and the eigenvectors computation. The Map-Reduce process is usually replicated during different steps of the algorithm.

The first step is the generation of the Similarity Graph and the Laplacian calculation. This step is parallelized including the generation of a sparse matrix which is helpful during the eigenvectors processing phase. The algorithm can be divided as follows (see Figure 2):

- **Similarity Graph (Map)**: the Similarity Graph is calculated comparing the data instances, this process is parallelized by sections.
- **Construct the sparse matrix (Reduce)**: the t-closest instances are chosen for each node creating a sparse matrix.
- **Laplacian Matrix Generation (Reduce)**: the Laplacian matrix is calculated using multiplications in order to keep the matrix in the distributed file system.

Once this first phase has been completed, it is needed to calculate the eigenvectors of the matrix. For this purpose, the algorithm used to identify the eigenvectors is an iterative algorithm called Lanczos Algorithm [35] (see Algorithm 4). This algorithm takes the Laplacian matrix and iteratively generates a triangular matrix called T (associated with a vector matrix called Q) which satisfies that each pair $< eigenvalue, eigenvector >$ of L (i.e., $(\lambda_k, Q_{n\times m}\vec{v}_k)$) corresponds to the pairs $< eigenvalue, eigenvector > (\lambda_k, \vec{v}_k)$ of T. This pair can be extracted using a QR iteration. In this case, the algorithm

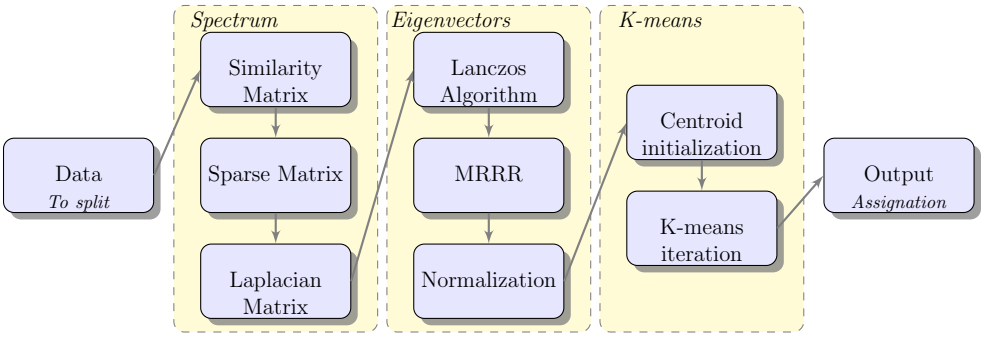

Figure 2: Map-Reduce process for Spectral Clustering.

Algorithm 4 Pseudo-code for Lanczos algorithm

Input: L Laplacian matrix.

Output: T tridiagonal matrix; Q matrix

1: $\vec{q}_0 = 0$; $\beta_0 = 0$; $\vec{q}_1 =$ random vector;
2: **for** $k = 1$ **to** M **do**
3: $\quad \vec{w}_k = L_{\vec{q}_k} - \beta_k \vec{q}_{k-1}$
4: $\quad \alpha_k = \langle \vec{w}_k, \vec{q}_k \rangle$
5: $\quad \vec{w}_k = \vec{w}_k - \alpha_k \vec{q}_k$
6: $\quad \beta_{k+1} = ||\vec{w}_k||_2$
7: $\quad \vec{q}_{k+1} = \vec{w}_k / \beta_{k+1}$
8: **end for**
9: **return** $Q_{n \times m} = (\vec{q}_1, \ldots, \vec{q}_m)$, $T = \begin{pmatrix} \alpha_1 & \beta_2 & 0 & \ldots & 0 \\ \beta_2 & \alpha_2 & \beta_3 & \ldots & 0 \\ 0 & \beta_3 & \ldots & \ldots & 0 \\ \vdots & \ddots & \vdots & \beta_{m-1} & 0 \\ \vdots & \ldots & \beta_{m-1} & \alpha_{m-1} & \beta_m \\ 0 & \ldots & 0 & \beta_m & \alpha_m \end{pmatrix}$.

called Multiple Relatively Robust Representation [18] (MR^3) is applied (see Algorithm 5). This algorithm extracts the eigenvectors of T.

The Lanczos algorithm uses an iterative method (lines 3 to 7) to generate a tridiagonal matrix (line 9) whose eigenvectors are related to the original matrix eigenvectors. The parallelization is performed in the matrix-vector product (line 3). The rest of the operations are computed locally.

Algorithm 5 Pseudo-code for MRRR algorithm

Input: T symmetric tridiagonal (irreducible). Γ_0 index of desired eigenpairs. *tol*: error tolerance (default 10^{-3}.

Output: $(\lambda_j, \vec{v}_j)$, $j \in \Gamma_0$ computed eigenvectors.

1: Split T into irreducible subblocks $T_1, \ldots, T_l$:
2: **for all** T_i, $i = 1, \ldots, l$ **do**
3: Choose μ_i and calculate L_0 and D_0 such that

$$L_0 D_0 L_0^t = T_i + \mu_i I,$$

is a factorization that determines the eigenpairs λ_j and $\vec{v}_j$, $j \in \Gamma_0$ to high accuracy
4: Compute the eigenvalues of $L_0 D_0 L_0^t$
5: Create a queue Q initialized as $Q = \{(L_0, D_0, \Gamma_0)\}$
6: **end for**
7: Recursively, while queue Q is not empty:
8: Remove an element (L, D, Γ): Partition the computed eigenvalues in nodes $\Gamma_1, \ldots, \Gamma_h$ according to the tolerance *tol*.
9: **for all** Γ_c, $c = 1, \ldots, h$ **do**
10: **if** $|\Gamma_c| = 1$ with eigenvalue λ_c **then**
11: Compute the eigenvector $\vec{v}_c$
12: **else**
13: Pick τ_c near the node and compute

$$LDL^t - \tau_c I = L_c D_c L_c^t$$

14: "Refine" the eigenvalues $\lambda_c - \tau_c$ in order to improve its acuracy with respect to $L_c D_c L_c^t$.
15: Set $\lambda_c = \lambda_c - \tau_c$ refined.
16: Add (L_c, D_c, Γ_c) to Q.
17: **end if**
18: **end for**

The MR^3 algorithm divides the tridiagonal matrix generated by the Lanczos algorithm (line 1). Then it generates a factorization to calculate the first eigenvector (lines 3-5). With this information, it generates a queue of factorization with the index of the eigenvalues (line 5). After, it recursively calculates the value of the eigenvectors using the factorizations generated. If the eigenvector can be calculated (lines 10 and 11) it is calculated, otherwise, it adds a new element to the queue (lines 13-16). This generates a tree of values which is represented in Figure 3.

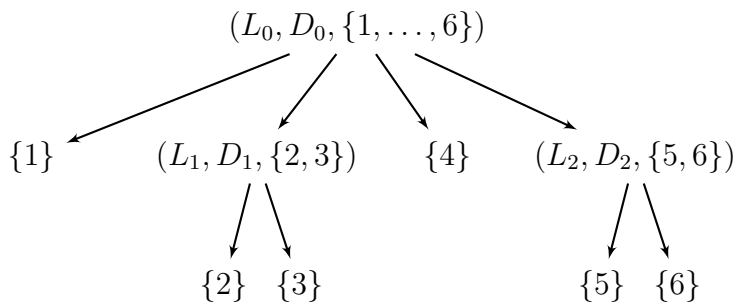

Figure 3: Tree representation of MRRR recursive algorithm.

Finally, once the k-first eigenvectors have been extracted, K-means (with Map-Reduce) is applied to the projection.

5.2 Online Clustering Analysis

The other important challenge of large data analysis is online analysis. Clustering methods have become promising techniques in this field because these algorithms can deal with unlabelled data. Usually, online analysis is focused on data streams.

Data streams generate data continually. The idea behind the online clustering algorithms is to analyse this data using real-time techniques. These techniques usually need to deal with large data quantities which produce several limitations on the algorithm representation. The most relevant limitations of these systems are:

- The data order matters and can not be modified.
- The data can not be stored or re-analysed during the process.
- The results of the analysis depends on the time the algorithm has been stopped.

These limitations also open new research lines, such as trend identification in the clustering behaviour. Due to the clustering results are constantly modified, the trend of the new results produce an interesting post-analysis phase which allows the analyst to predict where the clustering algorithm is going.

The main problem with these algorithms is that they need a specific space to update the information. This limits the possibilities of the new algorithm, producing for example, that medoid-based clustering algorithms could not be adapted to this kind of analysis [39].

One of the main tools used for online clustering analysis is the Massive On-line Analysis (MOA) tool. This framework provides the following online clustering algorithms:

- ClusTree [48]: This online algorithm iteratively updates the information of the clusters. It can consider the speed of the data stream generating the concept of "object's age". It also maintains stream summaries.
- CluStream [5]: This algorithm combines offline clustering and online clustering to provide partial clustering solutions, which measure the evolution of the clusters.
- Den-Stream [19]: This algorithm tries to identify shapes during the clustering process using a micro-clustering level. This algorithm keeps information of the micro-clusters composing the shape form of the shape-clusters.

All of these algorithms have been designed to identify more properties of the data stream than the final clusters in a specific moment. In this review, I have especially

focused on the classical online clustering implementation, which is described in the next section.

5.3 Classical Online Clustering

The classical online clustering implementation is based on the K-means algorithm [12]. It tries to reduce the cost function of K-means using a gradient descent approach.

The algorithm takes the following performance function as a starting point:

$$J = \sum_{i=1}^{N} \min_{j=1}^{k} ||x_i - c_j||^2 \tag{7}$$

Where x_i is a data instance, c_j is a centroid and the operation $||\cdot||$ is the norm. The equation represents that the algorithm tries to minimize this cost function. In order to perform this minimization, it calculates how the centroid should be minimized according to the gradient descent methodology.

Following this target, it calculates the closest centroid for each data instance as follows:

$$q^* = \arg\min_{q=1}^{k}(||x_i - c_q||) \tag{8}$$

Then, the algorithm updates each centroid using the gradient of the cost function, which is calculated as:

$$\nabla J = \frac{\partial J}{\partial c_{q^*}} = (x_i - c_{q^*}) \tag{9}$$

Using this information, the centroids are updated as follows:

$$c_{q^*}^{(new)} = c_{q^*} - \zeta(x_i - c_{q^*}) \tag{10}$$

Where ζ is the learning rate and is usually set to 0.05 or $1/N_i$, where N_i is the number of instances until that instance has appeared.

There are some variations of this algorithm. The most relevant are presented by Barbakh and Fyfe [12] which introduce three algorithms based on these ideas but using different cost functions. These algorithms are more robust according to the solution than the classical one.

6 Applications

Once the models are generated and validated, they are usually applied to a specific field. Some of the fields where I had applied different clustering approaches are:

- **Robotics**: In [46], classification and clustering techniques are used to identify team behaviours for robosoccer teams. The idea is to discriminate them according to different criteria. This work also studies the most successful behaviours during different robosoccer leagues.
- **Soccer**: [64] analyses the FIFA World Cup 2010. This work begins with a previous features selection analysis [62] which helps to determine the most relevant features for the clustering process. The clusters are used to identify different behaviours and are applied to determine how the Spanish strategy was successfully used during the 2010 World Cup.

- **Eurovision**: In [14] a genetic k-adaptive clustering algorithm for community detection is designed to identify different communities which vote in a similar way. This methodology tries to discover how the different countries of the Eurovision Contest Song formed alliances. This work was started in [15] where the clustering algorithm was not k-adaptive.
- **Baseball**: In [72] the idea was to predict baseball results. The model generated uses time-series clustering to include pass information of teams and matches.
- **Twitter**: In [68] the main goal is to discriminate meta-topics of different tweets, to group them. In this case, we use DBpedia and LSA to determine the tweet category and categorize those concepts which do not belong to DBpedia.
- **Image Segmentation**: [61] presents the application of a new genetic clustering algorithm to image segmentation. The results show that the algorithm performs the results of classical clustering algorithms.
- **Marketing**: [87] applies classification to analyse sentiment in Twitter. This analysis is used to identify the best classifier in this field. The analysis is extended in [88]. This last work also includes a social network analysis based on communities to measure propagation. Moreover, a parallel work is applied combining clustering [79] which is extended in [16].
- **Videogames**: [70] presents a new game called Dream which is an RPG-action game. Dream has been designed to extract data of different gameplays to analyse this data. The analysis is focused on the player evolution. It discriminates different player profiles according to their learning abilities. The model used to discriminate these profiles is based on time series clustering.
- **Quantum Key Distribution Networks**: In [38], we applied a medoid-based approach to the design of networks that distribute quantum keys. This approach was able to identify an optimum number of repeaters that need to be provided to serve a whole country.
- **Malware Detection**: There are several applications either for malware detection in Windows [8] and Android [56]. In [60], I present a review containing different machine learning methods applied to malware.

There are multiple applications of clustering methods to different machine learning problems, and there are going to be more, considering that these algorithms have a strong versatility and can deal with large amounts of data.

7 Conclusions

Clustering has multiple applications and can be used in several different contexts. This unsupervised technique has the ability to find patterns in data without human supervision. Currently, it can understand data under multiple representations and identifies patterns either related to their similarities or to the way the data connect. From the most classical algorithms such as K-means, to the most sophisticated online algorithms, clustering will be needed in the future for understanding massive amounts of data especially considering that human supervision scales very badly. The application of evolutionary computation to the clustering problem opens the door to creating more stable, scalable and robust algorithms, and it will become an important step forward for the future.

Acknowledgements

I would like to acknowledge this humble review to Gema Bello Orgaz, Victor Rodríguez and David Camacho, who are the main collaborators and friends that I have had during the development of all my work in machine learning.

License

References

[1] *Advanced Lectures on Machine Learning*, volume 33, 2012. Springer.

[2] Biomed central corpus, 2012. http://www.biomedcentral.com/about/datamining.

[3] K. Adamska. Cluster analysis of genetic algorithms results. *Inteligencia Artificial, Revista Iberoamericana de IA*, 9(28):25–32, 2005. doi: 10.4114/ia.v9i28.861. URL `http://iajournal.aepia.org/aepia/Uploads/28/283.pdf`.

[4] Charu C Aggarwal. *Data streams: models and algorithms*, volume 31. Springer, 2007.

[5] Charu C Aggarwal, Jiawei Han, Jianyong Wang, and Philip S Yu. A framework for clustering evolving data streams. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pages 81–92. VLDB Endowment, 2003.

[6] Jose Aguilar. Resolution of the clustering problem using genetic algorithms. *International Journal of Computers*, 1(4):237 – 244, 2007.

[7] Amir Ahmad and Lipika Dey. A k-mean clustering algorithm for mixed numeric and categorical data. *Data and Knowledge Engineering*, 63(2):503 – 527, 2007. ISSN 0169-023X. doi: 10.1016/j.datak.2007.03.016. URL `http://www.sciencedirect.com/science/article/pii/S0169023X0700050X`.

[8] Nadia Alshahwan, Earl T Barr, David Clark, George Danezis, and Héctor D Menéndez. Detecting malware with information complexity. *Entropy*, 22(5):575, 2020. doi: 10.3390/e22050575. URL `http://dx.doi.org/10.3390/e22050575`.

[9] Francis Bach and Michael Jordan. Learning Spectral Clustering, With Application To Speech Separation. *Journal of Machine Learning Research*, 7:1963 – 2001, oct 2006. URL `http://jmlr.csail.mit.edu/papers/volume7/bach06b/bach06b.pdf`.

[10] Amit Banerjee. An improved genetic algorithm for robust fuzzy clustering with unknown number of clusters. In *2010 Annual Meeting of the North American Fuzzy Information Processing Society*, pages 1–6. IEEE, jul 2010. ISBN 978-1-4244-7859-0. doi: 10.1109/NAFIPS.2010.5548175.

[11] Wesam Barbakh and Colin Fyfe. Clustering with reinforcement learning. In Hujun Yin, Peter Tino, Emilio Corchado, Will Byrne, and Xin Yao, editors, *Intelligent Data Engineering and Automated Learning - IDEAL 2007*, volume 4881 of *Lecture Notes in Computer Science*, pages 507–516. Springer Berlin / Heidelberg, 2007. ISBN 978-3-540-77225-5. URL http://dx.doi.org/10.1007/978-3-540-77226-2_52.

[12] Wesam Barbakh and Colin Fyfe. Online clustering algorithms. *International Journal of Neural Systems*, 18(03):185–194, 2008.

[13] Gema Bello, Héctor Menéndez, and David Camacho. Using the clustering coefficient to guide a genetic-based communities finding algorithm. In Hujun Yin, Wenjia Wang, and Victor Rayward-Smith, editors, *Intelligent Data Engineering and Automated Learning - IDEAL 2011*, volume 6936 of *Lecture Notes in Computer Science*, pages 160–169. Springer Berlin / Heidelberg, 2011. URL http://dx.doi.org/10.1007/978-3-642-23878-9_20.

[14] Gema Bello-Orgaz, Héctor D. Menéndez, and David Camacho. Adaptive k-means algorithm for overlapped graph clustering. *International Journal of Neural Systems*, 22(05):1250018, 2012. doi: 10.1142/S0129065712500189. URL http://www.worldscientific.com/doi/abs/10.1142/S0129065712500189.

[15] Gema Bello-Orgaz, Maria D. R-Moreno, David Camacho, and David F. Barrero. Clustering avatars behaviours from virtual worlds interactions. In *Proceedings of the 4th International Workshop on Web Intelligence; Communities*, pages 4:1–4:7, New York, NY, USA, 2012. ACM. doi: 10.1145/2189736.2189743. URL http://dx.doi.org/10.1145/2189736.2189743.

[16] Gema Bello-Orgaz, Héctor Menéndez, Shintaro Okazaki, and David Camacho. Combining social-based data mining techniques to extract collective trends from twitter. *Malaysian Journal of Computer Science*, 27(2), 2014.

[17] Michael W Berry and Malu Castellanos. Survey of text mining. *Computing Reviews*, 45(9):548, 2004.

[18] Paolo Bientinesi, Inderjit S Dhillon, and Robert A Van De Geijn. A parallel eigensolver for dense symmetric matrices based on multiple relatively robust representations. *SIAM Journal on Scientific Computing*, 27(1):43–66, 2005. doi: 10.1137/030601107. URL http://dx.doi.org/10.1137/030601107.

[19] Feng Cao, Martin Ester, Weining Qian, and Aoying Zhou. Density-based clustering over an evolving data stream with noise. In *SDM*, volume 6, pages 326–337. SIAM, 2006.

[20] Wen-Yen Chen, Yangqiu Song, Hongjie Bai, Chih-Jen Lin, and Edward Y Chang. Parallel spectral clustering in distributed systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(3):568–586, 2011.

[21] Cheng Chu, Sang Kyun Kim, Yi-An Lin, YuanYuan Yu, Gary Bradski, Andrew Y Ng, and Kunle Olukotun. Map-reduce for machine learning on multicore. *Advances in neural information processing systems*, 19:281, 2007. doi: 10.7551/mitpress/7503.003.0040. URL http://dx.doi.org/10.7551/mitpress/7503.003.0040.

[22] C.A.C. Coello, G.B. Lamont, and D.A. Van Veldhuisen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Genetic and evolutionary computation series. Springer Science+Business Media, LLC, 2007. ISBN 9780387367972.

[23] Rowena M. Cole. Clustering with Genetic Algorithms. Master's thesis, Nedlands 6907, Australia, 1998. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.27.9443`.

[24] David Corne, Joshua D. Knowles, and Martin J. Oates. The pareto envelope-based selection algorithm for multi-objective optimisation. In *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, PPSN VI, pages 839–848, London, UK, 2000. Springer-Verlag. ISBN 3-540-41056-2.

[25] S. Das, A. Abraham, and A. Konar. Automatic clustering using an improved differential evolution algorithm. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 38(1):218 –237, jan. 2008. ISSN 1083-4427. doi: 10.1109/TSMCA.2007.909595.

[26] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

[27] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[28] Kalyanmoy Deb and Deb Kalyanmoy. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, 1 edition, jun 2001. ISBN 047187339X.

[29] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: Nsga-ii. In *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, PPSN VI, pages 849–858, London, UK, 2000. Springer-Verlag. ISBN 3-540-41056-2.

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977. ISSN 00359246. doi: 10.2307/2984875. URL `http://web.mit.edu/6.435/www/Dempster77.pdf`.

[31] D. Doval, S. Mancoridis, and B. S. Mitchell. Automatic Clustering of Software Systems using a Genetic Algorithm. In *IEEE Proceedings of the 1999 Int. Conf. on Software Tools and Engineering Practice (STEP'99)*, pages 73–91, 1999. URL `http://citeseer.ist.psu.edu/doval98automatic.html`.

[32] V. Fernandez, R. G. Martinez, R. Gonzalez, and L. Rodriguez. Genetic algorithms applied to clustering. In *In Proceedings of the Winter Simulation Conference*, pages 1307–1314, 1997. URL `http://citeseer.ist.psu.edu/doval98automatic.html`.

[33] A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

[34] Alex A. Freitas. A review of evolutionary algorithms for data mining. In *Soft Computing for Knowledge Discovery and Data Mining*, pages 61–93, 2007. doi: 10.1007/978-0-387-09823-4_19. URL http://dx.doi.org/10.1007/978-0-387-09823-4_19.

[35] Roland W Freund, Martin H Gutknecht, and Noël M Nachtigal. An implementation of the look-ahead lanczos algorithm for non-hermitian matrices. *SIAM Journal on Scientific Computing*, 14(1):137–158, 1993. doi: 10.1137/0914009. URL http://dx.doi.org/10.1137/0914009.

[36] Colin Fyfe. Topographic maps for clustering and data visualization. In John Fulcher and L. Jain, editors, *Computational Intelligence: A Compendium*, volume 115 of *Studies in Computational Intelligence*, pages 111–153. Springer Berlin - Heidelberg, 2008. ISBN 978-3-540-78292-6. URL http://dx.doi.org/10.1007/978-3-540-78293-3_3.

[37] Colin Fyfe and Wesam Barbakh. Immediate reward reinforcement learning for clustering and topology preserving mappings. In Michael Biehl, Barbara Hammer, Michel Verleysen, and Thomas Villmann, editors, *Similarity-Based Clustering*, volume 5400 of *Lecture Notes in Computer Science*, pages 35–51. Springer Berlin - Heidelberg, 2009. ISBN 978-3-642-01804-6. URL http://dx.doi.org/10.1007/978-3-642-01805-3_3.

[38] Iván Garcia-Cobo and Héctor D Menéndez. Designing large quantum key distribution networks via medoid-based algorithms. *Future Generation Computer Systems*, 115:814–824, 2021.

[39] Sudipto Guha, Nina Mishra, Rajeev Motwani, and Liadan O'Callaghan. Clustering data streams. In *Foundations of computer science, 2000. proceedings. 41st annual symposium on*, pages 359–366. IEEE, 2000.

[40] Maya R. Gupta and Yihua Chen. Theory and use of the em algorithm. *Foundations and Trends in Signal Processing*, 4(3):223–296, 2010.

[41] Peter Haider, Luca Chiarandini, and Ulf Brefeld. Discriminative clustering for market segmentation. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '12, pages 417–425, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1462-6. doi: 10.1145/2339530.2339600.

[42] Julia Handl and Joshua Knowles. Evolutionary multiobjective clustering. In *In Proceedings of the Eighth International Conference on Parallel Problem Solving from Nature*, pages 1081–1091. Springer, 2004.

[43] Julia Handl and Joshua Knowles. Exploiting the Trade-Off - The Benefits of Multiple Objectives in Data Clustering. In Carlos A. Coello Coello, Arturo Hernández Aguirre, and Eckart Zitzler, editors, *Evolutionary Multi-Criterion Optimization*, volume 3410 of *Lecture Notes in Computer Science*, pages 547–560, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-24983-2. doi: 10.1007/b106458.

[44] E.R. Hruschka, R.J.G.B. Campello, A.A. Freitas, and A.C.P.L.F. de Carvalho. A survey of evolutionary algorithms for clustering. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 39(2):133 –155, march 2009. ISSN 1094-6977. doi: 10.1109/TSMCC.2008.2007252.

[45] Curtis Huttenhower, Avi Flamholz, Jessica Landis, Sauhard Sahi, Chad Myers, Kellen Olszewski, Matthew Hibbs, Nathan Siemers, Olga Troyanskaya, and Hilary Coller. Nearest Neighbor Networks: clustering expression data based on gene neighborhoods. *BMC Bioinformatics*, 8(1):250, 2007. ISSN 1471-2105. doi: 10.1186/1471-2105-8-250. URL `http://dx.doi.org/10.1186/1471-2105-8-250`.

[46] Guillermo Jiménez-Díaz, Héctor D. Menéndez, David Camacho, and Pedro A. González-Calero. Predicting performance in team games. In INSTICC Institude for systems, Control Technologies of Information, and Communication, editors, *ICAART 2011 - Proceedings of the 3ed International Conference on Agents and Artificial Intelligence*, volume 1, pages pages 401 – 406, 2011. ISBN 978-989-8425-40-3. URL `http://aida.ii.uam.es/wp-content/uploads/2011/06/icaart_2011.pdf`.

[47] Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2*, IJCAI'95, pages 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc. ISBN 1-55860-363-8. URL `http://dl.acm.org/citation.cfm?id=1643031.1643047`.

[48] Philipp Kranen, Ira Assent, Corinna Baldauf, and Thomas Seidl. The clustree: indexing micro-clusters for anytime stream mining. *Knowledge and information systems*, 29(2):249–272, 2011. doi: 10.1007/s10115-010-0342-8. URL `http://dx.doi.org/10.1007/s10115-010-0342-8`.

[49] K. Krishna and M. N. Murty. Genetic K-means Algorithm. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, 29(3):433–439, 1999.

[50] W.B. Langdon and R. Poli. Evolving problems to learn about particle swarm and other optimisers. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 1, pages 81–88, sept. 2005. doi: 10.1109/CEC.2005.1554670.

[51] Daniel T. Larose. *Discovering Knowledge in Data*. John Wiley & Sons, 2005.

[52] JungSong Lee, LimCheon Choi, and SoonCheol Park. Multi-objective genetic algorithms, nsga-ii and spea2, for document clustering. In Tai-hoon Kim, Hojjat Adeli, Haeng-kon Kim, Heau-jo Kang, KyungJung Kim, Akingbehin Kiumi, and Byeong-Ho Kang, editors, *Software Engineering, Business Continuity, and Education*, volume 257 of *Communications in Computer and Information Science*, pages 219–227. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-27206-6. doi: 10.1007/978-3-642-27207-3_22.

[53] Yangyang Li, Jing Chen, Ruochen Liu, and Jianshe Wu. A spectral clustering-based adaptive hybrid multi-objective harmony search algorithm for community detection. In *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pages 1–8. IEEE, 2012. doi: 10.1109/cec.2012.6253013. URL `http://dx.doi.org/10.1109/cec.2012.6253013`.

[54] David MacKay. *Information Theory, Inference and Learning Algorithms.* Cambridge University Press, 2003.

[55] J. B. Macqueen. Some methods of classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[56] Alejandro Martín, Héctor D Menéndez, and David Camacho. Mocdroid: multi-objective evolutionary classifier for android malware detection. *Soft Computing*, 21(24):7405–7415, 2017. doi: 10.1007/s00500-016-2283-y. URL `http://dx.doi.org/10.1007/s00500-016-2283-y`.

[57] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

[58] Nobukazu Matake, Tomoyuki Hiroyasu, Mitsunori Miki, and Tomoharu Senda. Multiobjective clustering with automatic k-determination for large-scale data. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation - GECCO '07*, page 861, New York, New York, USA, jul 2007. ACM Press. ISBN 9781595936974. doi: 10.1145/1276958.1277126.

[59] U Maulik. Genetic algorithm-based clustering technique. *Pattern Recognition*, 33(9):1455–1465, 2000. doi: 10.1016/s0031-3203(99)00137-5. URL `http://linkinghub.elsevier.com/retrieve/pii/S0031320399001375`.

[60] Hector Menendez. Malware: The never-ending arms race. *Open Journal of Cybersecurity*, 1(1):1–25, 2021.

[61] Héctor Menéndez and David Camacho. A multi-objective graph-based genetic algorithm for image segmentation. In *Innovations in Intelligent Systems and Applications (INISTA) Proceedings, 2014 IEEE International Symposium on*, pages 234–241. IEEE, 2014. doi: 10.1109/inista.2014.6873623. URL `http://dx.doi.org/10.1109/inista.2014.6873623`.

[62] Hector Menendez, Gema Bello-Orgaz, and David Camacho. Features selection from high-dimensional web data using clustering analysis. In *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics*, WIMS '12, pages 20:1–20:9, New York, NY, USA, 2012. ACM. URL `http://doi.acm.org/10.1145/2254129.2254155`.

[63] Héctor Menéndez, David F. Barrero, and David Camacho. A multi-objective genetic graph-based clustering algorithm with memory optimization. In *2013 IEEE Conference on Evolutionary Computation*, volume 1, pages 3174–3181, June 20-23 2013. doi: 10.1109/cec.2013.6557958. URL `http://dx.doi.org/10.1109/cec.2013.6557958`.

[64] Héctor Menéndez, Gema Bello-Orgaz, and David Camacho. Extracting behavioural models from 2010 fifa world cup. *Journal of Systems Science and Complexity*, 26(1):43–61, 2013. ISSN 1009-6124. doi: 10.1007/s11424-013-2289-9. URL `http://dx.doi.org/10.1007/s11424-013-2289-9`.

[65] Héctor D Ménendez. Varmog: A co-evolutionary algorithm to identify manifolds on large data. In *2019 IEEE Congress on Evolutionary Computation (CEC)*, pages 3300–3307. IEEE, 2019. doi: 10.1109/cec.2019.8790004. URL http://dx.doi.org/10.1109/cec.2019.8790004.

[66] Héctor D. Menéndez, David F. Barrero, and David Camacho. A genetic graph-based approach for partitional clustering. *Int. J. Neural Syst.*, 24(3), 2014.

[67] Héctor D Menéndez, David F Barrero, and David Camacho. A co-evolutionary multi-objective approach for a k-adaptive graph-based clustering algorithm. In *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pages 2724–2731. IEEE, 2014. doi: 10.1109/cec.2014.6900369. URL http://dx.doi.org/10.1109/cec.2014.6900369.

[68] Héctor D. Menéndez, Carlos Delgado-Calle, and David Camacho. Tweetsemminer: A meta-topic identification model for twitter using semantic analysis. In E. Corchado et al., editor, *Intelligent Data Engineering and Automated Learning - IDEAL 2014*, volume 8669 of *Lecture Notes in Computer Science*, pages 69–76. Springer International Publishing Switzerland, 2014. doi: 10.1007/978-3-319-10840-7_9. URL http://dx.doi.org/10.1007/978-3-319-10840-7_9.

[69] Héctor D Menéndez, Laura Plaza, and David Camacho. Combining graph connectivity and genetic clustering to improve biomedical summarization. In *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pages 2740–2747. IEEE, 2014. doi: 10.1109/cec.2014.6900370. URL http://dx.doi.org/10.1109/cec.2014.6900370.

[70] Héctor D. Menéndez, Rafael Vindel, and David Camacho. Combining time series and clustering to extract gamer profile evolution. In *Computational Collective Intelligence. Technologies and Applications - 6th International Conference, ICCCI 2014, Seoul, Korea, September 24-26, 2014. Proceedings*, pages 262–271. 2014. doi: 10.1007/978-3-319-11289-3_27. URL http://dx.doi.org/10.1007/978-3-319-11289-3_27.

[71] Héctor D Menéndez, Fernando EB Otero, and David Camacho. Sacoc: A spectral-based aco clustering algorithm. In *Intelligent Distributed Computing VIII*, pages 185–194. Springer, 2015.

[72] Héctor D Menéndez, Miguel Vázquez, and David Camacho. Mixed clustering methods to forecast baseball trends. In *Intelligent Distributed Computing VIII*, pages 175–184. Springer, 2015. doi: 10.1007/978-3-319-10422-5_19. URL http://dx.doi.org/10.1007/978-3-319-10422-5_19.

[73] O. du Merle, P. Hansen, B. Jaumard, and N. Mladenovic. An interior point algorithm for minimum sum-of-squares clustering. *SIAM J. Sci. Comput.*, 21(4): 1485–1505, dec 1999. ISSN 1064-8275. doi: 10.1137/S1064827597328327. URL http://dx.doi.org/10.1137/S1064827597328327.

[74] A. Mukhopadhyay, S. Bandyopadhyay, and U. Maulik. Clustering using multi-objective genetic algorithm and its application to image segmentation. In *Systems, Man and Cybernetics, 2006. SMC '06. IEEE International Conference on*, volume 3, pages 2678–2683, Oct. doi: 10.1109/ICSMC.2006.385268.

[75] Boaz Nadler, Stéphane Lafon, Ronald Coifman, and Ioannis G. Kevrekidis. Diffusion Maps, Spectral Clustering and Eigenfunctions of Fokker-Planck Operators. *Advance in Neural Information Processing Systems*, 18:955 – 962, 2006.

[76] G. Nathiya, S. C. Punitha, and M. Punithavalli. An analytical study on behavior of clusters using k means, em and k* means algorithm. *CoRR*, abs/1004.1743, 2010.

[77] A. Ng, M. Jordan, and Y. Weiss. On Spectral Clustering: Analysis and an algorithm. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems*, pages 849–856. MIT Press, 2001. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.19.8100`.

[78] Shintaro Okazaki, Kirk Plangger, Thomas Roulet, and Héctor D Menéndez. Assessing stakeholder network engagement. *European Journal of Marketing*, 2020.

[79] Gema Bello Orgaz, Héctor D. Menéndez, Shintaro Okazaki, and David Camacho. Extracting collective trends from twitter using social-based data mining. In *ICCCI*, pages 622–630, 2013.

[80] Alberto Pascual, Montserrat Barcéna, J.J. Merelo, and José-María Carazo. Application of the fuzzy kohonen clustering network to biological macromolecules images classification. In José Mira and JuanV. Sánchez-Andrés, editors, *Engineering Applications of Bio-Inspired Artificial Neural Networks*, volume 1607 of *Lecture Notes in Computer Science*, pages 331–340. Springer Berlin Heidelberg, 1999. ISBN 978-3-540-66068-2. doi: 10.1007/BFb0100500.

[81] P. Pokorný and P. Dostál. Cluster analysis and genetic algorithms. In *In: Management, Economics and Business Development in the New European Conditions*, pages 1–9, 2008. ISBN 978-80-7204-582-2.

[82] Riccardo Poli and William B. Langdon. Backward-chaining evolutionary algorithms. *Artificial Intelligence*, 170(11):953 – 982, 2006. ISSN 0004-3702. doi: 10.1016/j.artint.2006.04.003. URL `http://www.sciencedirect.com/science/article/pii/S0004370206000543`.

[83] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014. URL `http://www.R-project.org/`.

[84] K.S.N. Ripon and S. Kwong. Multi-Objective Data Clustering using Variable-Length Real Jumping Genes Genetic Algorithm and Local Search Method. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 3609–3616. IEEE, 2006. ISBN 0-7803-9490-9. doi: 10.1109/IJCNN.2006.247372.

[85] Dharmendra K Roy and Lokesh K sharma. Genetic kmeans clustering algorithm for mixed numeric and categorial data sets. *International Journal of Artificial intelligence and Applications(IJAIA)*, 1(2):23 – 28, 2010.

[86] Liang-Dong Shi, Ying-Huan Shi, Yang Gao, Lin Shang, and Yu-BinN Yang. Xcsc:: A novel approach to clustering with extended classifier system. *International Journal of Neural Systems*, 21(1):79 – 93, 2011. ISSN

01290657. URL http://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,cookie,url,uid&db=aph&AN=57325363&site=ehost-live.

[87] Mercedes Rozano Shintaro Okazaki, Ana M. Díaz-Martín and Héctor D. Menéndez-Benito. How to mine brand tweets: Procedural guidelines and pretest. *International Journal of Market Research*, 2014.

[88] Mercedes Rozano Shintaro Okazaki, Ana M. Díaz-Martín and Héctor D. Menéndez-Benito. Using twitter to engage with customers: A data mining approach. *Internet Research*, 2014.

[89] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The hadoop distributed file system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–10. IEEE, 2010.

[90] Lin Yu Tseng and Shiueng Bien Yang. A genetic approach to the automatic clustering problem. *Pattern Recognition*, 34(2):415 – 424, 2001. ISSN 0031-3203. doi: 10.1016/S0031-3203(00)00005-4. URL http://www.sciencedirect.com/science/article/pii/S0031320300000054.

[91] Ulrike von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, dec 2007. ISSN 0960-3174. doi: 10.1007/s11222-007-9033-z. URL http://www.kyb.mpg.de/publications/attachments/Luxburg06_TR_%5B0%5D.pdf.

[92] Ulrike von Luxburg, Mikhail Belkin, and Olivier Bousquet. Consistency of spectral clustering. *The Annals of Statistics*, 36(2):555–586, apr 2008. doi: this\%20http\%20URL. URL http://dx.doi.org/this%20http%20URL.

[93] Lin Wang, Minchu Jiang, Yinghua Lu, Minfu Sun, and Frank Noe. A comparative study of clustering methods for molecular data. *International Journal of Neural Systems*, 17(6):447 – 458, 2007. ISSN 01290657. URL http://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,cookie,url,uid&db=aph&AN=28225955&site=ehost-live.

[94] Xiang Wang, Buyue Qian, Jieping Ye, and Ian Davidson. Multi-objective multi-view spectral clustering via pareto optimization. pages 234–242. SDM, 2013. doi: 10.1137/1.9781611972832.26. URL http://dx.doi.org/10.1137/1.9781611972832.26.

[95] T Warren Liao. Clustering of time series data – a survey. *Pattern recognition*, 38 (11):1857–1874, 2005.

[96] Wojciech and Kwedlo. A clustering method combining differential evolution with the k-means algorithm. *Pattern Recognition Letters*, 32(12):1613 – 1621, 2011. ISSN 0167-8655. doi: 10.1016/j.patrec.2011.05.010. URL http://www.sciencedirect.com/science/article/pii/S0167865511001541.

[97] Rui Xu and Don Wunsch. *Clustering (IEEE Press Series on Computational Intelligence)*. Wiley-IEEE Press, illustrated edition edition, oct 2008. ISBN 0470276800. URL http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0470276800.

[98] Illhoi Yoo and Xiaohua Hu. A comprehensive comparison study of document clustering for a biomedical digital library medline. In *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '06, pages 220–229, New York, NY, USA, 2006. ACM. ISBN 1-59593-354-9. doi: 10.1145/1141753.1141802.

[99] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: cluster computing with working sets. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*, pages 10:1–7, 2010.

[100] Hui Zhang, Jason E Fritts, and Sally A Goldman. Image segmentation evaluation: A survey of unsupervised methods. *computer vision and image understanding*, 110(2):260–280, 2008. doi: 10.1016/j.cviu.2007.08.003. URL `http://dx.doi.org/10.1016/j.cviu.2007.08.003`.

[101] Weizhong Zhao, Huifang Ma, and Qing He. Parallel k-means clustering based on mapreduce. In *Cloud Computing*, pages 674–679. Springer, 2009.

[102] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report 103, Gloriastrasse 35, CH-8092 Zurich, Switzerland, 2001.