An Exploration of Methodologies to Improve Semi-supervised Hierarchical Clustering with Knowledge-Based -Constraints

By

Abeer Ahmed Aljohani

A doctoral thesis
Submitted in partial fulfillment of the requirements
for the award of

Doctor of Philosophy

Department of Computer Science

Loughborough University

October 2019

Supervisor: Prof. Eran Edirisinghe

Dr. Christian Dawson

Dr. Daphne Teck Ching Lai

Abstract

Clustering algorithms with constraints (also known as semi-supervised clustering algorithms) have been introduced to the field of machine learning as a significant variant to the conventional unsupervised clustering learning algorithms. They have been demonstrated to achieve better performance due to integrating prior knowledge during the clustering process, that enables uncovering relevant useful information from the data being clustered. However, the research conducted within the context of developing semi-supervised hierarchical clustering techniques are still an open and active investigation area. Majority of current semi-supervised clustering algorithms are developed as partitional clustering (PC) methods and only few research efforts have been made on developing semi-supervised hierarchical clustering methods. The aim of this research is to enhance hierarchical clustering (HC) algorithms based on prior knowledge, by adopting novel methodologies. Such prior knowledge is translated into triple-wise relative constraints, which can effectively be applied in hierarchical clustering. The research presented in this thesis contributes to: the proposal of a novel clustering algorithm taking into account six agglomerative linkage measures, with triple-wise relative constraints and the critical investigation of the performance of the algorithm with the use of various parameters integrating distance metrics, linkage methods and different levels of constraints; Enhancing the effectiveness of Constrained Ward's Hierarchical Agglomerative Clustering (CWHAC) algorithm by addressing the issues of constraint violation and redundancy and its efficiency by reducing the time-consuming process of generating constraints; development of a novel hybrid clustering approach for Constrained Ward's Hierarchical algorithm underpinned by the intelligent k-Means clustering algorithm (CWHC-IKM) for cluster initialization; to address the challenges of typical agglomerative clustering approaches; developing a novel framework to handle noise or irrelevant features named as, Constrained Weighted Ward Hierarchical Clustering algorithm based on intelligent K-means algorithm (CWWHC-IKM), which is designed to combine feature weighting approach with semi-supervised clustering. The thesis presents a rigorous performance analysis of the proposed novel Semi-Supervised Hierarchical Clustering (ssHC) algorithms proving their superiority in data clustering.

Abeer Aljohani, October 2019

Acknowledgement

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

In the name of Allah, the most Gracious and the Most Merciful Alhamdulillah, all honors to Allah for blessing and helping me complete this thesis. Special gratitude goes to my supervisor, Professor Eran Edirisinghe. I cannot find words to describe my appreciation for his guidance. I have been fortunate to be guided by a professor who cared deeply about my progress. His valuable insights, constructive criticisms and suggestions throughout the experiment and thesis writing have contributed immensely to the success of this study. Thank you Professor Eran Edirisinghe for all the help and support you have afforded me. Also, I would like to extend my sincere thanks to Dr. Daphne Teck Ching Lai, for her tremendous support and assistance with my research and thesis.

My deepest gratitude and affection goes to my dear husband, Nasreddin Alfarjani, who has held my hand throughout my PhD journey. I appreciate you for the unyielding love and support, encouragement, your confidence in me and for being a best friend. Thank you for everything.

My family deserves special recognition for their unrelenting support, love and for their prayers to Allah to help me achieve such success and honors during my PhD journey. Special appreciation to my father's souls, who I miss deeply and who taught me endurance and hard work and meaning of life. Although he is unable to partake of my success and happiness, I appreciate his input. Also, in a special way, I appreciate my mother, who is the most valuable gift I have in life. Special regards to my sisters, brothers and my son for their love, support and prayers.

Finally, my deepest gratitude to my sponsor (Taibah University, Saudi Arabia) for affording financial assistance and support throughout the research period.

Abeer Aljohani,October 2019

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AP	Anomalous Pattern
BSS	Between-Sum-of-Squares
CH	Calinski-Harabasz
CL	Cannot-Link
COP-COBWEB	Constraints- Partitioning COBWEB Clustering
COP-KMeans	Constraints- Partitioning K-Means Clustering
CWHAC	Constrained Ward's Hierarchical Clustering
CWHAC	Constrained Ward's Hierarchical Agglomerative Clustering
CWHAC- IP	Constrained Ward's Hierarchical Agglomerative Clustering- IPoptim
CWHAC-UT	Constrained Ward's Hierarchical Agglomerative Clustering- UltraTran
CWHC-IKM	Constrained Ward's Hierarchical Clustering Based on Intelligent K-Means
CWHC-IKM- IP	Constrained Ward's Hierarchical Clustering Based on Intelligent K-Means- IPoptim
CWHC-IKM- UT	Constrained Ward's Hierarchical Clustering Based on Intelligent K-Means- UltraTran
CWWHC-IKM- IP	Constrained Weighted Ward's Hierarchical Clustering based on intelligent K-means- IPoptim.
CWWp HC-IKM- IP	Constrained Weighted Wardp Hierarchical Clustering based on intelligent K-means- IPoptim.
CWWpB HC-IKM-IP	Constrained Weighted WardpB Hierarchical Clustering based on intelligent K-means-IPoptim method
EM	Expectation Maximization
HAC	Hierarchical Agglomerative Clustering
HC	Hierarchical Clustering
HCAKC	Hierarchical Clustering Algorithm based on K-means with Constraints
ik-means	Intelligent K-Means
IPoptim	Iterative Projection algorithm

ML	Must-Link
PC	Partitional Clustering
ssFCM	Semi-supervised Fuzzy c-Means
ssHAC	Semi-supervised Hierarchical Agglomerative Clustering
ssHC	Semi-supervised Hierarchical Clustering
ssPC	Semi-supervised Partitional Clustering
UltraTran	Modified Floyd-Warshall algorithm
WSS	Within-Sum-of-Squares

CHAPTER 1

An Overview

1.1 Introduction

Several application areas, such as biology, medicine, genetic studies of humans, and many other fields often require the separation of data into subgroups that are homogeneous, so that meaningful cluster analysis of the data can be carried out. Traditional computer-based data clustering methods are based on unsupervised learning, in which the analysis of clusters of data sets often involve no knowledge of relationships between the data sets and/or observations made by researchers [1].

Disregarding the prior knowledge during clustering is a major shortcoming of a clustering algorithm. It may lead to the algorithm not obtaining optimal partitions of data that can benefit the application. Dinler and Tural [2] argue that data could be more effectively clustered by integrating additional data-related information in the process of clustering, such as pairwise correlations among a couple of data points, so that the data could be better divided. This prompted the development of a novel type of semi-supervised clustering algorithms in the research field of machine learning. It is referred to as constrained clustering or clustering with side information. Such algorithms can extract pertinent information from the data by integrating prior knowledge into clustering [2]. Since prior knowledge has been shown to make clustering considerably more effective, the semi-supervised clustering algorithms have received the attention of a number of researchers [3]. This has led to endeavors to integrate constraints into Partitional Clustering (PC) and Hierarchical Clustering (HC), the two forms of standard clustering. The research conducted within the scope of this thesis is concerned solely on Semi-Supervised Hierarchical Clustering (ssHC).

In the remainder of this chapter, the research motivation, aims and objectives of the research conducted within the research context of this thesis are presented in sections 1.2 and 1.3 respectively, the key research contributions of the thesis are highlighted in section 1.4 and an overview of the thesis structure is presented in section 1.5.

1.2 Research Motivation

Vast quantities of data can be organised into smaller clusters that give greater clarity through the use of mechanisms for intuitive browsing and intuitive navigation when adopting clustering algorithms that are high quality and fast collections. However, large data samples need exploration and visualisation at different levels of granularity, which is made possible with HC solutions [4]. The problem is that previous or domain knowledge regarding the underpinning data configuration is disregarded by unsupervised HC algorithms in several applications. This in turn leads to the extraction of irrelevant structures from the data. Under these circumstances, ssHC algorithms constitute a viable option for integrating previous knowledge into the process of clustering.

The ssHC algorithms involve the use of Hierarchical Set Theory [1], [5], which represents the clustering results as dendrograms or trees. A tree is used to describe the organization of nested clusters, when clustering is defined as hierarchical, so that all data objects are contained within the tree root, and sub-clusters describe the union of cluster nodes with the exception of leaf nodes [6]. Divisions of clusters are more subjective and meaningful when HC is adopted [7]. HCs attempt to analyse large quantities of data using dendrograms that present data in the shape of a tree, so that data can be viewed and abstracted at various levels which are of great interest for a number of application domains. Due of the consistency of clustering solutions at different levels of granularity, flat partitions of different granularity can be extracted during data analysis making them ideal for interactive exploration and visualization. It can be noted that biological taxonomy and phylogenetic trees and other application domains involve sub-clusters within clusters, where solutions can be available when using HC analysis [8]. Sander, et al. [9] prove that clustering structures could be more easily detected by adopting HC methods, as this shows nested clusters, i.e. different point densities of clusters that represent less sensitivity, and cluster shapes that produce less influence on HC results. When using a partitioning algorithm, it is very difficult to detect nested clusters and different point densities of clusters that occur across various regions of data sets, which are important issues for many data sets involving real world information.

Far less research efforts have been allocated to investigate the advantages and disadvantages of the methodologies of ssHC algorithm. Further the majority of studies conducted have only been based on Semi-supervised Partitional Clustering (ssPC) [1], [5]. The HC analysis needs further investigations on overcoming the challenges that arise when applying semi-supervised approaches. The lack of development of semi-supervised hierarchical clustering algorithms can be attributed to many reasons. One reason is that most of the existing semi-supervised clustering methods focus on the use of background information in the form of so-called instance-level, Must-Link (ML) and Cannot-link (CL) constraints[5],[10],[11].These types of constraints are complex for HC, since HC merge all observations in a data set at some level of the clustering hierarchy. Consequently, a "CL" constraint will always be violated and likewise a "ML" constraint will always be satisfied at some level of the hierarchy [1]. Background knowledge can be interpreted by alternative ways such as by incorporating triple-wise relative constraints, which can effectively be applied within HC methods, since data patterns are connected over various levels of the hierarchy. In general, three instances (i.e. a triplet of instances a, b, c) are required in the definition of a triple-wise relative constraint. It conducts a comparison of similarity correlations between instance triplets a, b, and c, such that the distance between a and b (noted as d(a, b)) should be less than d(b, c), i.e. $d(b,a) < d(b,c)$) [11]. A further reason of shortage of development is that the clustering quality may be degraded as more data is joined [12] within a HC process. Moreover, by contrast to non-hierarchical clustering, HC necessitates more computational effort and memory space [13],[14], whilst also being susceptible to noise and outliers [15]–[17].

1.3 Thesis Aims and objectives

This study aims to put forth sets of clustered data with the highest possible similarity within clusters and the lowest possible similarity between clusters through developing enhanced and novel versions of the traditional ssHC algorithms.

In line with the above aim, this research proposes novel HC algorithms with knowledge-based, triple-wise relative constraints. The overall aim is achieved by achieving the main and sub-objectives described below.

1. Employ triple-wise relative constraint to create a novel semi-supervised enhancement of the well-known unsupervised Hierarchical Agglomerative Clustering (HAC) algorithm named as ssHAC, rigorously investigating the impact

of the selection of a range of standard clustering related parameters (i.e. number of constraints, linkage measures and distance metrics) on its optimal performance, when using different datasets.

To this end, three sub-objectives related to this objective are formulated.

1.1 Investigate the impact of different distance metrics and linkage measures in the optimal performance of the proposed ssHAC algorithm.

1.2 Assess the performance of ultra-metric dissimilarity matrix transformation methods of two constrained optimization methods (namely IPoptim and UltraTran) [11] in the context of the novel ssHAC algorithm based on the application of different number of triple-wise relative constraints, whilst using a range of distance measures and linkage techniques.

1.3 Determine the optimal selection of number of triple-wise relative constraints, linkage measures and distance metrics for a given dataset, investigating the nature of each dataset.

2. Develop a novel Constrained version of the well-known, Ward's Hierarchical Agglomerative Clustering (CWHAC) algorithm, thorough introducing novel approaches for enhancing both its effectiveness and efficiency.

In order to achieve this objective, two sub-objectives have been formulated.

2.1 Address the shortcomings (issue of non-satisfaction) of the use of triple-wise relative constraints with HC highlighted in the investigation of objective-1, to further improve the effectiveness of the novel CWHAC clustering algorithm.

2.2 Address the computational complexity of the process of generating constraints highlighted in the investigation of objective 1, to enhance the efficiency of the novel CWHAC clustering algorithm.

3. Integrate the well-known intelligent K-means (iK-means) clustering algorithm [18] with the CWHAC proposed under objective-2 to create a new hybrid ssHC algorithm referred to as constrained Ward's HC (CWHAC-IKM) to address the time complexity problem of agglomerative clustering.

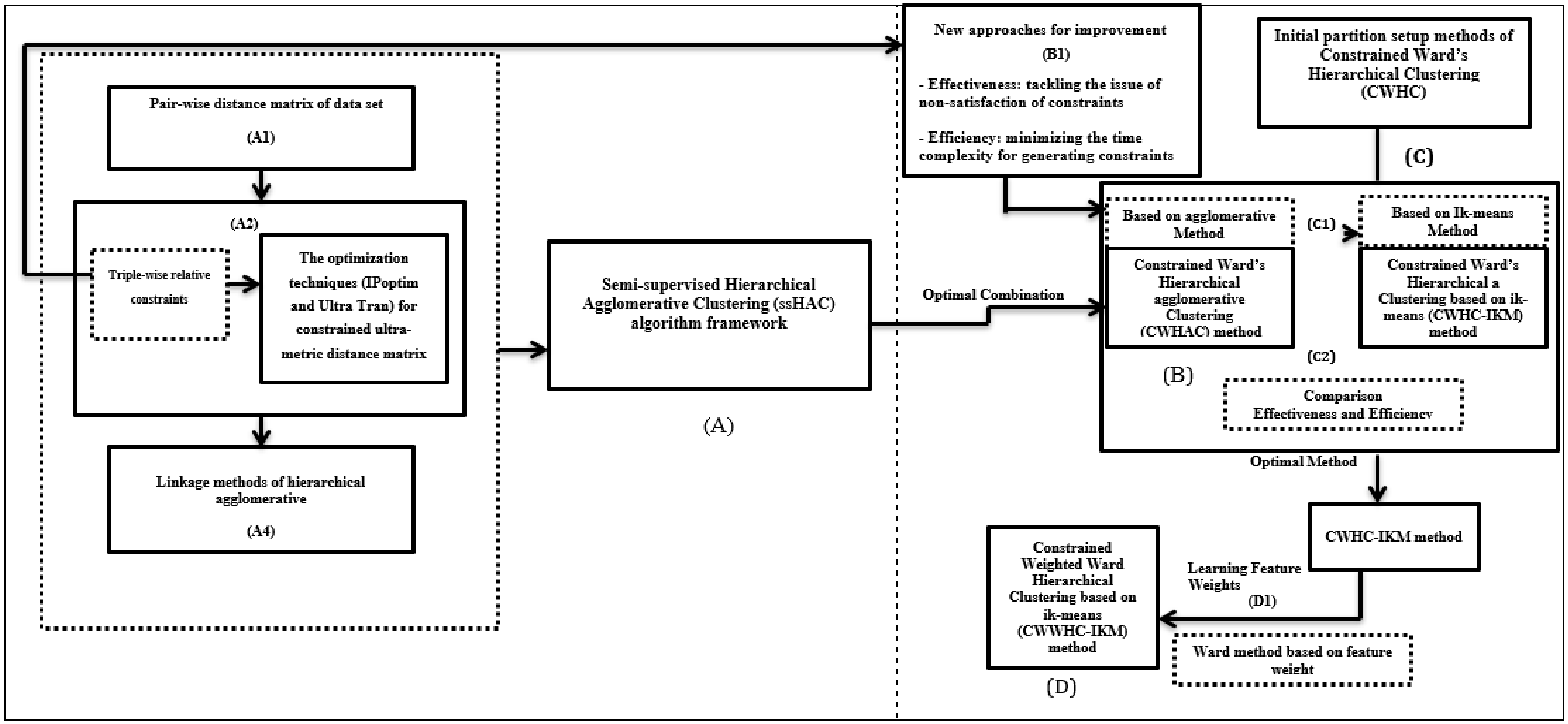
4. Rigorously investigate the application of feature weights in the training of the CWHC-IKM algorithm proposed under objective-3 to create further enhanced data clustering, by addressing the problem of irrelevant features and noise during the clustering process. It is referred to as Constrained Weighted Ward's HC (CWWHC-IKM).

The above objectives of the research conducted within this thesis have led to the following original contributions to the research field of data clustering.

1.4 Research Contribution

This thesis makes several original contributions to the field of ssHC algorithm. Given that a PC algorithm is the basis of the majority of semi-supervised clustering techniques proposed by earlier studies, the research presented in this thesis sought not only to formulate novel approaches to HC, but also to optimise the performance of these algorithms. Furthermore, to achieve better outcomes, prior knowledge in the form of triple-wise relative constraints are integrated into the proposed HC approaches.

The fundamental concepts of the adopted novel methodology in this thesis are depicted in figure 1.1. This diagram shows the main methodologies to develop new approaches for ssHC algorithm. (A) shows the first contribution which is the singular framework for a novel algorithm for ssHAC. The novel approaches for enhancing the effective and efficient Constrained Ward's Hierarchical Agglomerative Clustering Method are highlighted as the second contribution which is shown as (B). (C) shows the third contribution which proposes a novel Hybrid Constrained Clustering Algorithm (CWHC-IKM). By combining both (B) and (C), the comparison between the two approaches (CWHAC-and CWHC-IKM) also has resulted as the third contribution. (D) shows the application of feature weights in the training of the CWHC-IKM algorithm referred to as (CWWHC-IKM) which resulted in the fourth contribution.



Pair-wise distance matrix of data set
(A1)
(A2)
Triple-wise relative constraints
The optimization techniques (IPoptim and Ultra Tran) for constrained ultra-metric distance matrix
Linkage methods of hierarchical agglomerative
(A4)
Semi-supervised Hierarchical Agglomerative Clustering (ssHAC) algorithm framework
(A)
New approaches for improvement
(B1)
- Effectiveness: tackling the issue of non-satisfaction of constraints
- Efficiency: minimizing the time complexity for generating constraints
Optimal Combination
Based on agglomerative Method
Constrained Ward's Hierarchical agglomerative Clustering (CWHAC) method
(B)
Initial partition setup methods of Constrained Ward's Hierarchical Clustering (CWHC)
(C)
(C1)
Based on Ik-means Method
Constrained Ward's Hierarchical a Clustering based on ik-means (CWHC-IKM) method
(C2)
Comparison Effectiveness and Efficiency
Optimal Method
CWHC-IKM method
Learning Feature Weights
(D1)
Constrained Weighted Ward Hierarchical Clustering based on ik-means (CWWHC-IKM) method
(D)
Ward method based on feature weight

Figure 1. 1 Conceptual diagrams of the proposed contributions in the thesis.

The specific contributions of this thesis are as follows:

1.4.1 Developing a Novel ssHAC Algorithm supported by Triple-Wise Relative Constraints

Chapter 3 proposes a singular framework for a novel HAC algorithm based on a concept of semi-supervision of the clustering process considering the use of triple-wise relative constraints. The creation and performance optimization of this novel clustering algorithm draws on six commonly used linkages of agglomerative hierarchical method and ten distinct distance measures. Taking into account matters pertaining to decision-making and technical aspects, this contribution ultimately seeks to create a framework to facilitate the selection of the best mixture of linkages, distance measures and the number of constraints for the different datasets used in the context of the implementation of the proposed algorithm. Six widely used UCI datasets and one NTBC real dataset (see Appendix A) were employed for the analysis of the suggested framework when using various parameters, including linkage measures, distance metrics and different number of constraints. According to the experimental work that was conducted, the thesis shows that there exists dependence between the distance matrix factor and the linkage measure employed in the novel ssHAC algorithm.

Different dendrogram clustering outcomes can be generated by the proposed ssHAC algorithm when using different combinations of linkage measures with distance metrics. Nevertheless, in majority of the datasets experimented with, the best combination of parameters for the proposed ssHAC algorithm is proven to be the use of Ward linkage measure and Manhattan distance metric. Furthermore, the number of constraints is shown not always be associated with an exponential increase in clustering performance, but rather occasionally with a deteriorating performance, suggesting that not all constraints are useful. This warrants additional research to help constraints perform better in the context of an ssHAC algorithm.

The results related to this original research contribution has been presented in the conference paper entitled. “A Comparison of Distance Metrics in Semi-supervised Hierarchical Clustering Methods”, published by authors Aljohani, A., Lai, D.T.C., Bell, P.C. and Edirisinghe, E.A. at the International Conference on Intelligent Computing in 2017 [19].

1.4.2Proposing Performance Improvements (Enhancing of Effectiveness and Efficiency) to Constrained, Ward's Hierarchical Agglomerative Clustering Algorithm

A new approach semi-supervised HC algorithm is proposed in Chapter -4 as an enhancement to the well-known unsupervised Ward's HC algorithm named as Constrained Ward's Hierarchical Agglomerative Clustering (CWHAC). The approach effectively deals with the problem of non-satisfaction of triple-wise relative constraints to improve the effectiveness of CWHAC by managing the issue of constraint violation and redundancy, whilst also computationally simplifying the algorithm by speeding up the process of constraint generation based on three optimization principles to enhance the efficiency of CWHAC.

The results related to this research contribution have been presented in the conference paper titled, "An Effective and Efficient Constrained Ward's Hierarchical Agglomerative Clustering Method" and published by Aljohani, A.A., Edirisinghe, E.A. and Lai, D.T.C. in the Proceedings of SAI Intelligent Systems Conference in 2019 [20].

1.4.3 Creation of a Novel Hybrid Constrained Clustering Algorithm

The clustering algorithm proposed in Chapter -5 of constrained Ward's hierarchical clustering with intelligent K-means (CWHC-IKM) is a new hybrid semi-supervised clustering method incorporating triple-wise relative constraints that is capable of performing better and is computationally faster. This has been developed in order to resolve the problems associated with the conventional technique of initializing a HAC. The first step in this approach is the creation of an initial partition with a sufficiently large number of clusters. Thus, unlike in the conventional initialization technique, the process of cluster integration does not begin with a partition consisting constituting of only singletons, but with the created initial partition. Another goal of this study, besides formulation of a novel algorithm, is an in-depth comparative analysis of the CWHC algorithm based on two strategies of the initializing cluster, which are CWHAC (based on agglomerative setting) against CWHC-IKM (based on ik-means setting), in terms of how effective and efficient it is. According to the findings of empirical work, the suggested algorithm CWHC-IKM is computationally faster and highly effective in the case of the majority of datasets. Another feature that makes this study stand out is the use of internal and external measures to assess the outcomes of the constrained HC.

The above original work is being prepared as a part of a journal paper submission which will be submitted in due course to Elsevier – Knowledge Based Systems.

1.4.4 A Novel Approach for Feature Weight learning in Constrained Hierarchical Clustering

By integrating feature weighting techniques into semi-supervised clustering, a novel clustering algorithm named as Constrained Weighted Ward's hierarchical Clustering with Intelligent K-Means (CWWC-IKM) is proposed in Chapter-6. This algorithm permits the identification of concealed structures in the data being clustered, based on semi-supervised clustering associated with partly constrained patterns, whilst automatic determination of the weight of every feature is facilitated by a novel approach of feature weight selection considering that features can potentially have different relevance according to a given cluster. The automatic determination of feature weight is made possible by the application of Wardp [17] and WardpB [21] methods, which are underpinned by exponents of weighted Minkowski distance for the distance and feature weight. To choose the best combination of parameters for features and weights, a number of experiments have been conducted. This approach has been developed with the focus of dealing with noise and outliers that can affect the process of clustering.

This work will contribute to the journal publication being planned for Elsevier – Knowledge Based Systems, as described above.

1.5 An Overview of the Thesis

This thesis is presented as follows. Chapter 2 provides a review of the literature pertaining to clustering methods and research background on ssHC algorithms, techniques for relative weighting of features in the context of Ward’s methods, distance metrics and linkage measures associated with agglomerative clustering methods. Chapter 3 presents a novel algorithm for ssHAC that effectively utilizes the use of triple-wise relative constraints. Chapter 4 proposes a novel methodology to enhance the performance of CWHAC algorithm. Chapter 5 outlines a novel hybrid algorithm intended to overcome the challenges associated with traditional initialisation techniques related to constrained HAC, as well as extending a comparative analysis between the suggested hybrid algorithm CWHC-IKM and the CWHAC algorithm (based on the agglomerative strategy for initializing cluster). Chapter 6 presents a novel approach for learning feature weights in constrained HC, referred to as CWWHC-IKM that is intended to overcome irrelevant or unnecessary features during the clustering process. Finally, Chapter 7 concludes the research finding of the thesis giving an insight into future work.

CHAPTER 2

Background and Literature Review

This chapter introduces clustering algorithms in general and more specifically semi-supervised clustering methods, which are the main category of clustering algorithms this study centres on. The chapter additionally presents in detail the existing primary clustering approaches that underpin the novel clustering algorithms that will be presented in the subsequent contributory chapters of this thesis.

To better structure the information provided, this chapter is organised into a number of sections. In section 2.2, clustering methods are introduced, and the approaches and algorithms used as benchmarks and as a basis for the development of novel algorithms proposed in the present study are reviewed. In section 2.3, the metrics employed in the evaluation of the empirical results are detailed. In section 2.4, the existing semi-supervised clustering methods are presented, with an emphasis on constrained clustering. The semi-supervised HC algorithm providing the underpinning research of the models and algorithms developed in this study is also presented. In section 2.5, techniques for the selection of features and weighting of clustering algorithms are presented, with particular attention given to the available literature pertaining to the implementation of feature weighting for Ward's HC algorithm, while in section 2.6 and section 2.7, the cluster validity and evaluation methods and statistical analysis methods respectively employed in the empirical work are presented. Finally, in section 2.8, conclusions are made with an insight into the contributory work that will be presented in the subsequent chapters.

2.1 Introduction

According to Brea [22], the availability and usage of data have increased significantly over the recent years due to computer technology including mass storage media becoming widely available and affordable [22]. Demands have also increased for data or information to be processed and explored, which require automated computer-based approaches (e.g., data mining) to achieve these accurately and rapidly. Two basic fundamental approaches have traditionally been adopted to resolve these demands through data mining, i.e., Supervised Learning and Unsupervised Learning [2].

Classification is one data mining approach that is an example of a supervised learning method, which initially involves separating the dataset into training and test data. The process of supervision is formed by sets of labelled training data, so that a class label with a data point form the basis to select a fixed set of classes to support supervised classification techniques [23], [24].

The collection of labelled data involves intensive human effort and can be time-consuming, so this type of data is not widely available [2],[8]. They cannot be accurately assigned to respective classes, as their number is insufficient to teach accurate mapping due to limited availability of labelled data. Various studies report that to show feature value variability there needs to be enough training data to highlight differences between data pattern feature values in different classes and data pattern feature value variability within the same class of data patterns, so that discrimination between classes and accurate mapping is learned [8],[23]. Significant differences exist between unsupervised and supervised learning methods. An extremely popular example of the former is clustering, which is geared towards the identification of inherent "natural" structures in data that are not labelled [2].

The focus of research in this thesis is on data clustering. Hence the following sections presents the state-of-art in data clustering approaches.

2.2 Clustering

It was at the beginning of the 1930s that the foundations were laid for the research topic of investigation of ample volumes of data to derive relevant information regarding its group configuration based on how data entities are similar and how they are dissimilar. "Cluster" is the term given to a data group distinguished through exploratory analysis, while "cluster analysis" is the term for the collection of associated methods [25].

Clustering can be defined as a process whereby observations without labels are divided into clusters with similarities between observations in the same cluster and differences between observations in different clusters. Data analysts use the similarity criteria to organise data into helpful clusters in line with the requirements of the practical task that requires the data to be clustered [2].

Figure 2.1 illustrates an example of the clustering process that was proposed by Capaldo and Collova [26]. A set of samples and a measure of inter-sample similarity/ dissimilarity constitute the cluster analysis input, while a number of clusters creating a division or a framework of divisions of the dataset constitute the cluster analysis output. Data must be pre-processed prior to implementation of clustering on the dataset [26]. There are four stages of pre-processing. The first stage is data cleaning, whereby gaps in values are covered, noisy data are smoothed, outliers are detected or eliminated, and ambiguities are addressed. The second stage is data integration, whereby a number of different databases, data cubes or files are integrated. The third stage is data transformation, whereby data are normalised and pooled. The fourth stage is data reduction, whereby representation in terms of number is diminished, yet the analytical outcomes generated are identical or similar.

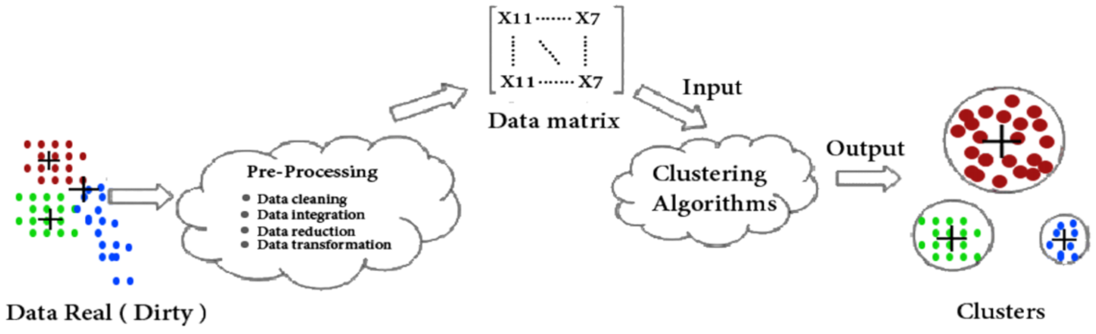


Figure 2. 1. Steps of clustering procedure [26].

Data comprehension has been approached via cluster analysis in numerous contexts, and therefore this technique has a wide range of applications, including image fragmentation [27], [28], biological processes [29], aggregation of associated genes from data pertaining to gene expression [30], social networks [31], business intelligence [32], as well as creation of models of disease processes [33]. In cases of analysis of breast cancer data and other biomedical datasets, it is possible to use clustering algorithms on congregated data patterns of greatest similarity and for group validation, particularly in cases in which labels are not known or are not enough. This can make it easier for professionals to identify how many relevant subgroups. Subsequently, class labels of relevance can be allocated by classification methods [23].

There are two major types of clustering algorithms, namely, HC algorithms and PC (also as known flat) algorithms. The latter involve the separation of a series of data items into a number of clusters so that every item is in solely a single subset, thus resulting in a series of clusters that are not specifically correlated or hierarchical. In formal terms, for any dataset D with n items and k number of clusters to create, the items are arranged by the partitioning algorithm into k partitions ($k \leq n$), with every partition constituting a cluster [34]. Meanwhile, hierarchical algorithms separate data into clusters that form a hierarchy and share their content between themselves. Unlike partitional algorithms, which generate separate clusters, hierarchical algorithms provide richer information due to organising clusters into a hierarchy [35].

The algorithms that are used as a basis in the development of novel and improved data clustering approaches presented in this thesis, namely, the K-means algorithm and HC, are presented in detail in the following sub-sections.

2.2.1 K-means Algorithm

The K-means algorithm is a popular technique of PC whereby initial pre-established K cluster centroids are chosen and the nearness from every point to each K centroid is determined by reducing the sum of squared error (SSE) as much as possible, as indicated in equation (2.1) [36].

$$\mathrm{SSE} = \sum_{i=1}^{k}\sum_{j=1}^{N_i}\left\| X_j - m_i \right\|^2 \tag{2.1}$$

where the number of data patterns pertaining to cluster i is denoted by N_i, the number of clusters is denoted by k, and the distance between point X_j and centre m_i is denoted by $\| X_j - m_i \|$.

The SSE criterion is diminished locally by the K-means algorithm, which is classified as an iterative algorithm as it takes a hyper-spherical structure for every cluster. The purpose of the K-means algorithm is to allocate every data point to clusters with the closest centroid or mean. To this end, the algorithm starts with centroids, detecting the closest centroid and allocating every point that remains, thus updating the cluster centroids. This procedure is iterated until the K centroids no longer shift or converge [37]. However, the K-means algorithm cannot find a universal optimum but only a local one, different runs of K-means on the same input data can give different output. This happens because different points of starting centroid may result in different clusters being generated. Therefore, selection of initial cluster centres for this algorithm is challenging [37]–[40].

2.2.1.1 Intelligent K-means Algorithm

Mirkin [18] propose the intelligent K-means (iK-means) algorithm in order to address the limitations of the K-means algorithm. Through iK-means, the initial centroids for K-means can be automatically identified based on anomalous pattern (AP) clusters [40], [41]. The non-grouped object of data located the farthest from the initial centre of gravity become, one at a time, tentative centroids. Subsequently, the cluster is filled with the objects nearer to the tentative centroid than to the center of gravity itself. Once the process of entity clustering is complete, the algorithm discards small groups using a pre-established threshold. More specifically, "anomalous" clusters are detected and eliminated successively from the dataset by the ik-means algorithm. Instead of being pre-specified, the number of clusters is determined on the basis of threshold θ (that is the smallest number of pieces of data needed to create a group) which is used to determine the final number of clusters. The number of anomalous clusters k* to be always larger than the number of actual clusters at a threshold value of 1. As indicated by the expression below, alternative minimisation enables the ik-means algorithm to determine the present anomalous cluster S_t and corresponding centroid C_t [21]:

$$W(S_t, C_t) = \sum_{i \in S_t} d(y_i, C_t) + \sum_{i \notin S_t} d(y_i, 0) \qquad (2.2)$$

where C_t represents the centroid of the cluster S_t, $W(S_t, C_t)$ is sum of square of distance of all points belonging to cluster S_t from its centroids and the square of distance of all the other points of data from gravity center of data, $d(y_i, C_t)$ represents the square of Euclidean distance between point y_i and centroid C_t and $d(y_i, 0)$ denotes the square of Euclidean distance between point y_i and centroid of data. The steps of the ik-means algorithm are outlined below [21].

Algorithm 2.1. The Intelligent K-means Method (iK-means) algorithm for detection of anomalous clusters [21].

```
1. Initial setting. Set the user-defined $\theta$. Set the centroid $C_Y$ to be the
   component-wise mean of $y_i, \in Y$.
2. Tentative centroid. Set $S_t = \emptyset$. Set $c_t$, a tentative centroid, to coincide with
   the entity $y_i, \in Y$ that is farthest from $C_Y$ according to equation (2.2).
3. Entity assignment. Assign each entity $y_i, \in Y$ to either $C_t$ or to $C_Y$
   depending on which is the nearest. Those assigned to $C_t$ form the cluster $S_t$.
   If there are no changes in $S_t$, go to Step 5.
4. Centroid update. Update $C_t$ to the component-wise mean of $y_i \in S_t$. Go to
   Step 3.
5. Save centroid. If $|S_t| \geq \theta$, include $c_t$ into C.
6. Remove clusters. Remove each $y_i, \in S_t$ from Y. If $|Y| > 0$, go to Step 2.
Cluster. Run k-means on the original data set Y, using as initial centroids those
in C.
```

2.2.2 Hierarchical Clustering

The HC method creates a family of clusterings, not just one. As explained by Johnson [42], the clustering of a series of items is achieved by this technique on the basis of a set of sequential integrations. Data are grouped into a tree of clusters known as a dendrogram, as shown in Figure 2.2. This structure represents a hierarchy of nested clusters assembled either top-down or bottom-up. Constituting a single cluster, the tree root comprises the entirety of the data points, whilst n clusters are represented by the tree leaves, with a single data point in every leaf. The data points are clustered into disjointed groups when the tree is cut at a particular level [43].

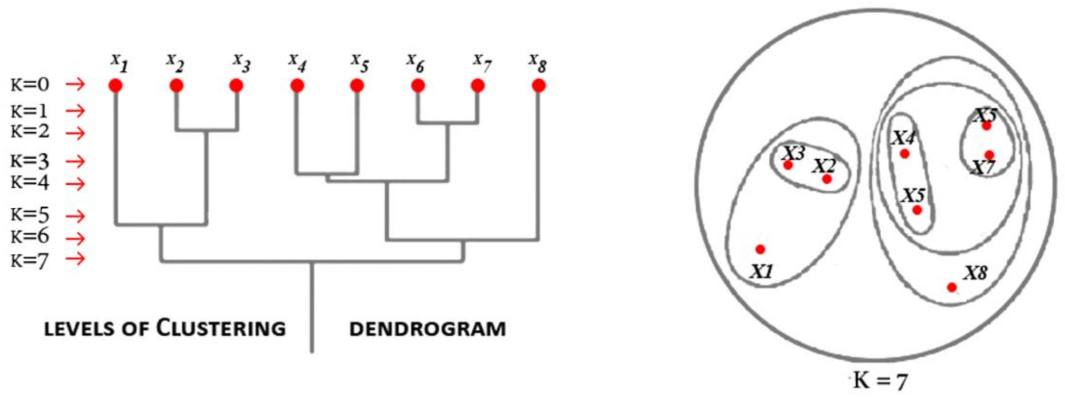


Figure 2. 2. The process of hierarchical cluster formation [26].

The hierarchical representation of dataset items in a dendrogram can be achieved via divisive and agglomerative types of the HC algorithm [44], [45]. The dissimilarities between these types in terms of how clustering is performed on a dataset of five items {a, b, c, d, e} are illustrated in Figure 2.3. The divisive type adopts a top-down procedure, starting with a single cluster encompassing the entirety of the items. This cluster is sequentially divided into two smaller portions until fulfilment of a cessation criterion. On the other hand, the agglomerative type of HC algorithm involves a bottom-up procedure, starting with n clusters, each of which includes a single item. Integration of two “close” clusters is performed at every repetition until fulfilment of a cessation criterion [46]. The algorithm commences with separate data items in their particular cluster, with equivalence between cluster distances of data items and their differences. The linkage criterion is used to merge between the closest clusters with the update of inter-cluster distances. The procedure terminates when every data item attains the highest hierarchy level or become aggregated in a single cluster [46].

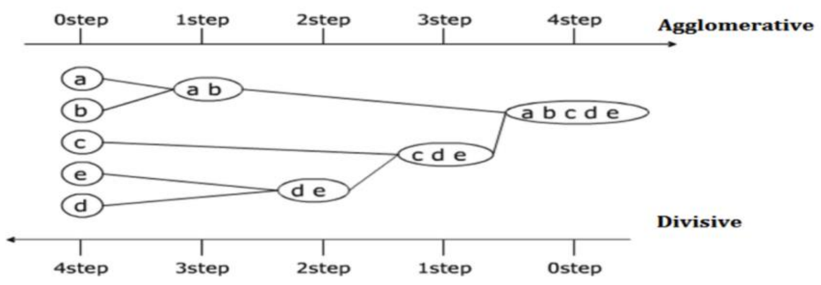


Figure 2. 3. The application of agglomerative and divisive clustering on a dataset [26].

To determine how similar a data pattern and a linkage criterion of data patterns are, various measures can be used. In the proposed work, the use of ten most common distance metrics (see Section2.2.3) are investigated and six-linkage measures of the agglomerative form of HC are applied (see Section 2.2.2.1).

With regards to the divisive form of HC, it has attracted much less research attention. Its main applications are in linguistics, information extraction, and document clustering [47]. The first step of the standard divisive clustering process is introduction of all data objects in one cluster, followed by the selection of the data object whose average dissimilarity from all the other objects is the largest. From a computational perspective, this second step is the costliest, being characterised by (N2) complexity [48]. A particular strength of the agglomerative algorithm is that it permits analysis of the developed cluster hierarchy to establish the ideal number of clusters, as it does not necessitate that the final number of clusters be pre-established. Moreover, the agglomerative algorithm is compatible with clusters with either regularly and irregularly forms [49]. Given these positive features, the agglomerative algorithm was chosen in the present work as the form of HC one which the algorithms presented in this thesis are based on.

2.2.2.1 Linkage Measure of Agglomerative Methods

Clustering exposes problems that involve more than one case, as it is not possible to simultaneously calculate three or more pairs of scores, as calculations traditionally calculate individual pairs of scores when adopting the squared Euclidean distance. In the case of the proximity matrix, there is a need to complete calculations of score differences between cluster pairs, but there is no single value for each variable in clusters. This means that a methodology is required to measure accurate distances between cluster pairs for every variable, when a cluster or clusters have more than one case [50]. For instance, for a N x N distance matrix of N items in the dataset, the pairs of items that are closest to one another should be grouped together in one cluster. However, an issue arises as the N x N matrix must be updated to an (N-1) x (N-1) matrix as the two items that were integrated and introduced in the same cluster cannot be further divided. Under such circumstances, the distance between the non-integrated N-2 items and the new item emerging from the previous integration can be determined by adopting the linkage technique to find out how different a data point or group of data points is from another group or how close they are to each other. There are several different linkage measures in order to merge two clusters that are close together. Each type of linkage has a different way of measurement. Thus, the procedure of integrating clustering together would be directly influenced through selecting of linkage measure [50], [51]. According to Gan, Ma and Wu [49], popular agglomerative HC algorithms (see Figure 2.4) could be applied to determine how close two clusters are. The short introductions to the six linkage measures are provided below.

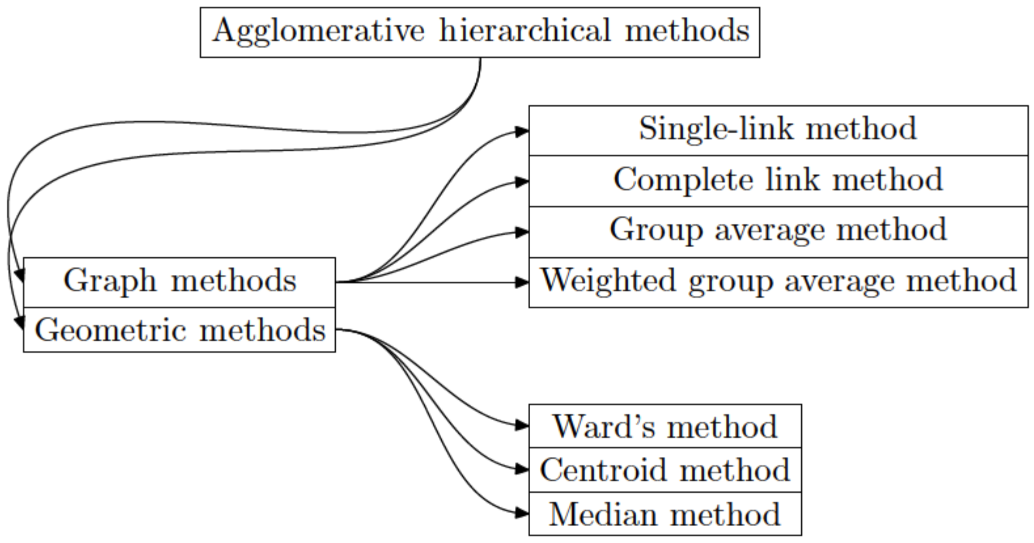


Figure 2. 4.Commonly used agglomerative hierarchical methods [49].

Single Linkage

This linkage approach is commonly known as the minimum method [42] or the nearest neighbour cluster analysis method [52]. It involves the measurement of the distance among two groups as the smallest distance between two points from the first and second clusters, respectively [53]. This means that, in the case of a group containing points a and b, and a group containing points c, d and e, the minimum distance between the point pairs (a, c), (a, d), (a, e), (b, c), (b, d) and (b, e) is the distance between the two groups [50]. The formula for the single linkage is [54]:

$$D_{SL}(p, q) = \min_{x_i \in p, x_j \in q} \{d(x_i, x_j)\} \tag{2.3}$$

where D is a distance between two clusters p and q; d is the distance function by which the dissimilarity matrix is computed between two points x_i, x_j.

The advantage of single linkage measure is its ability to manage cluster shapes of relatively high complexity as well as non-elliptical shapes. However, it exhibits sensitivity to noise and outliers [44], [45], [55].

Complete linkage.

The Complete Link Clustering method is also called the "maximum method," [42]; or the "furthest neighbour cluster analysis method" [52]. It is employed to measure the distance between two groups as the maximum (furthest) distance found between one point from the first cluster and one point from the second cluster [56]. It is defined as [54]:

$$D_{CL}(p, q) = \max_{x_i \in p, x_j \in q} \{d(x_i, x_j)\} \tag{2.4}$$

where the distance between clusters p and q is denoted by D, while the distance function underpinning calculation of the dissimilarity matrix between the points x_i and x_j is denoted by d.

This technique is impacted by convex shapes [16] and displays sensitivity to outliers [25], [50]. The integration of nearby clusters is hindered by outlying cases as the effects of outlying data are worsened by the measure of the remotest neighbour. For instance, if a, b, c and d from the previous examples are close to each other according to the pre-set series of variables, but e is markedly dissimilar from the others, then the score dissimilarities between (a, e) and (b, e) will prevent the integration of cluster 1 and cluster 2 [50]. Meanwhile, owing to the occurrence of the outlier pattern, which may hinder the integration of a cluster to other members of the same cluster in future repetitions, distances between clusters will significantly increase at a subsequent repetition if a cluster is integrated with an outlier at a particular repetition [25].

Average linkage

Sokal and Michener [56] introduced to overcome the limitations of single and complete linkage by proposing the Average linkage approach. Due to integrating information about the variance of the distances, the average distance value is more accurate than the distance between two clusters of cases [50].

Yim and Ramdeen [50] define the average linkage as the distances between each case in the first group and every case in the second group, averaged. For example, "the distance between cluster 1 (including points a and b) and cluster 2 (includes points c and d) would be the average of all distances between the pairs of cases listed as: (a,c), (a, d), (a, e), (b, c), (b, d), and (b, e) [50]. It is defined as [6]:

$$\text{proximity}(C_{i,}C_j) = \frac{\sum_{\substack{x\in C_i \\ x\in C_j}} \text{proximity}(x, y)}{m_i * m_j} \tag{2.5}$$

where the cluster proximity (C_i, C_j) of clusters C_i and C_j, which are of size m_i and m_j respectively.

The average linkage approach uses average pairwise distances between members to merge clusters, and this criterion forms an effective compromise, but remains susceptible to outliers and noise [25], and is biased towards globular clusters [6].

Median

The method identifies observations in a cluster and observations in another cluster and calculates the median distance between these. This method uses the median instead of the mean that used in the averaging technique. Thus, it reduces the effects of outliers by downweighting [57].

Centroid Method

According to Miyamoto and Terami [58], the distance between two mean vectors of clusters equates to the distance between two clusters, so that two clusters are combined at each stage of the process that have the smallest centroid distance, and is defined as:

$$d(G, G') = \| \; M(G) - M(G')\|^2 \tag{2.6}$$

where the clustering is denoted by G, G', while the mean vectors of the clusters are denoted by M(G) and calculated based on the formula:

$$M(G) = \frac{1}{|G|}\sum_{x \in G} x_K \tag{2.7}$$

In contrast to Average, Complete and Single linkage approaches, cluster distances do not demonstrate greater monotony with increased iterations in centroid linkage method, which is considered to be an important characteristic [25].

Ward's Method

Referred also as the minimum variance method [59], Ward's method is the most common technique of hierarchical agglomerative cluster analysis and involves integration at each iteration of the two clusters that ensure the smallest intra-cluster variance. The calculation of this variance is based on the weighted sum of squares, considering the cardinality of every cluster and resulting in the cost function below [17], [59]:

$$\mathrm{Ward}(\mathrm{S_i}, \mathrm{S_j}) = \frac{\mathrm{N_{S_i}} \mathrm{N_{S_j}}}{\mathrm{N_{S_i}} + \mathrm{N_{S_j}}} \, \mathrm{d}\left(\mathrm{C_{S_i}}, \mathrm{C_{S_j}}\right)^2 \tag{2.8}$$

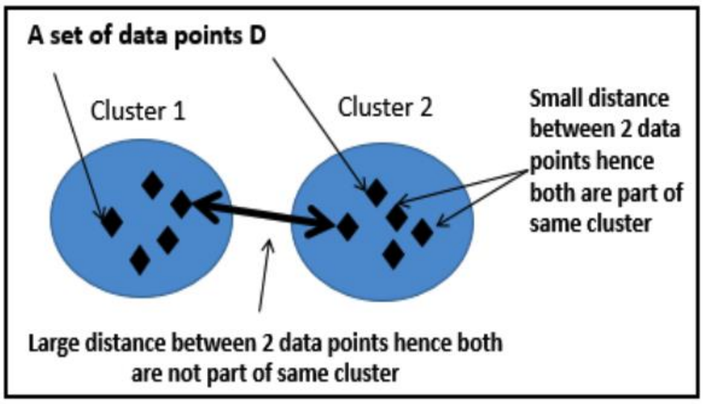
where the function that returns the distance between the centroids of the two clusters $\mathrm{S_i}$ and $\mathrm{S_j}$ is denoted by $\mathrm{d}\left(\mathrm{C_{S_i}}, \mathrm{C_{S_j}}\right)$. $\mathrm{S_i}$ has $\mathrm{N_{S_i}}$ as its cardinality and $\mathrm{C_{S_i}}$ as its centroid, while $\mathrm{S_j}$ has $\mathrm{N_{S_j}}$ as its cardinality and $\mathrm{C_{S_j}}$ as its centroid.

The unification procedure identifies the specific difference between other linkage procedures and Ward's method, which does not combine groups with the shortest distance, but when a heterogeneity given measure is not increased too much, groups are joined. Therefore, Ward's method aims to create groups that are reasonably homogeneous and unifies groups when variations do not increase too significantly [60]. What also sets Ward's method apart is that no other agglomerative clustering approach employs a traditional sum-of-squares criterion, generating groups that keep intra-group scattering at every binary fusion down to a minimum [59].

However, Ward's method displays sensitivity to cluster shape and its size, so it may be ineffective in cases with complex cluster shapes departing from "the hyperspherical shape" [61].

2.3 Distance Metrics

As explained by Ortega et al. [62], a function describing a distance among the elements or objects in a set is known as a metric or distance function. Figure 2.5 illustrates a basic clustering process based on distance measures. According to research evidence, distance metrics do not constitute an input-output correlation among labels and data points that could possess continuity; rather, distance metrics serve as functions for point pairs which take part in the capture of relationships among input data points, which is informative about the degree of differences between point pairs [63].



A set of data points D
Cluster 1
Cluster 2
Small distance between 2 data points hence both are part of same cluster
Large distance between 2 data points hence both are not part of same cluster

Figure 2. 5. Illustration of the basic clustering process employing distance metrics [64].

(Dis)similarity of clusters and patterns are measured by distance metrics. The distance metrics must be chosen prior to clustering. Clusters embedded within the data set are distinguished by their characteristics, which should correspond to the separation of target objects or closeness of target objects in terms of similarity that is reflected in the measure. There is no universal measure that can be used for all types of clustering problems, because characteristics depend on the context of the problem or on the data, so no individual measure can be used that is optimal for all forms of problems in clustering analysis. It is very difficult to choose the correct distance measure for data mining applications, so selecting the most effective measure is determined by clearly understanding how different measures are able to represent data similarity for the different types of data [55], [65]–[67].

Several requirements must be fulfilled by a distance in order to be considered a metric [44], [68]. Thus, for two objects, x and y, with d (x, y) being the distance between them, these requirements are: two points need to have a distance that is non negative or d(x, y) ≥0; when there are two identical objects, the distance between them must be 0, d(x, y) = 0 if x = y; the distance between y and x is the same as the distance between x and y, so the distance is symmetric d (x, y) = d(y, x); a distance metric is an triangle inequality distance if it satisfies d (x,y) ≤ max {d(x,z) , d(y,z) } $\forall\, x, y, z \in M$ [11], [44], [68].

Euclidean distance

Geometrical problems are often solved by Euclidean distance. It is also called L2 norm [68]. The formula for the Euclidean distance between the data pattern X_i and X_j [55] is:

$$D_e(X_i\ , X_j) = \sqrt{\sum_{l=g}^{d} |x_{ig} - x_{jg}|^2} \tag{2.9}$$

where, x_{ig} and x_{ig} represent the g^{th} dimension of x_i and $x_j$respectively.

This distance often has a negative impact on the performance of clustering in high dimensional datasets having different scales [69]. Furthermore, an input attribute will be capable of overcoming the rest of the attributes if its range is relatively extensive [68].

Mahalanobis distance

It is known as quadratic distance [70], involves detection and analysis of various patterns on the basis of establishing correlations between variables. The formula for this distance measure is as follows [8], [71]:

$$D_{Ma}(x_i, x_j) = (x_i - x_j)V^{-1}(x_i - x_j)^T \tag{2.10}$$

where $D_{Ma}(x_i\ , x_j)$ represents Mahalanobis distance between points x_i and x_j and V is covariance matrix between the components of points x_i and x_j.

Mahalanobis distance is a scale-invariant and takes account of correlations of the dataset. Furthermore, it is sensitive to sampling fluctuations and violates the triangle inequality [72].

Standardised Euclidean Distance

The Euclidean distance among the data points divided by the standard deviation of the points gives the standardised Euclidean distance. The following expression represents the squared standardised Euclidean distance between $x_i$and x_j [55]:

$$D_{Se}(x_i, x_j) = (x_i - x_j)D^{-1}(x_i - x_j)^T \quad (2.11)$$

where the variance of points x_i and x_j over N data points is represented by D as the diagonal matrix.

A diagonal Mahalanobis distance measure is produced when the squared standardized Euclidean distance is multiplied by the geometric mean of the variances [73].

Manhattan Distance

The sum of the absolute discrepancies of the coordinates of two data points represents the Manhattan distance. The mathematical expression of this distance is [55]:

$$D_{Mn}(x_i, x_j) = \sum_{L=1}^{d} |x_{iL} - x_{jL}| \quad (2.12)$$

where $D_{Mn}(x_i, x_j)$ represents Manhattan distance between point x_i and x_j . x_{iL} and x_{jL}denotes the L^{th}coordinate of points x_i and x_j respectively.

The Manhattan distance gives rise to a cluster of rectangular form and its generalisation is straightforward in high dimensions [55]. On the upside, the Manhattan distance has a shorter computation time than the Euclidean distance [74], but on the downside, the Manhattan distance is reliant on the coordinate system rotation [55].

Cosine Spearman Distance

Cosine Spearman Distance is often used in text documents [66], for clustering analysis [75], and in applications for information retrieval [76]. This measures the cosine of the angle between two vectors, defined as [55].

$$D_{cos}(x_a, x_b) = 1 - \left(\frac{x_a^T x_b}{||x_a|| ||x_b||}\right) \tag{2.13}$$

where $D_{cos}(x_a, x_b)$ represents Cosine Spearman distance between point x_a and x_b and $||x_a||$ and $||x_b||$ are magnitudes of vectors x_i and x_j respectively.

The Cosine Spearman Distance is applied for measurements of cohesion within clusters [6]. It is bounded between 0 and 1 and non-negative. The result of the Cosine function is less than 1 when the angle is of any other value and equal to 1 when the angle is 0. There is increased similarity for the representations of the vectors when they get closer, so that the cosine angle approaches 1, and the angle between the vectors shortens [77]. However, this measure is not invariant to shifts, cannot provide information regarding the magnitude of differences, it violates triangle inequality [6], [55].

Correlation Distance

The Pearson correlation coefficient is the basis for the correlation distance measure [45], which measures the degree of linear dependency between two data points. The formula for this distance is [55]:

$$D_{Corr}(x_i, x_j) = 1 - S_{CR}(x_i, x_j) \tag{2.14}$$

$$S_{CR}(x_i, x_j) = \frac{\sum_{k=1}^{d}(m_{ik})(m_{jk})}{\sqrt{\sum_{k=1}^{d}(m_{ik})^2 \sum_{k=1}^{d}(m_{jk})^2}} \tag{2.15}$$

where, $m_{ik} = x_{ik} - \overline{x_i}$, $m_{jk} = x_{jk} - \overline{x_j}$, $\overline{x_i} = \frac{1}{d}\sum_{k=1}^{d} x_{ik}$ and $\overline{x_j} = \frac{1}{d}\sum_{k=1}^{d} x_{jk}$.
$S_{CR}(x_i, x_j)$ is the correlation between points x_i and x_j and $\overline{x_i}$ and $\overline{x_j}$ are mean of x_i and x_j respectively.

This distance cannot detect magnitude of differences between two data points and has a tendency to infer differences shape differences [68].

Spearman Distance

The Spearman distance measure is underpinned by the Spearman correlation coefficient [78] and its mathematical expression takes the following form [55]:

$$D_{Spear}(x_i, x_j) = 1 - S_C(x_i, x_j) \tag{2.16}$$

$$S_C(x_i, x_j) = \frac{\sum_{k=1}^{d}(m_{ik}^{r})(m_{jk}^{r})}{\sqrt{\sum_{k=1}^{d}(m_{ik}^{r})^2 \sum_{k=1}^{d}(m_{jk}^{r})^2}} \tag{2.17}$$

where, $m_{ik}^{r} = r(x_{ik}) - \bar{r}$, $m_{jk}^{r} = r(x_{jk}) - \bar{r}$. $S_C(x_i, x_j)$ is the correlation between two ranked vectors x_i and x_j, $r(x_{ik})$ is rank of x_{ik} and $\bar{r}$ is mean of ranks.

This measurement adopts nonparametric similarity, and is stronger against outliers than the Pearson correlation, but when data are converted, there is a loss of information, which is a disadvantage [55].

Chebyshev Distance

This measurement calculates the maximum of the absolute differences between the features of a pair of data points, and is also called chessboard distance, maximum metric, Tchebyschev distance [55]. It is mathematically defined by Kumar, et al. [55].

$$D_{Ch}(x_i, x_j) = \max_1 \leq l \leq d(|x_{il} - x_{jl}|) \tag{2.18}$$

where $d(|x_{il} - x_{jl}|)$ is the distance between the Lth dimensions of points x_i and x_j. And $\max_1$ is the maximum value of distance of any dimension of points x_i and x_j.

This measure shortens the time to determine distances between data sets, which is an advantage [79] and, for quantitative variables and ordinal variables; this metric could be used [80].

Canberra Distance

This metric was developed by Lance and Williams [81] and measures the sum of absolute fractional differences between the features of a pair of data points. This is mathematically defined by Kumar, et al. [55] as

$$D_{Can}(x_i, x_z) = \sum_{i=1}^{d} \frac{|x_{il} - x_{zl}|}{|x_{il}||x_{zl}|} \tag{2.19}$$

where x_{il} and x_{zl} is the l^{th} coordinate of points x_i and x_z.

This measurement is normally adopted when data are scattered around the origin [77]. When both coordinates are close to zero, this metric is sensitive to small changes [55].

Bray-Curtis Distance

Bray and Curtis [82] develop this measurement, which is also called Sorensen distance, and calculates the absolute differences divided by the summation. This is mathematically defined by Kumar, et al. [55] as

$$D_{Bc}(x_i, x_j) = \frac{\sum_{i=1}^{d}|x_{il} - x_{zl}|}{\sum_{i=1}^{d}(x_{il} + x_{zl})} \tag{2.20}$$

Where x_{il} and x_{zl} is the l^{th} coordinate of points x_i and x_z.

This measurement does not satisfy triangle inequality, if two data points are close to zero values, then the results are undefined, which is a disadvantage [55].

2.4 Semi-Supervised Learning

In some instances, the data analyst has access to a priori (domain) knowledge regarding the fundamental composition of the data. The supervised learning technique may not be suitable, given the unlabelled data or only partly labelled. In this case, data may be learning by deploying the unsupervised learning approach with neglecting the prior knowledge. This may cause the outputting of irrelevant structures from the data. Therefore, an alternative approach may be found in the semi-supervised learning, as it can produce a better outcome quality by incorporating a partially prior knowledge into the learning process [2].

Semi-supervised learning is a fairly recent approach to data mining, pattern recognition and machine learning that borders the supervised and unsupervised learning approaches. The semi-supervised learning can fall into two broad areas: 1) semi-supervised classification which is the integration of unlabelled data to supervised classifier learning; and 2) semi-supervised clustering which is the integration of some supervision into clustering [83]. According to Zhu [84], the semi-supervised clustering technique is the viable option for data miners who are attempting to cluster and have unlabelled data containing a range of pair-wise constraints. However, for those whose goal is to classify and have a large number of labelled, the semi-supervised classification is the most suitable approach.

There is a growing organisational and academic interest in semi-supervised learning in terms of putting theory to practice, given it leads to improved accuracy with relatively less human effort [85]. Numerous studies have reported considerable interest in semi-supervised learning when merging labelled data and unlabelled data [86], [87], and can be applied to clustering [11], [88]–[91] and classification [92]–[95].

In most domains, knowledge of the appropriate categories is partial. According to Basu [86], the semi-supervised classification, unlike the semi-supervised clustering (in the model-selection framework), is capable of grouping data using the categories in the initial labelled data while extending and modifying the existing set of categories accordingly to reflect other regularities in the data. For Rizoiu [96], the semi-supervised clustering technique is more viable for considering the additional information entrenched in complex data compared to the semi-supervised classification technique, which, for an array of reasons, is incapable of handling complex data. First, the quantity of categories must be predetermined and known beforehand and secondly, labelled examples must be available for each category. These conditions could prove unfeasible when dealing with complex data, for which the classification typology might not be known beforehand (or not even the class count). Additionally, supervised information might be present only under the guise of some pair-wise connections. For instance, the knowledge of the fact that two individuals ought to be classified together, but no further information regarding the category under which they ought to be classified exists [96]. Therefore, this study focuses primarily on semi-supervised clustering throughout the thesis.

2.4.1 Semi-Supervised Clustering

Based solely on similarity information, clustering is fundamentally ill-posed problem in which the aim is to partition the data into an unknown number of clusters to maximize within-cluster similarity, while minimizing between-cluster similarity [97]. It is difficult for a clustering algorithm to retrieve the data partitions to meet the various criteria of a concrete task. Consequently, for Ma and Dhavala [5], any side or external information from other sources could prove significantly useful in guiding clustering solutions. Clustering using external information or semi-supervised clustering algorithms have gained a footing within the clustering community, as they could potentially improve the efficiency and relevance of traditional unsupervised clustering by incorporating prior knowledge into the clustering process to obtain relevant information from the data [2], [98].

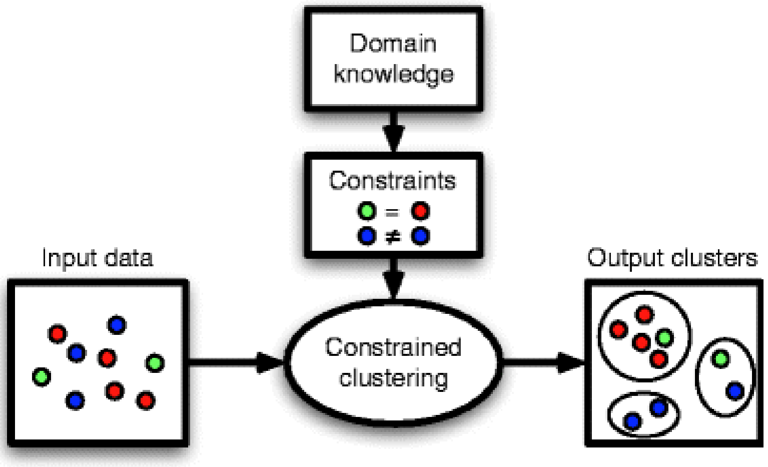


Figure 2. 6. Steps of constrained clustering algorithm with pair-wise constraints [99].

Figure 2.6 illustrates the instance of the constrained clustering algorithm with pair-wise constraints that has been proposed by Wagstaff [99]. This type of algorithm is identical to an unsupervised clustering algorithm in terms of the inputs, but it necessitates a set of constraints based on domain knowledge. It takes into account both input constraint and data patterns to determine the clusters. Some data patterns are forced to neither merge nor assign together, which is not supported by the traditional unsupervised clustering algorithm. As is illustrated in Figure 2.6, the first constraint (ML) sits at a position where it forces the red point and green to merge. The second constraint is positioned in a way that it will hamper two blue points from linking together with the same cluster. As a result, when these constraints are applied, the red cluster has a green point which results from the application of the first constraint, and the blue point is assigned or merges in the green cluster, a consequence of the second constraint. Therefore, the difference between clustering with and without constraints can be seen clearly.

Several approaches have been developed for constrained clustering. The prominent approaches can be divided into three classes, namely search-based (constraint-based), distance-based (similarity-based) and hybrid (search and distance-based) approaches [100]. When using a constraint-based method, traditional clustering algorithms are adapted to include previous knowledge into the clustering task. That is, the space in which to search for the solution is modified in accord with the constraints. The reason for using this method is to supervise and guide the algorithm towards partitioning the dataset so that the constraints are not violated [89], [101], [102]. Frequent techniques used in search-based methods are to modify the objective function through the addition of penalty terms for constraints which are not satisfied and use previous knowledge to prepare the clusters [2]. In distance-based methods, clustering methods are mainly used with change the distance between patterns according to previous knowledge in the form of constraints. Adjustments are made to the distance measure so that data objects are positioned in the same cluster and are nearer to each other, whilst data objects should be positioned in different clusters further away from one another [2], [103]–[105]. Hybrid methods use a combination of distance and search-based methods. These methods incorporate the advantages of both and usually show improved performance over individual methods [2].

Constrained clustering algorithms produce better clustering results, prompting their use in a range of domains, including image analysis [106]–[108], gene expression [109],

[110], social networks [111].

Several clustering algorithms are proposed in the literature for adding constraints. Constrained versions of PC and HC. The great interest for research in semi-supervised clustering, however, emerged in the early 2000's, when Wagstaff and Cardie [10] . They introduce constrained partitional clustering is called COP-COBWEB, an algorithm that employs pair-wise constraints ML and CL [10] . Through these constraints, the user can indicate whether two instances must or must not belong to the same cluster. The authors have shown that these constraints could significantly improve accuracy performance when compared to unsupervised methods and then developed COP-K-means which accommodates the constraints by restricting item assignments to exclude any constraint violations [89]. These methods also provide robustness to noise and outliers. Later, algorithms such as PCK-means and MPCK-means [98] permitted the violation of constraints when necessary by introducing a violation penalty. It is useful when the constraints may contain noise or internal inconsistencies, which are especially relevant in real-world domains. On other hand, constrained versions of the HC methods were developed, such as with pair-wise constraints [58], [104], [112] and with triple-wise relative constraints [11].

According to Liu [113], whether the distance metric is modified, constrained clustering algorithms can fall into two categories: distance-flexible algorithms and distance-fixed algorithms.

- Distance-fixed constrained clustering algorithm: In this category, constraints are viewed only as the relations between data points to guide the clustering process or to prime cluster centres accordingly. The clustering algorithm will not change the distance metric.
- Distance-flexible constrained clustering algorithm: In this category, constraints can be considered as distance pointers that data points belonging to a ML constraint should have a small distance, and data points belonging to a cannot-link must have a large distance. The clustering algorithm will gain knowledge of an adaptive distance metric given the current constraints.

As the distance factor is an important fundamental of a clustering algorithm, the novel constrained algorithms proposed throughout this thesis will follow the approach of distance-flexible constrained clustering algorithms.

2.4.1.1 Constraints with Semi-Supervised Clustering

The prior knowledge can either be given as class labels or constraints, whereby a few class labels of the data object are identified, but the amount of available labelled data may not be adequate to perform classifications [2], [98]. Constraint of the clusters or data objects is a type of useful knowledge that is of practical importance than sometimes the labelled data itself. In practice obtaining a data object's correct label may be a computationally expensive task or may necessitate much effort. However, the deliberation of whether a data object pair belongs to different clusters or the same cluster is relatively easily done by an expert opinion based on the requirements of the problem owner, or via gathering system user feedback [2].

Incorporating external information regarding the similarity relationships among instances can enhance clustering. Such side information is often characterised by two types of constraints: pair-wise constraints and triple-wise constraints, pertaining to the similarities regarding instance pairs and triplets, correspondingly [114]. A pair-wise constraint stipulates absolute similarity correlation between two instances. That is, considering a pair of instances, x_a and x_b, ML constraint is introduced if they are similar to each other, and a CL constraint is presented otherwise. Relative constraints, in comparison, show comparative similarity correlations among instance triplets x_a, x_b, and x_c. That is, each constraint stipulates whether instance x_a is more similar to x_b than to x_c [114]. It is also known as Must-link-before, which stipulates the order followed in the merging of the objects and can be naturally incorporated into the HC process [11].

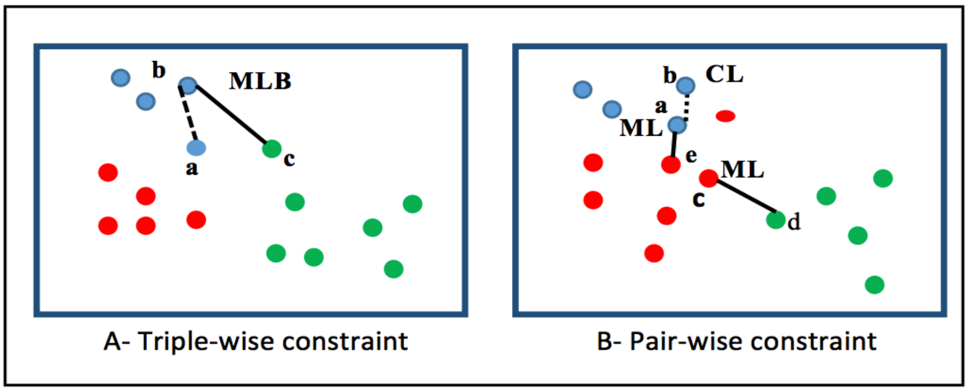


Figure 2. 7. The two types of constraints based on knowledge, namely, pair-wise constraints and triple-wise or MLB constraints, respectively shown on the left- and right-hand side.

Figure 2.7 shows two forms of prior knowledge that can be integrated into the clustering process. Solid lines indicate ML constraint and dashed lines denote CL constraint. In Figure 2. 7, the left block/diagram, A, illustrates triple-wise constraint which stipulates that point b and point c must be assigned or merged before the assigning and merging of points a and b. This is in spite of the reality that points a and b are closer and located in the same cluster while points b and c are further and will be found in different clusters, following the rule, d(b,a) < d(b,c). Hence, when clustering based on constraints is applied, the result is the forcing of the condition d(b,c) < d(b, a), which would assign or merge point b and point c before assigning or merging point a with point b.While, right block/diagram, B, represents a pair-wise knowledge-based constraint. An analysis of Figure 2.7 B shows the inputting of two ML constraints. First constraint ML(c,d) forces point c and point d, which are located in different clusters, to assign and merge together. The second constraint ML(a,e) forces point a and point b, which are located in different clusters to assign and merge together. Cannot link constraint (CL(a,b)) prohibits points a and b from assigning or merging, even though they are located in the same cluster. Hence, after the application of ML constraints-based clustering, point c with the red colour will be assigned and merged in the green cluster. point a, which has a blue colour, will be assigned and merged in the red cluster.

Pair-wise constraints are easier to manage than triple-wise constraints when there are fewer instances to be examined in each query. However, providing pair-wise constraints demands absolute dissimilar or similar judgements that has long been believed to be harder to obtain than the comparative judgment required by triple-wise constraints [115]. More accurate information is produced by triple-wise constraints, but these also reveal limitations, such as a greater possibility of mistakes in labelling triple-wise constraints when there are more instances to consider. This leads to higher chance of making an incorrect judgment on individual instances [114].

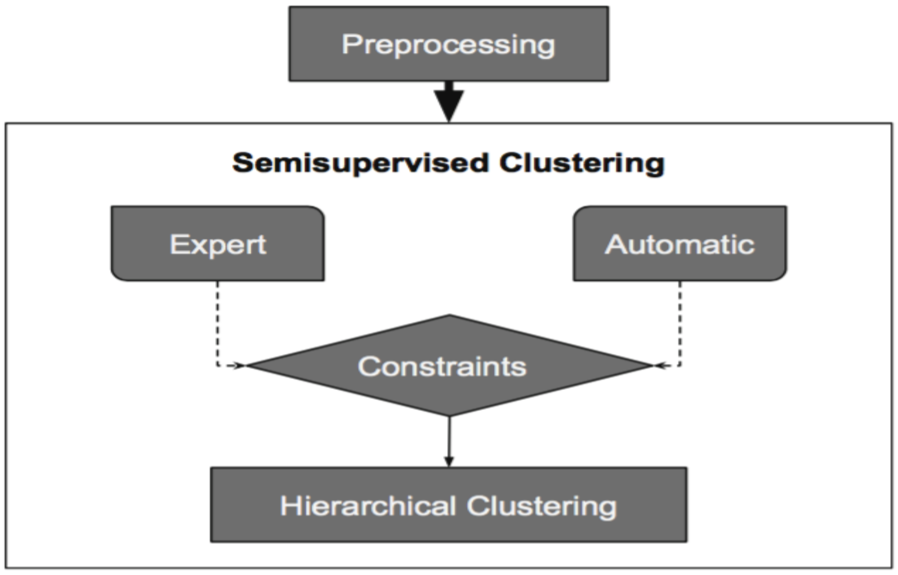


Figure 2. 8. The expert-based and automatic approaches for production of constraints in the context of the semi-supervised clustering algorithm [116].

As illustrated in the Figure 2.8 above, an automatic method or reliance on the knowledge of human experts is the approach used to produce knowledge-based constraints associated with semi-supervised clustering [116]. In cases where supervision is advisable, human expertise is desirable, but acquisition of recommendations from users is frequently effort-intensive and costly. Therefore, knowledge produced in an automatic manner is a feasible substitute for human expertise in certain methods. Moreover, the performance of partitioning based on the production of automatic constraints is higher compared to the performance of partitioning based on constraints established by users in keeping with class labels [116]. An approach underpinned by human expertise was adopted by Cohn et al. [103] and Huang and Mitchell [117], while an automatic approach was employed in the studies by Diaz-Valenzuela et al. [116] and Zheng and Li [11].

The constraints can be classified as “Cluster-level constraints” and “Instance-level constraints”. “Cluster-level Constraints” is specifying requirements on clusters which is considering as whole sub clusters. “Instance-level constraints” is specifying requirements on pairs of objects that is considering as single instances [118],[119]. Both studies report that cluster-level constraints generated greater information than instance-level constraints. On the other hand, instance-level constraints are easier for the user compared to cluster-level constraints. The instance-level constraints can be interpreted by users easier than cluster-level constraints. Furthermore, instance-level constraints not only do not require as high a degree of expertise as cluster-level constraints, but they are also computationally more inexpensive. The cost of algorithms associated with cluster-level constraints is further heightened by the fact that these algorithms are dependent on feedback and necessitate a preliminary clustering process [119]. Given these considerations, instance-level constraints are adopted in the present work, with volumes of constrained data ranging between 10% and 60% of the overall patterns of data to assess the effect of varying degrees of knowledge-based constraints.

2.4.1.2 Semi-supervised Hierarchical Clustering

Different supervised HC methods consider different forms of knowledge-based constraints. To give an example, the clustering of observations connected by a “ML” constraint has to be done at the minimal hierarchy level [58], whereas observations separated by a “CL” constraint have to be in different clustering hierarchies. Therefore, the approach suggested by Miyamoto and Terami [58] yields more than one clustering hierarchy instead of just one. Furthermore, in the case of observations covered by a “CL” constraint, a hierarchy is generated for every individual observation.

A set of so-called “Must-Link before” constraints have been proposed by Bade and Nurnberger [120] for HC, involving the grouping of a series of observations prior to their grouping with other data points. Meanwhile, the cluster-wise tolerance-based pair-wise constraints put forth by Hamasuna, Endo and Miyamoto[121] establish the “ML” and “CL” constraints among cluster pairs according to the weighted count of how many of these constraints occur among observations within clusters. Zheng and Li [11] claim that HC could be conducted based on triple-wise relative constraints (xi, xj, xk), specifying that xi and xj should be more similar or closer than xi and xk, which is expressed as d(xi, xj) < d(xi, xk). Addressing an issue of constrained optimisation is necessary for the algorithm to attain the ultra-metric transformation of the dissimilarity matrix representation. A particular form of tree metric, the ultra-metric is characterised by the fact that each input dataset component is a leaf in the underpinning tree and each leaf is equally distant from the root [11].

The use of the algorithm proposed by Zheng and Li [11] in the proposed work is justified due to several reasons. Firstly, the algorithm is suitable for applications requiring the clustering process to be hierarchically structured (for example, issues related to object collection organisation) as it is underpinned by triple-wise relative constraints. Secondly, the algorithm considers updated dissimilarities with the "reducibility condition", which are discussed in the next section. Thirdly, unlike other algorithms, this algorithm can update distances based on the constraints supplied in accordance with the configuration of the data; as previously highlighted, despite the significance of distances for the clustering algorithm, it is not possible to apply the one type of distance technique to all datasets for obtaining best results.

The output of the Constrained Hierarchical Clustering (CHC) algorithm is a dendrogram, which can be described as a rooted tree whereby each data point is represented by a leaf, and each internal node is represented by a cluster comprising of its falling leaves. The patterns inside the clusters grow in similarity to each other as the internal nodes get deeper into the tree, thus forming more refined clusters [1], [5], [11]. For most HC algorithms, two clusters that represent reciprocal nearest neighbours are merged without changing the initial merge orders, a property is known as irreducibility. If merge orders were changed, clustering would necessarily stop before reaching the root node and violate a constraint. The ultra-metric inequality (Eq. 2.21) is fulfilled to reflect the updated differences taking into account the 'reducibility condition" [11], [122].

$$d(x_i, x_J) \leq \max\left(d(x_i, x_k), (x_J, x_k)\right) \forall x_i, x_J, x_k \in X \tag{2.21}$$

The two constrained optimisation approaches are employed for distance updating to obtain the ultra-metric distance matrix [11] are represented in Algorithm 2.2 and Algorithm 2.3

Algorithm 2.2 represents first approach namely, the optimisation-based approach or iterative projection optimisation (IPoptim), whereby the least-square loss function taking the form of a distance matrix is optimised in keeping with the ultra-metric and triple-wise relative constraints. Algorithm 2.3 represents second approach namely, the transitive dissimilarity-based approach (UltraTran), which is an altered version of the Floyd-Warshall algorithm and seeks to minimise a transitive dissimilarity matrix to an ultra-matrix according to the triple-wise relative constraints underpinning the process of transitive dissimilarity formulation [11].

Algorithm 2.2. Iterative projection algorithm (IPoptim) [11].

```
Input:
$\vec{d}$: Vector representing pair-wise dissimilarities,
C:  Triple-wise relative constraints,
 E:  identity matrix.
Init $\vec{a} = \vec{d}$ and $\vec{u} = \vec{O}$
    1.  while not converge do
    2.      p = t mod r
    3.      $\vec{s} = \vec{a}\,(t-1) + E\,\vec{c}_p \vec{u}\,(t-1)_{p/2}$
    4.      for q = 1 to r do
    5.          if q = p then
    6.              $\vec{u}\,(t)q = \max(0, 2 * c^{T\rightarrow}{}_q\,\vec{s} / c_q q E \overrightarrow{c_q})$
    7.          else
    8.              $\vec{u}\,(t)q = \vec{u}\,(t-1)\,q$
    9.          end if
    10.     end for
    11.     $\vec{a}\,(t) = \vec{s} - E\,\overrightarrow{c_q}\,\vec{u}\,(t)_q/2$
    12.     t = t + 1
    13. end while
    14. return $\vec{\vec{d}} = \vec{a}$
Output: $\vec{\vec{d}}$
```

From the algorithm 2.2, the pair-wise differences of D are denoted by the vector and $\vec{\vec{d}}$ and $\vec{\mathrm{d}}$, a m×m identity matrix is denoted by E, and an r×m matrix encompassing every r triple-wise relative constraint is denoted by $C = [C_1^T\ , C_2^T, \ldots \ldots C_r^T]$. $\vec{a}$ (t) and $\vec{u}$ (t) are sequence of estimated solutions and sequence of Kuhn-Tucker vectors respectively, which are represent $\vec{a}$ and $\vec{u}$ in iteration t.

The IPoptim algorithm can be described as follows. For *n* observations and r relative constraints, the $n \times n$ symmetric dissimilarity matrix D takes the form of a vector $\vec{d}$ of dimension $m \times 1$ with $m = n \times (n\text{-}1)/2$. The entries of the superior/inferior triangle elements of the dissimilarity matrix *(D)* are denoted by the components of the vector $\vec{d}$. Every relative constraint $(x_i, x_j, x_k) \in$ C takes the form of a vector $\vec{c}$ of dimension $m \times 1$, with the indices equivalent to D_{ij} and D_{ik} being respectively established at 1 and -1. The dissimilarity matrix $d^T c \geq 0$ for any constraint C that is inconsistent with it. The dissimilarity constraints associated with IPoptim have the following vector representation:

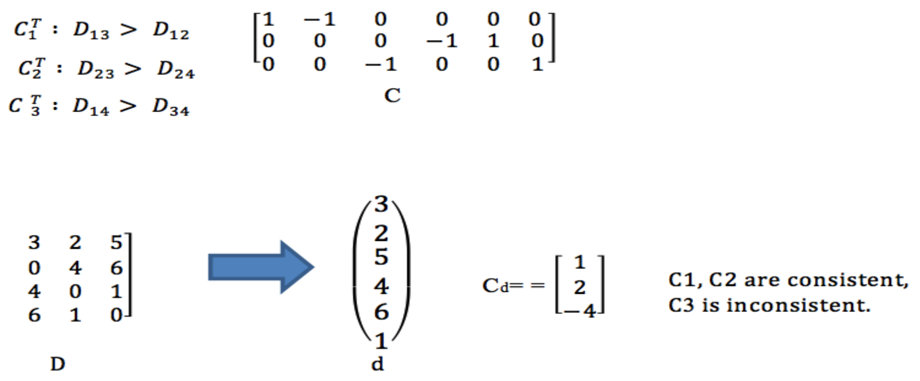
$$\underset{\vec{d}}{arg\min} \left(\vec{d} - \vec{\hat{d}}\right)^T E \left(\vec{d} - \vec{\hat{d}}\right)$$

Subject to,

$$\widehat{D}_{ij} \leq \max\{\widehat{D}_{ik}, \widehat{D}_{jk}\}, \quad \forall \left(x_i, x_j, x_k\right) \in \text{ X},$$

$$C\vec{d} \leq \vec{0}$$

This algorithm method the optimization problem by conducting iterative projection that supports optimal solution to reduce the least square loss function under disparity or dissimilarity constraints."This algorithm runs by repeatedly following the iterative “augmenting” steps. This means that at each iteration, the parameter estimates are first projected onto closed convex sets defined by the constraints $C\vec{d} \leq \vec{0}$ and then updated by subtracting a vector of the changes made in the previous projection" [11]. as shown in the following example in Figure 2.9.



$C_1^T : D_{13} > D_{12}$
$C_2^T : D_{23} > D_{24}$
$C_3^T : D_{14} > D_{34}$
$\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 \end{bmatrix}$
C
$\begin{bmatrix} 0 & 3 & 2 & 5 \\ 3 & 0 & 4 & 6 \\ 2 & 4 & 0 & 1 \\ 5 & 6 & 1 & 0 \end{bmatrix}$
D
$\begin{pmatrix} 3 \\ 2 \\ 5 \\ 4 \\ 6 \\ 1 \end{pmatrix}$
d
Cd= $= \begin{bmatrix} 1 \\ 2 \\ -4 \end{bmatrix}$
C1, C2 are consistent,
C3 is inconsistent.

Figure 2. 9. The optimization process based on constraints [11].

Algorithm 2.3. Modified version of the Floyd-Warshall algorithm (UltraTran) [11].

```
Input:
G: Pair-wise distance matrix of data set.
C: Triple-wise relative constraints.
Initiation: M = G.
    1.  for k ← 0 to N do
    2.      for i ← 0 to N do
    3.          for j ← 0 to N do
    4.              for all c = ($x_i$, $x_j$, $x_l$) do
    5.                  minCon = min(minCon, d($x_i$, $x_l$))
    6.              end for
    7.              $m_{ij}$ = min{$m_{ij}$ , max($m_{ik}$,$m_{kj}$),minCon}
    8.          end for
    9.      end for
10: end for
11: return M
Output: M: Minimum Transitive dissimilarity matrix.
```

where G denotes the pairwise distance matrix; C the triple wise constraint set; M the distance matrix which is being updated in the current iteration; m_{ij} the current matrix value for column j and row i; minCon the minimum merge order of points i and j.

The UltraTran algorithm calculates the minimum transitive dissimilarity which meets the original dissimilarity matrix to an ultra-matrix and at the meantime to integrate the provided relative constraints. It is updated value for m_{ij} to identify the pairwise dissimilarities associated to xi and xj and controlled by constraints.

The path Pij's transition dissimilarity can be presented as

$$T(P_{ij}) = \max(d_{i,k_1}, d_{k_1,k_2}, d_{k_2,k_3}, \dots, d_{k_{n-1},k_n}, d_{k_n,j})$$

The minimal transitive dissimilarity between all the paths that exist for any particular pair of vertices xi and xj can be defined as:

$$m_{ij} = \min_{p_{ij}} \left(T(P_{ij}) \right)$$

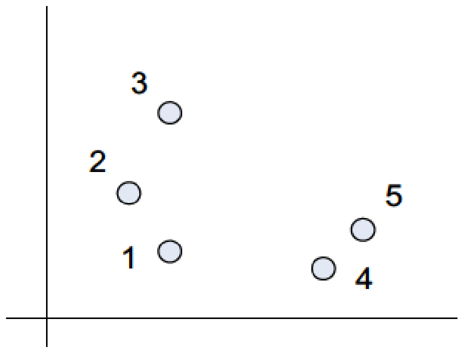


Figure 2. 10. Cluster separation due to the transitive dissimilarity in 2D space for a simple dataset of 5 points.

The procedure associated with $T(P_{ij})$ within Ultra-Tran algorithm can be clarified through the following example. Based on the considerations that $T(P_{ij})$ denotes the transitive dissimilarity for the path, path P_{13} from Figure 2.10 being equivalent to P_{1453}, and d_{14}, d_{45} and d_{53} being respectively 80, 20 and 100, then $T(P_{1453}) = T(P_{13}) = \max(80,20,100) = 100$. For a different path P_{123} from 1 to 3, with d_{12} and d_{23} equivalent to 40 and 60, then $T(P_{123}) = \max(40, 60) = 60$. When every path from i to j is taken into account, the minimal value of $T(P_{ij})$ is given by m_{ij}. Therefore, P_{1453} and P_{123} are the two paths from 1 to 3 in the previous example. Calculation of m_{13} can then be undertaken as $m_{13} = \min(T(P_{1453}), T(P_{123})) = \min(100,60) = 60$.

2.5 Selecting and Weighting of Features in the Context of the Clustering Process

In terms of current learning domains, features that are potentially useful are determined by humans, but this could include some features that are irrelevant and unimportant. Labelling for human studies is considered to be subjective and expensive, and manually labelling by humans to categorise each instance is difficult when dealing with large amounts of data, so that there are several current databases that are unlabelled [37], [123].

Furthermore, some predictive accuracy functions are maximised by feature selection algorithms adopted for classification techniques, so that researchers want to maintain features that lead to classes or are related to classes, as they are given class labels. However, class labels are not given in clustering [123] or few labelled data for semi-supervised clustering.

Grouping similar objects together is a clustering goal, but the probability density model and distance metric define types of similarity, these depend on data features [37]. Often, equal importance is given for treating features, when dissimilarities are represented by distance metrics, but features should not always be weighted equally [124]. According to Chan, et al. [125], when several attributes are unimportant to some clusters, the clustering results may be less accuracy. An algorithm could be misled if there is insufficient information demonstrated by features but could still be presented as equal relevance as features with strong levels of information features. Classifiers would be likely to recognise a group of features that are redundant but carry the same information and are identified as being relevant [126]. Although high dimensional data often causes some clustering algorithms to fail, this problem can be overcome by improving understanding by computing and exploring the most relevant and important features [123].

Features containing the least information may misinform a clustering algorithm, as they may be assigned the same level of relevance as information-containing features. Redundant features, which are a pair of two or more features, each of containing the same information [126]. The remedy for this occurrence may lie in feature selection techniques. Only the relevant features are applied to clustering algorithms, and the construct similarity used to develop the final partitioning [127].

An alternative technique for feature selection is feature-weighting techniques. In these techniques, features are not excluded but suitable weights are assigned to them based on their relevance. In the feature-weighting technique, more significant features are prioritised over others and hence more weight. This ensures that the significant features can play a bigger role in determining the membership of instances to clusters. Feature selection is viewed as a sub-branch of feature weighting instances where features contain only binary weights (0 and 1), or where it is okay for weights to drop down to 0 [128]. However, it does not explain why feature weighting ought to be a pre-processing step and, nor why users should deploy a method limited to weights of either 0 or 1. According to De Amorim [17], feature weighting can be performed alongside clustering itself.

Within clustering are several feature selection techniques. These feature selection techniques can fall into two categories—wrapper or filter—based on whether the evaluation methods are influenced by the learning algorithms [37].

(i) Filter methods

Feature selection algorithms of this nature are not dependent upon the learning algorithm being used. They employ the attributes of the data itself to select which features must be retained without considering the classification or clustering algorithm(s) that will be subsequently adopted. The name of the approach is intuitive, as the idea is to filter the data before classification or clustering is performed [37], [126]. Figure 2.11 illustrates how the filter approach uses the data alone to determine which features ought to be retained, without running the learning algorithm [37].

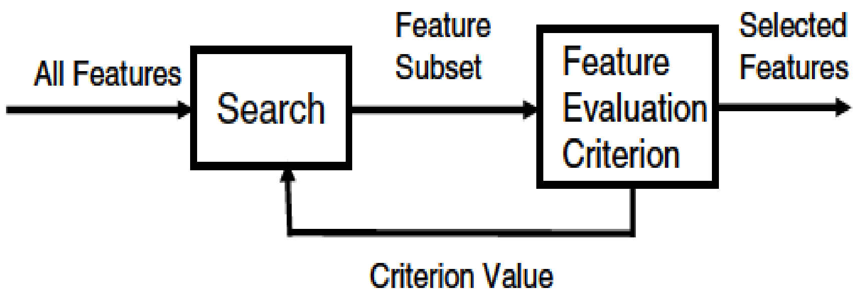


Figure 2. 11. Selection of features for clustering based on a filter technique [37].

(ii) Wrapper methods

In such techniques, feature selection algorithms employ the learning algorithm to decide the quality of the subset of the selected features. If clustering is involved, the wrapper method would integrate the feature selection algorithm inside the selected clustering algorithm [37], [126]. As illustrated in Figure 2.12, the wrapper method wraps the feature search around the learning algorithms that will eventually be applied and uses the learned results to select the features [37].

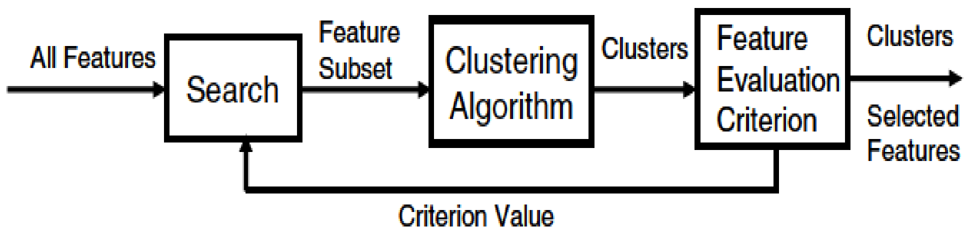


Figure 2. 12.Selection of features for clustering based on a wrapper technique [37].

When compared to wrapper methods, filter methods tend to have lower computational cost as they minimize the feature subset's size. When using the wrapper methods, every candidate within the feature subset's learning algorithm has to be considered Nonetheless, better outcomes are usually obtained using wrapper methods than in the case of using filter methods within clustering algorithms [37]. In some cases, when filter methods are used, they may destroy the structures of clusters as they only choose features for the construction of (dis)similarity patterns for final partitioning. The selection of features may fail to contain the same information or could lose their relationship with the full features. Several domains may need all variables for them to sustain the physical interpretation of these features [123]. Given this premise, this study employs the wrapper technique to be deployed with the constrained clustering algorithm for automatic determination of feature weights. This enables a feature to be important to a varying extent for different clusters.

2.5.1 Feature weighting with Ward 's Hierarchical clustering

Ward's hierarchical clustering approach operates on the assumption that all features are equally relevant, which poses challenges in clustering data with irrelevance and noise. The remedy for this may lie in applying a feature selection algorithm to a dataset prior to deploying Ward's method. However, this algorithm is not integrated into Ward's approach and it does not consider the fact that even among important features there may be different levels of relevance [17].

As a result, the Wardp [17] and WardpB [21] techniques have developed an interesting approach that incorporates major alterations to Ward's hierarchical clustering criterion (see. equations 2.22 and 2.23) to allow a feature to contain varying degrees of importance at different clusters. The idea would be to incorporate feature weighting and using the p-the power of the weighted Minkowski metric [17], [21]. The Wardp and WardpB techniques combine the two clusters that contain the smallest cost as per the equations below.

$$\mathrm{Ward}_{\rho}\left(S_i, S_j\right) = \frac{N_{S_i} N_{S_j}}{N_{S_i} + N_{S_j}} \sum_{v=1}^{v} w_{kv}^{P} \left| c_{S_i v} - c_{S_j v} \right|^{P} \qquad (2.22)$$

$$\mathrm{Ward}_{\rho B}\left(S_i, S_j\right) = \frac{N_{S_i} N_{S_j}}{N_{S_i} + N_{S_j}} \sum_{v=1}^{v} w_{kv}^{B} \left| c_{S_i v} - c_{S_j v} \right|^{p} \qquad (2.23)$$

where centroids c_{S_i} and c_{S_j} denote the S_i and S_j clusters, and w_{kv} represents the weight of feature v. Where user-defined factors are p and B Weights and distances both utilise the same exponent parameter, p within Wardp, while WardpB is extending to permit the deployment of varying exponents parameters for the distance (p) and the feature weights (B). Feature weight updating is performed by every weight w_{kv} based on equation 2.24.

$$w_{kv} = \frac{1}{\sum_{u \epsilon V}[\, D_{kvP} / D_{kuP}\,]^{1/(P-1)}} \qquad (2.24)$$

where D_{kvp} denotes the summation of within-cluster variances of feature *v* weighted by cluster k cardinalities: $D_{kvp=} \sum_{i \epsilon S_K} |\, y_{iv} - c_{kv} \,|^{p}$. where y_i is an object in the dataset Y.

2.6 Cluster Validity and Evaluation

The selection of an approach for result evaluation is the chief decision to be made in the context of experiment design. Cluster validation is challenging. There is no "gold standard" solution available to evaluate how good the outcomes of clustering algorithms are. There are common types of measures for clustering validation, namely, internal and external measures [129]. The former is based on the information inherent to the data being interrogated alone; for example, the ratio of average inter-cluster to intra-cluster resemblance, requires only the data. Whereas the latter is based on known class labels regarding data which corresponds to the clustering solution to some pre-established knowledge; for example, a fundamental class labelling of the data. The experiments in the present study have employed two metrics for cluster validation: F-measure, and Calinski-Harabasz (CH) index . The F-score encompasses external indexes that measure the quality of clustering regarding a certain underlying class labelling of the data. In this technique, near-zero values indicate low agreement, while near-1 values denote otherwise. CH index, on the other hand, represents internal indexes for assessing cluster quality in terms of compactness and well-separateness.

Evaluation of the ssHC algorithm has been undertaken based on the F-measure in a number of works [11], [119], [130], [131]. Nogueira et al. [119] point out that F-measure is an evaluation method suitable for HC. Furthermore, comparative analysis of a broad range of internal measures indicates that the CH measure is among the best internal measures [132], [133].

F-measures

It is also referred to as the F-score [75]. It has been described as the harmonic mean of pairwise precision and recall, the conventional information recovery measures adapted for assessing clustering by observing pairs of points.

The accuracy of HC is evaluated considering the entire hierarchy using the real classes of the data object. Suppose that the hierarchy is cut at a certain level and the group $\mathrm{G_i}$ is generated, where $\mathrm{D_j}$ is a group of data sharing the same label over C. The F-score are calculated according to equation 2.24.

$$\mathrm{F-score}(\mathrm{G_i}, \mathrm{D_j}) = \frac{2*\mathrm{Recall}(\mathrm{G_i}, \mathrm{C_j})*\mathrm{Precision}(\mathrm{G_i}, \mathrm{C_j})}{\mathrm{Recal}(\mathrm{G_i}, \mathrm{C_j})+\mathrm{Precision}(\mathrm{G_i}, \mathrm{C_j})} \quad (2.25)$$

$$\mathrm{Recall}(\mathrm{G_i}, \mathrm{C_j}) = \frac{n_{ij}}{n_i},$$

$$\mathrm{Precision}(\mathrm{G_i}, \mathrm{C_j}) = \frac{n_{ij}}{n_j}$$

where n_{ij} = the number of elements in $\mathrm{C_j}$ that belongs to $\mathrm{G_i}$, $n_i$is size of $\mathrm{C_j}$ and $n_j$is size of $\mathrm{G_i}$. The F-score of group $\mathrm{G_i}$ is determine as the maximum F-score over all C classes:

$$\mathrm{F-score(G_i)} = \max_{j \in C} \mathrm{F-score}\left(\mathrm{G_i}, \mathrm{D_j}\right)$$

To calculate the F-score for the entire hierarchy, we calculate the weighted sum of each group's F-score of the form:

$$\mathrm{F-score(H)} = \sum_{\mathrm{i=1}}^{\mathrm{N}} \frac{|\mathrm{G_i}|}{\mathrm{D}} \mathrm{F-score(G_i)} \tag{2.26}$$

Where N is possible clusters can be created by cutting at various levels for hierarchical clustering D which are calculated as $\frac{(1+|D|)*|D|}{2}$.

Calinski-Harabasz index

The ratio of between-sum-of-squares (BSS) to within-sum-of-squares (WSS) underpins the Calinski-Harabasz (CH) index [134], which has the following formula:

$$CH_k = \frac{\mathrm{BSS}}{k-1} * \frac{n-k}{\mathrm{WSS}} \tag{2.27}$$

where the overall count of points and the count of clusters are respectively denoted by n and k, while the inter-cluster distance underpins the BSS. The formula for the BSS is:

$$BSS = \sum_{i=1}^{K} n_i\, d(z_i, z_j)^2 \tag{2.28}$$

where the middle of cluster c_i and the point count in c_i are respectively denoted by z_i and n_i. The formula for the WSS is:

$$WSS = \sum_{i=1}^{K} \sum_{x \in c_i} (x, z_i)^2 \tag{2.29}$$

where a data point included in cluster c_i is denoted by x. Maximisation of BSS and minimisation of WSS are required to ensure cluster separation and compactness. An appropriate division for the dataset is reflected by the highest CH value.

2.7 Statistical Analysis for Clustering algorithm

The role of statistical significance tests is to determine the statistical significance in the variances in the performance of the machine learning algorithms. It is also the basis for the selection of the algorithm that is significant [23], [119].

Friedman test

The Friedman test can be described as a non-parametric test employed to make a comparison of three classifiers or more. It computes the rank-based tests that calculate data ranks, and then X^2 is applied on the average ranks [135]. These tests work by allocating ranks to algorithms for each set of data. The algorithm that performs best is ranked using the number 1 and the algorithm that performs worst is ranked *n*, with *n* denoting the aggregate number of algorithms. In the event that two or more algorithms give the same value for performance, the two will be assigned an algorithm of all the values. To calculate the average performance value, the performance of every set of data involved in the algorithm is averaged [135]. Where the R_i stands for the average rank of the algorithm, *i* then:

$$\chi_R^2 = \frac{12N}{k(k+1)} \left(\sum_{i=1}^{k} R_i^2 - \frac{k(k+1)^2}{4} \right) \tag{2.30}$$

The equation above represents the $\chi 2$ distribution of average ranks with k-1 degree of freedom. N stands for the aggregate quantity of datasets while the aggregate quantity of algorithms involved is represented by k.

Post-hoc test

When it comes to the Friedman test, one of the main issues is that it only has the capacity to determine whether all algorithms perform in an equal way or not. In the event that the hypothesis "all algorithms perform same on the same dataset and same splits of data" [136] is rejected, it is then not possible to make a comparison of the performance of the separate algorithms. The post-doc test then has to be implemented to compare the performance of individual classifiers. The brief introductions to the post-hoc test methods are provided below.

A) Nemenyi

It implemented in the event that a null hypothesis is rejected. It makes it possible to compare the many variables using a single hypothesis (Friedman test). It has the capacity to detect the variables that have a significant difference from each other [137].]. Equation 2.31 is used for its computation [137].

$$AP_N = m*p \tag{2.31}$$

where m=k (k-1)/2 where k=no. of algorithms, p, p-value linked to the null hypothesis.

B) Holm Test

The basis of Holm is in the sequential rejection of Bonferroni test [138]. This is a test based on working from the p-value that is the most significant to the p-value that is least significant. The p1 value is compared with the $\alpha/(k-1)$. In the event that the $\alpha/(k-1)$ has a bigger value when compared to p1 value, then the current hypothesis will be rejected. This is then followed by moving to the subsequent step where a comparison of the significant p-value with $\alpha/(k-2)$ that follows is made. This procedure will be repeated until the least p-value is reached. In the event that the p-value is bigger than $\alpha/(k-j)$ at the j stage, all the hypothesis will then be accepted following that [138]. Equation 2.32 shows how this is calculated [138].

$$AP_{Holm}(i) = \max(m-j+1)* P(i)\ \ p \tag{2.32}$$

where i<=j<=i and m=(k*(k-1)/2), k=no. of algorithms, p,p-value associated with null hypothesis .

C) Shaffer Test

According to Shaffer, the application of the Holm technique at stage j leads to the rejection of hypothesis $_i$ (Hi) where if $P_j \leq \alpha/t_j$ instead of rejecting Hi if $P_j \leq \alpha/(m - i + 1)$. In this instance, $t_j$is the maximum number of hypotheses which can be true given that any (i,. . 1) hypotheses are false[139], [140]. It is calculated according to equation 2.33.

$$AP_{\text{Shaffer}}(i)) = \max(t_j * P_j): 1 \leq j \leq i \tag{2.33}$$

where t_j= highest number of hypotheses which could be true, considering that preceding hypotheses are false.

Mann-Whitney test

The Mann-Whitney test (also referred to as the Wilcoxon rank-sum test; Mann–Whitney U test; Wilcoxon–Mann–Whitney test) can be defined as a non-parametric test used in the comparison of the variances existing among two groups that are independent. This is a null hypothesis test which makes it equally probable that a value selected randomly from a certain group will be lower than or higher than a value selected randomly from a second group [141].

The operation of the Mann-Whitney test is based on two ordered samples $(x_1, x_2 \ldots \ldots x_n)$ and $(y_1, y_2 \ldots \ldots y_n)$. Pairwise and test comparison is made on all of Sample 1 and Sample 2 elements. The aggregate number of comparisons are $n_1 * n_2$, where n_1 is the size of sample 1 and n_2 is the size of sample 2. It calculates U, which stands for the number of instances when y comes before x, and $\overline{U}$ where the number of times when x comes before y. In the event that $P(U < \overline{U}) = \alpha$, the null hypothesis is accepted in the event that the destructions are the same. In the event that the distributions are not seen as equal, then the null hypothesis is rejected [141].

2.8 Conclusion

This chapter discussed various aspects relating to the background of clustering and semi-supervised clustering approaches as well as algorithms employed in this thesis to reinforce existing research on ssHC algorithm. It outlined approaches to developing ssHC algorithm as well as measures for assessing the performance of the proposed approach. The background information that this chapter presents is going to be crucial for readers to gain an understanding of the approaches and algorithms presented throughout this thesis and suggested in the study's structural chapters (Chapters 3, 4, 5, and 6). However, it needs to be noted that each of them has a supplementary literature review. Also, any readers that still need further clarification can resort to the references and primary sources.

CHAPTER 3

Semi-supervised Hierarchical Agglomerative Clustering with Triple-Wise Relative Constraints

3.1 Introduction

As discussed in the previous chapter, researchers have investigated a wide variety of semi-supervised clustering algorithms in recent years. The advantage of semi-supervised clustering algorithms is that they can apply external knowledge to improve the quality of their clustering results. Unlike, unsupervised clustering algorithms which use only an objective function to identify clusters, semi-supervised algorithms apply additional constraints based on external knowledge.

This chapter proposes a novel method to advance the state of the art in clustering research using knowledge-based constraints. It involves incorporating into the linkage criterion of HAC methods [Note: HAC is a specific HC algorithm which is unsupervised] a triple-wise relative constraints mechanism, creating a novel semi-supervised Hierarchical Agglomerative Clustering (ssHAC) algorithm. The performance of the proposed clustering algorithm is rigorously evaluated using IPoptim and UltraTran optimization approaches which are two techniques to seek an approximate dissimilarity metric (ultra-metric) that satisfies the given triple-wise relative constraints. Further the use of several different linkage methods and distance metrics are also investigated within the evaluation of the proposed ssHAC algorithm with triple-wise constraints.

The purpose of this chapter is to present the outcomes of research that emerged from the research conducted and to provide recommendations about how to select an appropriate combination of a distance metric, linkage method and constraints aimed at optimizing the performance of the proposed novel ssHAC algorithms.

The remainder of this chapter is organized as follows. Section 3.2 introduces the research background and the motivation for proposing a new algorithm for semi-supervised clustering. Section 3.3 provides the details of the algorithm. Section 3.4 outlines the experimental methodology and the measures used to evaluate the results. The results are discussed in section 3.5 and the conclusions are made in section 3.6.

3.2 Background and Motivation

HC techniques and other similar clustering techniques have the drawback that they can return information (i.e. cluster data) which is not relevant to the real-world user. The solution is to use prior knowledge about the specific data to apply constraints to the clustering hierarchy used to extract useful information from the input data. These constraints usually apply between data pairs. The literature refers to this prior knowledge as background [2], [112], [142]–[144]. The ssHAC method is based on HAC algorithm [1], [5], which uses tree structures, known as dendrograms, to represent the clustering result. The dendrograms are generated using linkage measures. There are a variety of methods of linkage measures, each of which determines the measurement in its own way, which means they can return different results from the same input data. [50], [51], [145].

The existing literature includes several studies which compare different agglomerative linkage measures applied to unsupervised HC algorithms [146]–[148] and semi-supervised hierarchical agglomerative (ssHAC) algorithms with pair-wise constraints [58], [121].

Hands and Everitt [147] explore the differences among five unsupervised agglomerative linkage measures using binary data, concluding that using Ward's linkage method [59] is the most effective. Ferreira and Hitchcock [146] determine that Ward's method is generally the most optimum, though other situations may require the use of average linkage methods, for instance, when the number of clusters was above specified recovery. Milligan and Cooper [148] conduct a similar study, using four agglomerative HC methods, determining that the least effective technique is using single linkage. The best recovery overall is found using Ward's method.

Existing studies for semi-supervised HC with agglomerative linkage measures typically concentrated on background information using ML and CL constraints. By definition, a

ML requires that two instances are included in the same cluster, and a CL requires that the two instances must not be clustered [58], [121]. It is important to note that ML and CL constraints are not compatible with HC methods because objects are linked at different levels [11]. This present study makes two novel contributions. Firstly, this study investigates the effects of different similarity/distance measures on the performance of ssHAC in combination with different linkage measures. Secondly, this study introduces external knowledge in the form of triple-wise relative constraints, which are more suitable than pair-wise constraints (CL and ML) for HC methods.

Previous studies have determined that there is no single ideal distance metric that is effective for clustering all types of data. The characteristics are dependent upon the specific data set or context of the problem, meaning that problems of clustering may require investigation of the use of a variety of distance metric measures. Additionally, it can be quite complicated to choose the exact distance when using clustering for data mining applications. It is critical for researchers to have a full understanding of the parameters for each distance measure considered [55], [65]–[67].

The choice of distance metric and linkage to use with ssHAC algorithms influences the clustering results, and therefore this choice should be made based on the nature of the application. The results of ssHAC algorithms can be visualised as a tree diagram, or dendrogram, in which the inner nodes represent nested clusters containing varying numbers of objects. The linkage measure determines which objects and clusters should be combined together in order to build a hierarchical dendrogram cluster. It does this based on the similarity between them, which means it needs methods for measuring that similarity. This is why distance metrics are important, because they quantify similarity, so they can be used to assess the similarity between a data pattern and a cluster, which then provides additional structural information. The degree of similarity determines how strongly a data pattern belongs to a given cluster. Although there are many methods for measuring similarity, this study focuses on the most popular ones, which were outlined in chapter 2.

This raises the question of to how to choose an appropriate distance metric to use with various linkage methods in order to obtain optimal results from ssHAC algorithms. To answer this question, this chapter proposes an integration scheme incorporating a series of different distance measures using semi-supervised clustering. To the best of the authors' knowledge, this study is the first to demonstrate that the semi-supervised clustering method using constraints can combine multiple distance metrics with different linkage measures. Most of the comparative studies in the existing literature have focused on comparing distance metrics in terms of their effects on both unsupervised hierarchical and non-hierarchical clustering [80], [149]–[153]. There are comparatively few studies on semi-supervised PC, and these only explore flat clustering using semi-supervised fuzzy c-means (ssFCM) algorithms [154].

Mohammed and Abdulazeez [150] analyse Euclidean, Manhattan, Minkowski, Cosine, and Mahalanobis distance measures and their effect on the partitioning around medoids (PAM) algorithm for microarray datasets. Their research involved three different types of gene expression containing colonic epithelium, epididymal fat tissue and hematopoietic stem cell (HSC). They concluded that the Manhattan distance measure performed "better" than the others. The Mahalanobis metric also applies a geometric adjustment to the distribution of the data to minimise the distance between similar points, which means that the Mahalanobis distance is the most efficient distance metric for detecting compact gene clusters in microarray datasets when integrated with the PAM algorithm, which outputs optimal cluster solutions for various k partitions. Five techniques of agglomerative linkage measurement were explored using Euclidean and Manhattan distance by Morlini and Zani [151]. The results of that study showed that no one distance measure was significantly better at performing agglomerative linkages.

The impact of various distance metrics for both hierarchical and non-hierarchical clustering algorithms, including Minkowski, Euclidean, Manhattan, Chord, Average, Pearson correlation, Cosine, and Mahalanobis is analysed by Shirkhorshidi, Aghabozorgi, and Wah [152]. In particular, they applied the metrics to k-means and k-medoids algorithms as PC algorithms, and HC algorithms (Single-link and Group Average) algorithms and compared them using the aforementioned distance measures on 15 datasets which were categorised as low- and high-dimensional. Their results showed that Average method was the best-performing distance metric, and that Pearson correlation does not work properly when applied to low-dimensional datasets but gives better results for high-dimensional datasets. Vakharia and Wemmerlöv [153] compare seven HAC methods (namely: Ward, centroid, median, set merging, single, complete and average linkages) and evaluated their performance in conjunction with five dissimilarity measures (parametric dissimilarity, Jaccard dissimilarity, squared Euclidean, and Euclidean distance measure) on 24 binary data sets. There was no single best combination of clustering method and distance metric, but instead the best combination varied across data sets. By using 24 different data sets, the researchers were able to rule out some techniques and measures as unsuitable for clustering, whereas others were found to be relatively reliable and robust. Not only is there no single combination that is best for all data sets, but also, there is no single clustering technique and no single distance metric that was found to be best in all cases. The researchers explained that the performance of the clustering techniques is dependent on the dissimilarity measures and datasets used. The performance of K-means combined with various distance metrics was investigated by Singh, Yadav, and Rana [80]; and Bora, et al. [149]. By using Euclidean, Manhattan, and Minkowski distance metrics and k-means, dummy dataset was evaluated by Singh, Yadav, and Rana, [80]. Results of their experiments indicated that using Euclidean distance metrics with K-means shows better outcomes than using Manhattan distance metrics [80]. Bora et al. [149] explore Cityblock, Euclidean, Cosine, and Correlation metrics on two different data sets taken from the UCI machine learning repository. The findings showed that using Cityblock distance resulted in better performance for both datasets, because of the shorter computation time. The Cosine distance required additional computation time compared with other distance metrics. The outcome of correlation distance measure shows a clearer interpretation of the clustered data. Lai and Garibaldi [154] ran some distance-based ssFCM algorithms on the UCI dataset. They compared four algorithms using three distance metrics: Mahalanobis distance by Pedrycz-97 and Li-08, the kernel-based distance by Zhang-04 and Euclidean distance by Endo-09. However, they did not compare the three distance metrics on a single ssFCM algorithm.

Further to the above literature review, this study makes the following contributions.

• Empirically investigates the effect of various distance metrics on the performance of ssHAC algorithms when used with different linkage measures and using triple-wise relative constraints as a way of understanding the internal structure of the various datasets.

• Determines which combination of clustering algorithm, distance metric and linkage metric are the best, on the basis of a non-parametric statistical test.

• Provides insights and recommendations to help future researchers choose an optimal combination of linkage method, distance metric, and the number of constraints for their own datasets, when applying with ssHAC algorithms.

3.3 Proposed Approach

For the current study, the proposed method intends to perform clustering using knowledge-based triple-wise relative constraints. Specifically, knowledge is included that allows for a variety of instances (S) and constructs Cluster Order constraints(C) that are based on pairwise dissimilarities (D). The relationship can be expressed as $S = \{s_1, s_2, \cdots, s_n\}$, $D = \{d(s_i, s_j) | s_i, s_j \in S\}$, $C = \{(s_i, s_j, s_k) \mid d(s_i, s_j) < d(s_i, s_k), s_i, s_j, s_k \in S\}$.

The aim of this method is to construct accurate tree hierarchies to satisfy numerous constraints, while maintaining the order of the individual pairs based on their dissimilarities. As developed by Zheng and Li [11], the Transitive Closure Constraints and Removing Conflict Constraints methods for pre-processing constraints are implemented.

The design of the proposed ssHAC algorithm is based on well-known HAC algorithm, with the addition of relative triple-wise relative constraints resulting in a novel ssHAC algorithm. The ssHC algorithms was originally introduced by Zheng and Li [11], as discussed in the previous chapter. We perform the conjunction between the ultra-metric transformation of dissimilarity matrix and the six agglomerative linkage measures: single linkage, complete linkage, average linkage, median linkage, centroid linkage and Ward linkage. The IPoptim and UltraTran techniques are used to incorporate the triple-wise relative constraints in order to modify and update the initial distance similarity matrix to convert it into the ultra-metric transformation of the dissimilarity matrix. The initial distance similarity matrix is calculated through selecting one of the ten distance measures namely: Cosine, Correlation, Manhattan, Euclidean, Standardized Euclidean, Mahalanobis, Spearman, Chebyshev, Canberra and Bray-Curtis.

The proposed algorithm (ssHAC) have been constructed using two constrained optimization techniques (UltraTran and IPoptim) which integrate the various parameters that result in the production of different solutions for clustering. The blend of various parameters is applied, employing each type of distance measure with different type of linkages methods and different amount of constraints. The algorithm proposed (ssHAC) for each of constrained optimization technique uses an amalgamation of different parameters to produce 360 methods of clustering process. Thus, framework of proposed algorithm (ssHAC) is able to produce different solutions for clustering based on the parameters used which lead to determine which best combination of the proposed clustering algorithm. The detailed steps regarding the proposed framework are presented below.

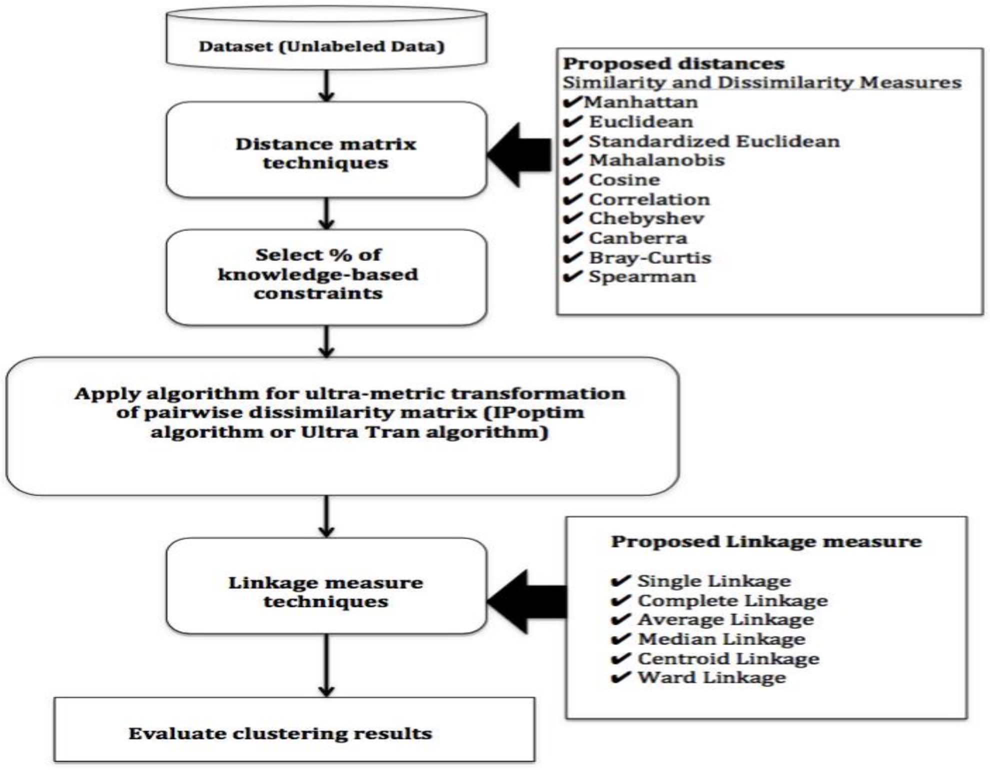


Figure 3. 1.Semi-supervised hierarchical agglomerative clustering (ssHAC) algorithm framework.

The framework of the proposed algorithm is shown in figure 3.1, which takes the dataset, the distance measures, and lists of triple-wise relative constraints as inputs. It calculates the dissimilarity matrix of the given dataset and then implements order constraint on the distance matrix to update the dissimilarity metrics to fulfil the given constraints. This is followed by the application of the linkage measure on the updated distance matrix to output the hierarchical clustering, represented as a dendrogram structure. The steps of the framework for ssHAC algorithm are outlined below.

Step 1: Distance computation: This step involves choosing one of the distance measures (see Chapter 2, Section 2.3) to calculate the proximity (dissimilarity) matrix between the elements within the dataset. The proximity matrix is matrix N*N that defines the distances (similarity) between all data objects N.

$$\mathbf{D=}\begin{bmatrix} \mathbf{0} & \boldsymbol{d_{12}} & \boldsymbol{d_{13}} & \dots\dots & \boldsymbol{d_{1n}} \\ \boldsymbol{d_{21}} & \mathbf{0} & \boldsymbol{d_{23}} & \dots\dots & \boldsymbol{d_{2n}} \\ \boldsymbol{d_{31}} & \boldsymbol{d_{32}} & \mathbf{0} & \dots\dots & \boldsymbol{d_{3n}} \\ \boldsymbol{d_{n1}} & \boldsymbol{d_{n2}} & \boldsymbol{d_{n3}} & \dots\dots & \mathbf{0} \end{bmatrix}$$

where d_{ij} represent the distance between points i and j and $d_{ij}=d_{ji}$

Step 2: Constraint Generating: This is the step where the generation of constraints occurs using the chosen samples of the dataset as a basis. For instance, where N represents the dataset size and the percentage of needed constraints is a %, the aggregate amount of constraints would be a* N /100.

Furthermore, the samples of constraint are generated based on the class label of each sample. Three samples are selected randomly from two different classes to create a constraint. For instance, if x_i, $x_j \in$ Class1 and $x_k \in$ Class2, then c = (x_i, x_j, x_k) is a triple-wise relative constraint.

Step 3: The constrained optimisation approaches: The UltraTran algorithm or IPoptim algorithm is used to seek for updating dissimilarity metrics based on the provided constraints. This step results in the ultra-metric distance matrix, which meets the requirements of the constraint rule. Chapter 2 (Algorithm 2.2 and Algorithm 2.3) describes the details of the two algorithms.

Step 4: Cluster merging: Once the ultra-metric distance matrix is constructed, the following step chooses one of the agglomerative hierarchical linkages (more details of them are discussed Chapter 2 in Section 2.2.2.1) to apply on ultra-metric distance matrix for the merging process.

Step 5: Validating and evaluating clusters: The final step involves the outputting of the hierarchical cluster dendrogram. The rate of accuracy (rate of true positives) is computed using matches between clustering results (which are obtained by ssHAC algorithm) and class labels for measuring the level of agreement.

3.4 Experiment

3.4.1 Experimental Methodology and Evaluation Measures

The experiments were conducted to evaluate the performance of proposed method (ssHAC algorithms) with triple-wise relative constraints on seven datasets of various dimensions and class numbers which are obtained from [155] and [156]. The specifications of the datasets are provided in Appendix -A.

The performance of the proposed ssHAC algorithm with IPoptim and UltraTran optimization techniques was rigorously evaluated using different combinations of distance measures, linkage methods and numbers of constraints. For each dataset the use of ten distance measures, six linkage methods and various amounts of constraints are investigated. It is however challenging to represent and scientifically compare such a large number of experiments. Therefore, this chapter presents and visualises the experimental results which can be demonstrated and discussed in a way that can easily be read and understood. This visualisation and presentation method proposed is outlined section 3.4.2.

The ssHAC algorithms were implemented using the R programming language. The ssHAC algorithms need to update the similarity matrix and restore the matrix. Many of the matrix calculation algorithms have running time of order $O(n^2)$ or $O(n^3)$ (where n is the number of data points), therefore, the Rcpp library with the C++ programming language was used to increase efficiency.

Because class labels were available for the datasets used in this experiment, F-measures (external measures; [75]) were calculated based on matches between ssHAC clusters and the given class labels to quantify the level of agreement with known labels. A Friedman test was performed with the post hoc approach proposed by Demšar, [136] to compare different combinations of algorithms, distance metrics and linkage measures. This test outputs ranks for each algorithm for each individual dataset, in order to determine whether any significant differences exist among the algorithms considered when applied to the given datasets.

3.4.2 Experimental Result Setup

The data points composing the constraints were chosen randomly and with replacement, that is, the same data object can appear in different constraints. Due to the randomness in the construction of the constraints, ten runs of the algorithms are performed, and the results are averaged, obtaining the final results. The constraints were chosen based on the percentage of selected samples in the dataset, for example, size N of samples, the number of constraints needed would be a x% value from N * x%/100. The algorithms are run with varying proportions of constraints for all datasets, ranging from 10% to 60%. All constraint sets are pre-processed to eliminate any conflicts.

Varying Distance Metrics: To evaluate the effect of the use of different distance metrics, the two most effective linkage methods were employed in our proposed method, which are single and complete methods and the number of constraints was fixed to 10%. The performance of both ssHAC algorithms with two techniques (IPoptim and UltraTran) was evaluated for the ten different distance metrics. The results are presented in Figure. 3.2

Varying Linkages: The effect of changing linkages on the performance of ssHAC algorithms with two techniques (IPoptim and UltraTran) was evaluated. Two most effecting distance metrics in the proposed method, which are Euclidean and Manhattan distance, are considered with 10% constraints. The results are presented in Figure. 3.3

Varying the Number of Constraints: The changing the number of constraints on IPoptim and UltraTran within ssHAC algorithms was evaluated. Here all ten of the distance metrics were used, but only one linkage was kept, namely the single linkage. The performance of Iris, Wine, NTBC and CTG, BCWD, and BCWO datasets are evaluated for the proposed method. The results are presented in Figure. 3.4.

Impact of High Dimensionality: High dimensional datasets on the two-ssHC algorithms were evaluated and analyzed. Here, all ten of the distance metrics were used, along with six linkage measures, and the number of constraints. The performance of BCWD, BCWO and NTBC datasets are evaluated for the two ssHC algorithms. The results are presented in Figures 3.2, 3.3, and 3.4.

Evaluating the Clustering Efficiency: The two types of non-parametric statistical tests are conducted on the most favourable 30 combinations of the two ssHAC algorithms, with the following distance measures: Manhattan, Euclidean, Correlation, Canberra, and Bray-Curtis. These were used with the following linkage measures: Single, Complete, Average, and Ward, with 10% constraints for all combinations. These achieved significant results in terms of F-score when employed to both IPoptim and UltraTran.

Firstly, the non-parametric Friedman test was used to evaluate the rejection of the hypothesis that all the classifiers perform equally well for a given level. It ranks the algorithms for each dataset separately, with the best performing algorithm receiving the higher ranking. Then, the Friedman [135] test compares the average ranks of the algorithms and calculates the Friedman statistic. If a statistically significant difference in the performance is detected, which means that some of the hypotheses in the experimentation have different distribution from one another.

The computation of ranks using the Friedman test is achieved through the following steps. i) The F-score results of the algorithms and the datasets used are represented into rows and columns respectively. ii) From each data set, all algorithms are taken into account in each column are checked in decreasing order. The value 1 is given to the algorithm with the highest f-measure value, while the algorithm with the lowest f-measure value in the column will be allocated value 30 (this is based on the fact that there is an aggregate of 30 algorithm combinations/ column). iii) In the event that more than one algorithm has come up with a similar f-measure value within a column, the average of numbers allocated to them will be taken. iv) Once all the algorithms have been assigned values for all of the dataset, the average of each algorithm's allocated values (that is, each column's average) will be taken. v) The algorithms will then be ranked according to the average value: Rank= (Iris+ Wine+ BCWO+ BCWO + Pima+ NTBC+ GTC)/number of datasets =7).

The next step will be to try to determine which pairs of the algorithms are significantly different than each other. Therefore,, a post-hoc test was conducted, using the Nemenyi, Holm, and Shaffer tests [137]–[140], to find out which of the tested methods are distinctive among an NxN comparison. The post-hoc procedure is based on a specific value on the significance level α. Additionally, the obtained p-value should be examined in order to determine how different given two algorithms are. The significance level was fixed at $\alpha = 0.05$ for all comparisons. Average rankings of the algorithms over 7 benchmark biomedical datasets produced by the Friedman test are shown in Table 3.1. The results achieved in post-hoc comparisons for $\alpha = 0.05$ are depicted in Table 3.2. The unadjusted values and adjusted p-values for Nemenyi, Holm, and Shaffer tests for NxN comparisons for the top ten algorithm combinations out of 435 combinations produced by the post-hoc test are placed in Table 3.2. The p-values below 0.05 indicate that respective algorithms combinations differ significantly in prediction errors.

3.5 Results and Discussion

The observations from the experiments performed over several datasets of varying size, class and dimension are as follows:

Choice of distance measure: It is clearly evident from Figure. 3.2 that Cosine distance performs poorly, plausibly due to its insensitivity to the scale of the data points (it only measures the angle between two vector points). Euclidean and Manhattan distance measures perform well among a range of datasets. However, Canberra appears to perform in a consistently favorable way in both IPoptim and UltraTran. It is difficult to tell whether these measures outperform others for an arbitrary dataset. One could be recommended to use standard distance measures such as Euclidean, Manhattan, or Mahalanobis distance first, and then to experiment with Correlation or Canberra (for widely separated clusters) or Chebyshev or Bray-Curtis.

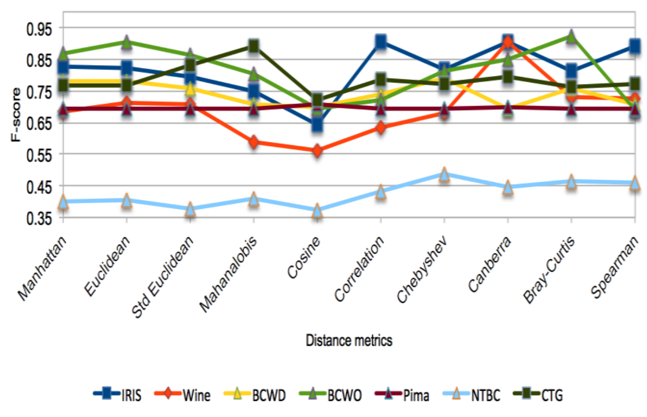
Choice of clustering algorithm: Both IPoptim and UltraTran perform almost equally for the seven datasets. However, UltraTran performs slightly better than IPoptim in some cases. However, the two algorithms may be experimented on more diverse datasets to evaluate their difference in performance, which appears to be minor. Any one of these two algorithms may be recommended.

Choice of Linkage: It can be observed from Figure. 3.3 that single, complete, or average linkages perform consistently better over most datasets. Median and centroid linkages should be avoided, as they are shown to degrade the F-score considerably. Ward linkage sometimes proves to be a good measure (even better than the first three), but it possibly works better only when there are no outliers, as suggested in Gan, Ma, and Wu [49]. Therefore, in the proposed work Single, complete, average, and Ward linkages are recommended to be used.

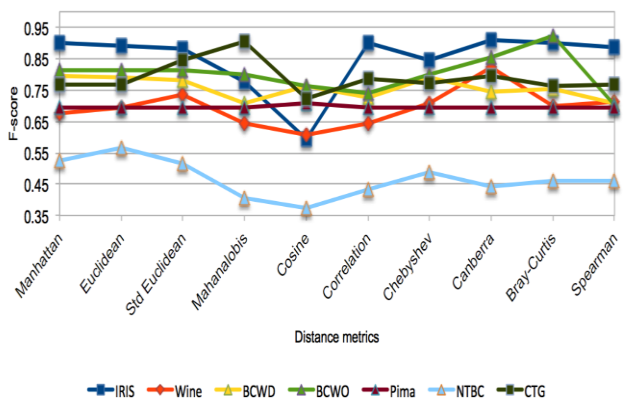
Choice of constraints: Figure. 3.4 shows interesting trends that the performance of ssHAC algorithms sometimes drop when increasing the numbers of constraints. This effect seems to be stronger for small datasets (such as Iris or Wine) than for large datasets (such as GTC). The safe threshold would be to use 30% constraints, after which the performance of ssHAC algorithms decreases in the case of most datasets. Furthermore, the same amount of constraints provides different accuracy values when employing different distance measures.

Evaluating the impact of high dimensionality: In general, increasing the data dimension and data size would be expected to degrade clustering performance significantly because of the so-called "curse of dimensionality". In the current experiment, simply comparing arbitrary datasets with different dimensions cannot result in any meaningful conclusion, because of the differences between the datasets. For example, three datasets in a higher dimensional dimension are compared (32D for BCWD, 21D for NTBC, 10D for BCWO). The BCWO and BCWD datasets have the same number of classes (2 classes), and the performance of BCWO in a 10-dimensional space is outperformed in comparison to the performance of BCWD in a 32-dimensional space. However, F-score performance on the 25-dimensional NTBC dataset is considerably lower than for the other six datasets, especially with high-dimensional datasets such as the 32-dimensional BCWD with 2 classes. Therefore, a new analysis is conducted on the NTBC dataset after reducing 6 classes to 2 (by merging the first three classes and the remaining three classes into two larger classes) and to 3 classes (as discussed in Soria, et al., [156]) respectively. This resulted in an increase in the F-score from 50-55% to 80-90%. This could mean that the ssHAC algorithms cannot directly identify these subgroups. It may not be premature to infer that datasets with too many clusters may be vulnerable to decreased performance in ssHAC algorithms.

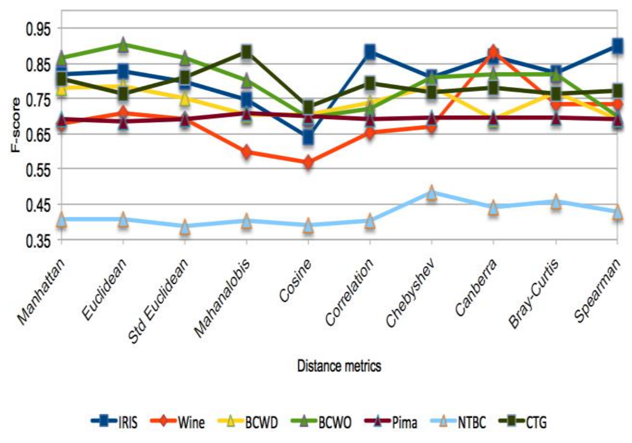
Choice of appropriate combination of ssHAC algorithmss with linkage measure and similarity metric for given dataset: Table 3.1 shows the output of Friedman test that determines the average ranking of different combinations of ssHAC algorithms and the significant difference in performance across the reported observations. The average ranks of the algorithms reveal significant differences in performance for IPoptim and UltraTran techniques with different linkages and distance metrics. For example, there is a difference rank among UltraTran and Canberra distance with complete linkage, average linkage, and Ward linkage. Moreover, it is also observed from the Table 3.1 that among all linkage methods, the Ward linkage method performs consistently better in comparison to other linkage measures for given datasets. Additionally, the difference among average ranks of different combinations is a motivation for conducting an investigation into which pairs of algorithms differ significantly. Hence, post-hoc analysis has been conducted and the output of the test is presented in Table 3.2. Nemenyi, Holm, and Shaffer tests have been used to demonstrate the significance of the NxN comparisons for top ten combinations of algorithms out of 435 combinations produced by the post-hoc test are placed in Table3.2 (with p-value < 0.05). These tests have been conducted on F-score values and the output is the p-value for Nemenyi, Holm, and Shaffer which is presented. The lower the p-value (p-Nemenyi, p-Holm, p-Shaffer), the better the performance of the given algorithm / combination. It can be seen also from Table 3.2 that Ward linkage with Manhattan distance is the best combination for the given datasets. These analyses also match the output of the Friedman test and the F-score analysis. Hence, it is advised to use ssHAC algorithms with Ward and Manhattan distance for given datasets to effectively cluster the data.



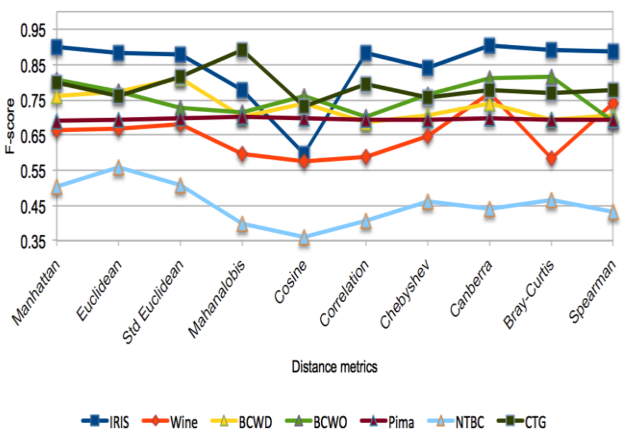
(a) IPoptim with single linkage



b) UltraTran with single linkage

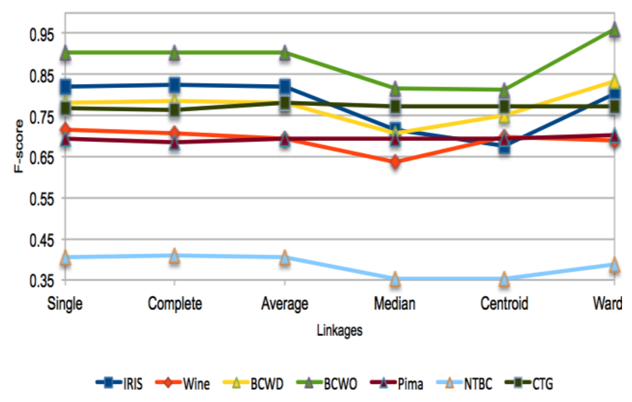


(c) IPoptim with complete linkage

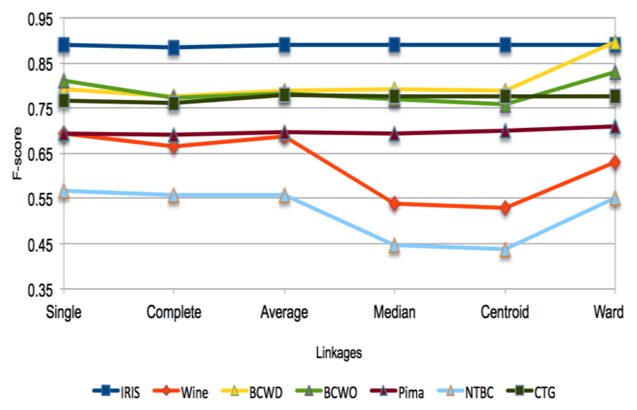


(d) UltraTran with complete linkage

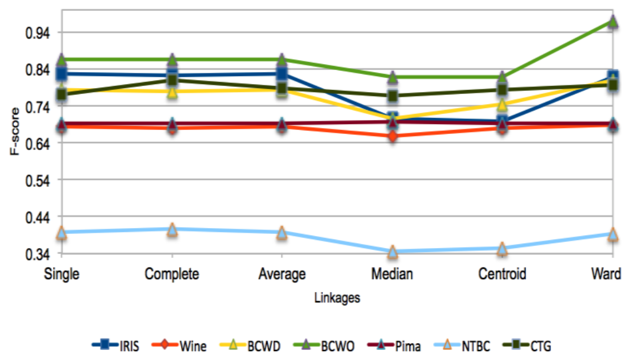
Figure 3. 2. Performance of different distance measures on ssHAC algorithms.



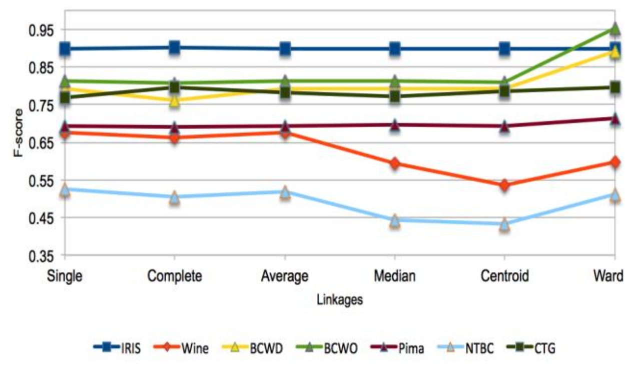
(a) IPoptim with Euclidean distance



(b) UltraTran with Euclidean distance

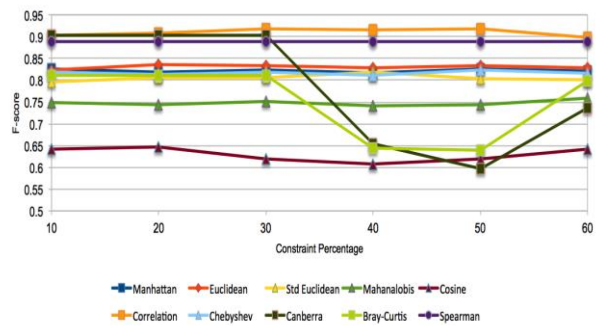


(c) IPoptim with Manhattan distance

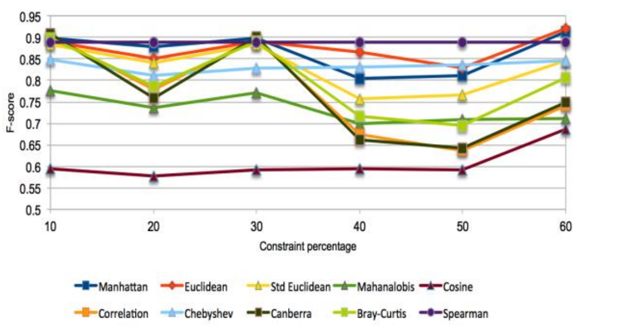


(d) UltraTan with Manhattan distance

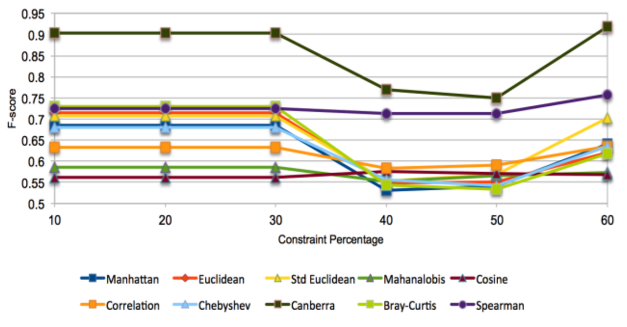
Figure 3. 3. Performance of different linkages on ssHAC algorithms.



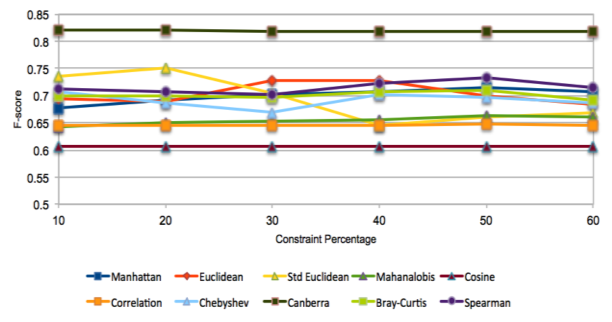
(a) Iris IPoptim



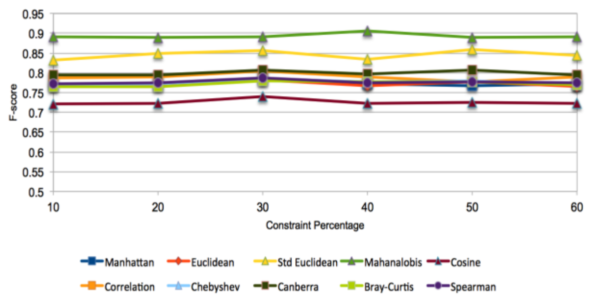
(b) Iris UltraTran



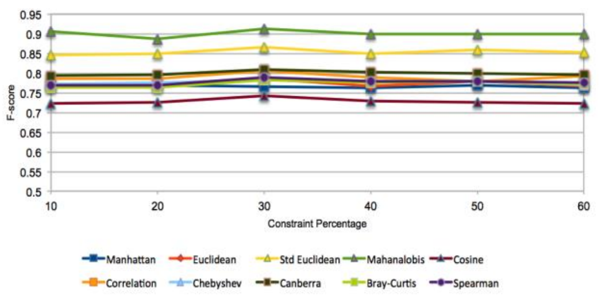
(c)Wine IPoptim



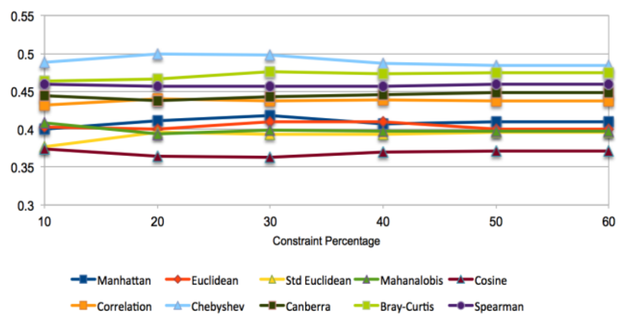
(d)Wine UltraTran



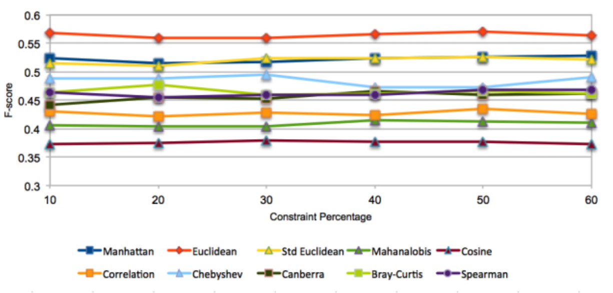
(e) CTG IPoptim



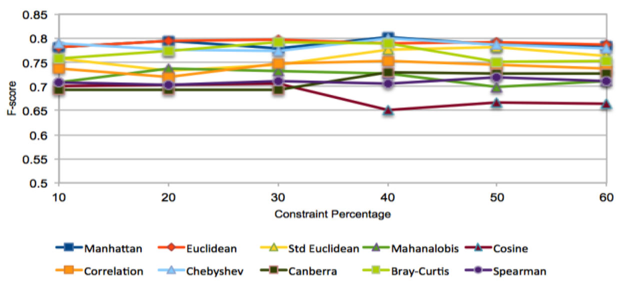
(f) CTG UltraTran



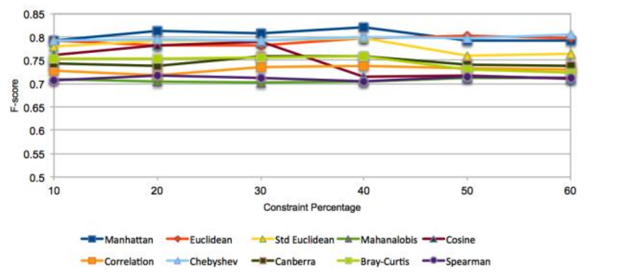
(g) NTBC IPoptim



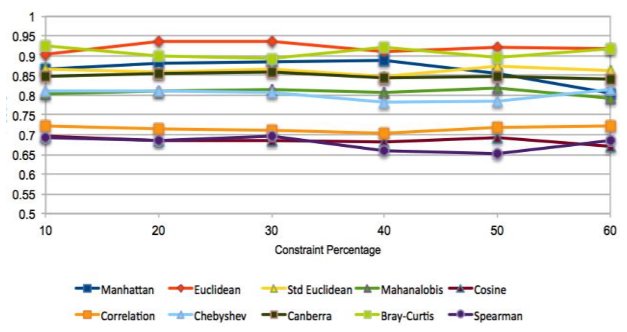
(h) NTBC IPoptim



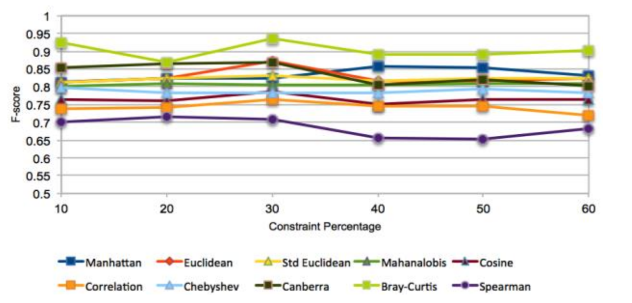
(i) BCWD IPoptim



(j) BCWD UltraTran



(K) BCWO IPoptim



(l) BCWO UltraTran

Figure 3. 4. Performance when varying the number of constraints on ssHAC algorithms.

Table 3. 1. The monotonic of average rankings of the IPoptim (I) and UltraTran (U) algorithms with distance measure (Man- Manhattan, Eucl- Euclidean, Corr-Correlation, Canb -Canberra, and Bray-Bray-Curtis) and linkage mechanism (Comp- complete, Avg- average and Ward). * Represents the best performing algorithm with distance measure and linkage mechanism.

Algorithm	#Rank	Algorithm	#Rank
U-Man-Ward*	7.00	U-Man-comp	11.07
I-Bray-Ward	11.92	U-Eucl-Ward	12.28
U-Canb-Ward	12.85	I-Canb-Ward	12.85
U-Eucl-Avg	13.07	U-Canb-comp	13.28
U-Canb-Avg	13.71	U-Bray-Avg	13.71
I-Man-Ward	13.78	I-Canb-Comp	14.50
I-Canb-Avg	14.64	U-Man-Avg	14.64
I-Bray-Avg	14.86	I-Eucl-Ward	15.28
U-Bray-Ward	16.00	I-Man-Comp	16.14

I-Corr-Comp	16.50	I-Bray-Comp	16.64
I-Eucl-Avg	17.07	U-Eucl-Comp	17.35
U-Corr-Ward	17.42	I-Man-Avg	17.42
I-Eucl-Comp	18.78	U-Corr-Avg	19.00
I-Corr-Ward	19.71	I-Corr-Avg	20.14
U-Bray-Comp	21.28	U-Corr-Comp	22.00

Low rank = Best performance
High rank = Lowest performance

Table 3. 2. Adjusted p-VALUES (*Represents the best performing algorithm with distance measure and linkage mechanism).

I	Hypothesis	Unadjusted p	p-Neme	p-Holm	p-Shaf
1	U-Man-Ward vs U-Corr- Comp	0.001434*	0.6239	0.6239	0.6239
2	U-Man-Ward vs U-Bray-Comp	0.002398	1.0433	1.0409	0.9737
3	I-Corr-Avg vs U-Man-Ward	0.005222	2.2716	2.2611	2.1201
4	I-Corr-Ward vs U-Man-Ward	0.006894	2.9988	2.9781	2.7988
5	U-Man-Ward vs U-Corr-Avg	0.010768	4.684	4.641	4.3718
6	I-Eucl-comp vs U-Man-Ward	0.001434	5.3326	5.2713	4.9771
7	U-Man-Comp vs U-Corr-Comp	0.020209	8.7908	8.6695	8.2047
8	I-Man-Avg vs U-Man-Ward	0.026678	11.605	11.418	10.831
9	U-Man-Ward vs U-Corr-Ward	0.026678	11.605	11.418	10.831
10	U-Man-Ward vs U-Eucl-Comp	0.027735	12.065	11.815	11.26

3.6 Conclusion

This chapter presented a novel algorithm (ssHAC) which are designed on well-known HAC algorithm and with the addition of knowledge-based relative triple-wise relative constraints. The ssHAC was tested on real-world datasets by varying the ratios of triple-wise relative constraints so that the performance of the IPoptim and UltraTran algorithms could be evaluated on the ultra-metric transformation of the dissimilarity matrix using given triple-wise relative constraints. The test was conducted using six different linkage measures and with ten different distance metrics. The experimental results confirmed that the proposed ssHAC method can produce different dendrogram clustering results, where the differences depend on which linkage measure and distance metric are used. Hence, the experimental results conclude the existence of a relationship between the distance matric factor and the linkage measure technique used in ssHAC algorithms. A Friedman test and Post-hoc analysis (a non-parametric statistical test) were used to validate the experimental results, which showed that the Ward linkage method with the Manhattan distance metric is an optimal combination. It should be noted, however, that it is difficult to identify a single set of measures that will work best for all datasets when using ssHAC algorithms. As a general principle, the research conducted recommends conducting initial exploration of ssHAC algorithm on a dataset using Euclidean, Standardized Euclidean or Canberra measures with single, complete, average or Ward linkages and a small number of constraints. The results also showed that increasing the number of constraints does not always have a positive effect on the performance of the ssHAC algorithm. The performance of ssHAC can sometimes decrease with higher numbers of constraints, and this effect seems to be stronger for small datasets than for large datasets. The results of this study, in accordance with those obtained by Zheng and Li [11], showed that the performance of ssHC algorithms depends on the quality of the generated constraints. Not all provided constraints carry the same degree of influence on clustering performance. Given these observations and conclusions, the next chapter will further investigate the optimization of the constraints affecting ssHAC algorithms on different datasets.

CHAPTER 4

An Effective and Efficient Constrained Ward's Hierarchical Agglomerative Clustering Method

4.1Introduction

The major problems concerning semi-supervised clustering algorithms involve challenges in incorporating adequate amount of high-quality knowledge through external information and minimizing the time required for generating constraints. In the previous chapter, experimental results demonstrated that the Ward method is the best method for ssHAC algorithm compared to other linkages. However, the performance of ssHAC algorithms sometimes decreases when numbers of constraints increase. Therefore, further investigation is needed on the optimization of constraints affecting the ssHAC algorithm. Further, due to eliciting as well as utilising knowledge-based constraints, there are considerable additional computational costs associated with ssHAC algorithm as compared to that of the AHC algorithm. Thus, it is difficult to apply the ssHAC algorithm on large datasets, particularly when providing a significant number of constraints.

Consequently, in this chapter, the emphasis is largely on the development of a novel Constrained, Ward's Hierarchical Agglomerative Clustering (CWHAC) algorithm that is able to optimise the information that the constraints carry, while at the same time decreases the time-consumption of creating constraints. Hence, two issues are addressed within the proposed CWHAC algorithm: firstly, the minimization of the computational time required to obtain the constraints for improving Ward's HAC algorithm efficiency; secondly, the challenges in specifying triple-wise relative constraints for boosting the performance of the standard, Ward's HAC algorithm.

4.2 Background and Motivation

The majority of practical applications can easily generate pairs of instances for generating constraints. This generation can be made randomly, as noted by [89], [10] and Zheng and

Li [11], or, it is possible to nominate two instances while a query is made by a user or domain expert, regarding a pair of points, if they must remain in the same cluster or in different clusters, in other words whether they are cannot-linked or must-linked nodes [83].

There have been numerous studies on the ways in which algorithms utilize user information for creating knowledge-based constraints. Cohn, Caruana and McCallum [103], for example, added the constraints regarding documents' initial PC. Such a method enables users to address questions such as whether a given document is not part of a cluster, whether two documents cannot or should not remain in the same cluster, and whether a document should be moved to another cluster or not. Such constraints are implemented in an algorithm that is founded on the Expectation Maximization (EM) algorithm by approximating the Bayes probability theorem for modelling the clusters. These constraints were also examined by Huang, Zhang, and Lam [157] in terms of the EM algorithm as well as including user feedback for an initial clustering for further refining the cluster model with local weights based learning. Huang and Mitchell's[158] study involved user interacting with clustering which was attained using a probabilistic model named as SpeClustering. Here, the user was able to determine if an object belonged to a cluster or not. The use of feedback was extended in a subsequent study by Huang and Mitchell [117] to a hierarchical clustering process which enabled the user to determine the importance of deletion, addition, splitting, as well as fusion of clusters along with modifying the cluster membership of examples. Dubey et al.'s [159] study involved a K-means-based algorithm that included user feedback, Cluster-Level Interactive K-means algorithm (CLIKM) which involved the user using a K-means clustering result to determine an object's membership to a particular cluster and modifying the cluster centroids as per its domain knowledge.

On the other hand, some studies implemented an automatic random constraint extraction. The researches by [10,89] involve Constraints-Partitioning COP-COBWEB as well as COP-KMeans respectively. The studies randomly selected initial constraint through instances from the data set and generated two instances, after which their labels were checked. Moreover, a ML constraint was created in case of two instances having the same label, the CL constraint was developed, if not. The studied showed that clustering accuracy can be enhanced by the randomly generated constraints. It is, however, important to note that this improvement is dependent on the data set. It is possible to note improvements even on unconstrained instances, if the constraints can be generalised regarding the entire data set [89]. Zheng and Li [11] further explore constraints concerning Hierarchical Clustering. They generated automatic random constraints from the given data labels using three samples from two different classes to generate a constraint. For instance, if x_i, $x_j \in$ Class 1 and if $x_k \in$ Class 2, then $c = (x_i,x_j,x_k)$ becomes a triple-wise relative constraint. Hence, every generated constraint is founded on the actual class label information and it must depict the domain knowledge.

It should be noted the constraints in several practical applications tend to be given by users who are not aware of the data's spatial disposition, resulting in useless or redundant constraints which do not help in improving the clustering results [160]. Brea [22] explores the problems regarding the two methods to generate constraints. The automatic constraint extraction methods typically function through the generalisation of, to an extent, explicit concepts regarding the data domain to cluster. For example, two text documents with the same author and source should be in the same cluster. As it is evidently possible for such generalisations to always be plausible, it is often noted that. The constraints being generated through user input of information can include mis-judgements. Generally, clustering can be considered as an exploratory tool, which is why it is implemented when the data configuration and data structure is not well-known. Hence, it is not always clear whether two data instances should be in the same cluster. This can be further exacerbated if there are multiple users participating in the creating the constraint as they may have significant differences regarding their criteria of data configuration. Therefore, the algorithms' robustness concerning noisy constraint sets which involves erroneous constraints often has a significant impact in their overall efficiency [22].

Lei et al. [161] state that sometimes users are not well-informed about the clustering algorithms and they just utilize them in the analysis. It is thus not possible for the constraint parameters to be accurately altered for achieving the business goal. Further, there are other studies that have relied on the expert for establishing the parameters at first. However, regarding a big data field, there will a large number of data vectors. Various users tend to focus on the diverse features of the data because it is possible for them to belong to diverse domains. It is difficult to include all probable constraint parameters concerning each domain. To address this, the user's target/aim regarding the outcomes of the clustering process, must be considered.

Further, according to Nogueira [162], "It is not always true the human supervision can be considered the "ideal" users, i.e., users that do not insert wrong or contradictory constraints" . This is because the human supervisor can make errors during interacting in the clustering process. The following are the mistakes that such an environment can include:

- Including a must-link constraint that has two instances with distinct concepts
- Two instances from the same concept having a cannot-link constraint between them
- Forcing conflicting constraints directly or indirectly between objects

There are several reasons for such mistakes to take place such as incorrect interaction concerning the constraints-posing process or misunderstandings regarding the problem domain [162]. It can be stated that when elements are grouped or rearranged, it may lead to unique and interesting dependencies that user expectations are not aware of.

It is important to note that it is possible to improve the diverse semi-supervised clustering algorithms results after obtaining constraints through querying a user or a domain expert, which can be expensive, based on how much information the user has and how much effort is required by the user. Hence, the present study is based on Zheng and Li's [11] method for choosing constraints within the proposed method. On the other hand, the previous experiments (see Chapter 3) regarding ssHAC algorithm used Ward's method [19] is best approach for ssHAC algorithm comparing to other linkages. However, performance of ssHAC algorithms sometimes drop when increasing numbers of constraints. As stated by Zheng and Li [11], all constraints are not able to improve the performance of clustering uniformly. They noted that the performance of the ssHC algorithm does not increase monotonically as per the number of constraints. There are two possible reasons for this. First, the proposed framework in the present study intends not to fulfill all constraints but to identify a suitable approximation of the constrained ultra-metric. Second, the clustering performance relies on the generated constraints' quality [11].

Hence, the proposed study in this chapter intends to introduce unique constraint pre-processing methods for enhancing the CWHAC effectiveness. Such methods aim to address the triple-wise relative constraint problem in the CWHAC algorithm through decreasing/ reducing the number of violated and redundant constraints.

Selecting the constraint is crucial for improving the semi-supervised clustering. As stated by Wagstaff [163], calculating the constraint set quality is not trivial. It is difficult to identify constraint set properties which have a correlation with their utility. Numerous studies have indicated that improperly selecting constraints may degrade the resulting clustering instead of improving [45], [164]–[167]. As noted by Davidson, Wagstaff, and Basu [164], there are two constraint qualities which impact their performance, which are, informativeness and coherence. Informativeness concerns the amount of extra information which is provided by the constraints to the clustering process while coherence describes the extent of agreement within provided constraints which enhances the effectiveness of the clustering algorithm. The findings of Ganji [168] are in tandem with those of Davidson, Wagstaff, and Basu [164], as he noted that suitable sets of constraints provide high informativeness as well as high coherence and are able to provide the clustering process with better quality information.

Constraints that are randomly selected can create constraints which are unnecessary, redundant, or even harmful to the results of clustering [166]. It is important to prioritise the constraint so that it can be satisfied. The constraints that have higher priorities can be easily satisfied as there is higher frequency of constraint violations when there is an increase in the number of constraints [169]. In addition, enforcing the unnecessary or unsuitable constraints can result in no overall benefit and may also worsen the clustering performance, particularly if the constraints violate the underlying similarity space significantly [170]. Thus, certain constraint algorithms for clustering [89], [169], [171] strive to reduce the extent of constraint violation during clustering. That is, regarding all the algorithm iterations, the partitions should fulfill all the constraints. Hierarchical Clustering, for example, which is founded on K-means with pair-wise Constraints (HCAKC) attempts to reduce the number of violated constraints. Introducing the penalty factor for addressing the constraint violation alters the similarity metric. Hence, it is possible to identify a partition, which agrees with the provided constraints [171]. As noted by Wagstaff,et al., [89], the k-means algorithm is the foundation for the COP-k-means algorithm. This algorithm adds to the k-means algorithm a constraint violation checking process [89] that fulfills constraints through allocating every data point to the nearest cluster centre, in which case constraint is not violated by the assignment.

In general, compared to PC, the unsupervised hierarchical clustering (HC) algorithms require more computational cost in space and time for verifying all cluster association combinations [13], [14], [172]. Due to eliciting as well as utilising knowledge-based constraints, there are considerable additional computational costs for ssHC algorithm compared to (HC) algorithm. Because of this, it is difficult to apply the algorithm on large datasets, particularly when providing a significant number of constraints. Hence, apart from enhancing the CWHAC algorithm's effectiveness, this study also aims to enhance the CWHAC efficiency through recommending the new three-optimization principles for decreasing the time-consuming process concerning generating constraints.

4.3Proposed Approach

This study proposes the novel CWHAC algorithm (Section4.3.3) and the approaches adopted for improving its efficiency and effectiveness (Section4.3.1 and Section4.3.2).

4.3.1 Optimization the Initially Generated Constraints

This study proposes the optimization of the initially generated constraints, before pre-processing (see Section4.3.2), to increase the speed and decrease the computational costs for the final-generated constraints, which will lead to improving the efficiency of the proposed CWHAC algorithm (see Section 4.3.3).

It has been observed that some constraints are repeated due to the randomness of initial sampling. These repeated constraints mean that higher computational operations are needed to obtain the unique constraints within each iteration of the clustering algorithm.

The triple wise constraint selection process of state-of-the-art semi supervised clustering algorithms can be presented as follows: The known class labels of the dataset are first used to generate the initial constraints. More specifically a pair of random points from the same randomly selected class is first selected and then the third point is selected from a different randomly selected class. The following steps provide further details of the approach used:

1. Number of required initial constraints N is determined. This is usually a given percentage of the total size of the dataset.
2. First elements in the pair will represent class of the elements which should have the same label the second element of the pair should represent another class.
3. For selected classes random selection of the elements from those classes was done. And so, triples were form (x_i,x_j,x_k).
4. It is verified that all the elements from the list are unique is not satisfied. If that’s not correct, then non-unique elements are removed and the process of the steps 1-3 will be repeated to get exactly N unique parameters.

It has been observed that some constraints were repeated in the above implementation of constraint selection that will subsequently lead to several problems.

1. The algorithm convergence can take a long time: as the samples are selected within classes completely randomly and the procedure may select elements that already have been added to the list.
2. It is required to recheck multiple times already selected elements within selected constraints to check if they are unique with the new constraints added to the list.
3. Another issue affecting the generation of unique constraints involves their non-uniform distribution. If the number of elements in class 1 is much smaller than in class 2, then there will probably be a high percentage of constraints applied to class 1 as the overall number of constraints available for this class is smaller. However, for best clustering performance the selection of a class for inclusion within constraints should be uniform.

In order to address the above issues related to the state-of-art semi supervised clustering algorithms, the following procedure is proposed:

a. Firstly, the total number of available constraints C is computed. For each class i with a total number of elements $n_{i,}$ it is estimated that the number of combinations of pairs available in that class would be (n_i-1) *n_i. The total number of possible constraints [x_i, x_j, x_k] where x_i and x_j are from the class i and x_k are from class j is

$(n_i-1) * n_i * n_j$. The sum of all combinations of these class pairs (class i, class j) will give a total number of constraints, C.

b. A rule was formed where each number in the range from 1 to C is assigned to a single unique constraint. So, when given the integer number from 1 to C it can be converted to the triple $[x_i, x_j, x_k]$ and then make the re-conversion back to the integer number from the triple. The translation is performed in the way that different constraints will be matched with the different numbers, and if we convert the same constraint multiple times, the same number will result always. This conversion rule is called "mapping". When the number is converted to the constraint, it can be said that the number was mapped to the constraint's space, and when the different conversion happens, it is assumed that the constraint was mapped to the number.
c. N random numbers without repetition are selected from the range from 1 to C. Using the rule from the step b, numbers are converted to the constraints. After that, there is no need to check if all the constraints are unique because unique numbers are being generated in the range, 1 to C, and, according to the conversion rule they will be converted to unique constraints. The constraints will also be selected proportional to the available number of elements in the class, as a specific class number is not used during generation.

For example, assume that there are three classes: class1 = {1,2}, class2 = {3}, and class 3 = {4}, with 1, 2, 3, and 4 being the data points in the set. The sum of constraints possible with the pairs from class 1 are computed as 2*(2-1) *1+ 2*(2-1) *1, which are four possible constraints. Moreover, no constraints are possible with pairs of elements from classes 2 or 3. The number of random constraints generated from classes 1 and 2 would be [1,2,3] and would be converted to [2,1,3], and from classes 1 and 3, it would be [1,2,4] and converted to [2,1,4]. The total of unique constraints are [1,2,3] and [1,2,4].

The next step in state-of-the art constraint selection approaches would be to examine whether the constraints fulfill the following rule $d(x_i, x_j) < d(x_i, x_k)$ and to filter constraints that do not satisfy this rule. The significant amount of time is dedicated to excluding the non-complying constraints that have already been checked and stored due to their initial selection. The number of these constraints can be very large and checking and storing them can take a considerable amount of memory and also time to check. In order to address this, we propose that in the search, rather than examining each individual element in terms of whether they satisfy the rule or not, the interval in its entirety to be is examined, and, if it is found that they are not satisfied, the results can be skipped.

For example, there are the full list of constraints (in square brackets) and numbers (that come after the hyphen and the closing square bracket) assigned to them: [1,2,3]-1, [1,2,4]-2, [2,1,3]-3, [2,1,4]-4, [3,4,1]-5. Non-helpful/satisfied constraints are [1,2,3]-1, [1,2,4]-2, [2,1,3]-3, [2,1,4]-4. Therefore, we need to generate the five storages. It should store number 5 in the pre-computed set for further selection and should store numbers 1,2,3,4 to remember that these numbers should be excluded from the analysis. Any new constraint will be verified at each step to check whether the numbers are equal to 1, or equal to 2 or equal to 3 or equal to 4 as they should be excluded, requiring 4 verifications in total. Alternatively, when the algorithm is working in the way that only a few numbers correspond to the non-helpful constraints we can store them in a more compact way as intervals. If we store these four numbers as an interval between $1 \leq x \leq 4$: we only need to store two numbers 1 and 4 instead of 1,2,3,4. Further, we also need only two verifications which lead to save time and memory.

The third step is to switch off the expansion and conflict removal examination for the initial stage of the constraint selection, as a significant percentage of time spent is used to expand constraints. Within this stage of the process, the constraints are examined in order to evaluate whether they can be produced utilizing the following rule: $d(x_i,x_j) < d(x_i,x_k)$ with the form $[x_i, x_j, x_k]$ where element x_i and x_j are from the same class and element x_k is from different class. Basically, the initial constraints must be generated based on the above rule. Thus, initial constraints cannot be broken this generation rule. Hence, the expansion examination step is not applied to the initial stage. For example, let's assume we have 2 constraints $[x_i, x_j, x_k]$ and $[x_i, x_k, x_l]$. Essentially, they are equivalent to enforcing to conditions on the distances within the system: $d(x_i,x_j) < d(x_i,x_k)$ and $d(x_i,x_k) < d(x_i, x_l)$ respectively . Combined they give the rule of the $d(x_i,x_j) < d(x_i, x_l)$. Thus, the constraint representing this condition should include in the form of $[x_i, x_j, x_l]$. However, we discussed above that elements x_i and x_j are taken from the same class and elements x_i and x_k as well as x_j and x_k will be from different class. Using this generation algorithm there is no way we have a constraint produced in form $[x_i,x_k,x_l]$ as that would mean that xi and x_k are from the same class. This statement contradicts with our previous assumption so there is no need for further verification of the condition and so there is no need to apply the constraint expansion examination step on the initial constraint generating.

Similarly, the conflict constraint removal examination is also not applied to the initial constrain due to this convention xi and x_j to be taken from the same class and xi and x_k from different class and its relation to the rule (d $(x_i, x_j) <$ d (x_i, x_k)), as constraints that concurrently say $d(x_i,x_j) < d(x_i,x_k)$ and the opposite condition $d(x_i,x_k) < d(x_i,x_j)$ cannot be present; therefore, this pre-processing examination can be erased from the initial stage.

4.3.2 Constraint Setup and Pre-processing

The amount of chosen constraints is based on the percentage of selected samples of the dataset. For example, if N is the size of the dataset and the percentage of required constraints is a%, the total number of constraints would be a* N /100.

It can be considered that elements belonging to the same class should be classed within the same cluster. For automatically generating constraint, the distance among those elements is less compared to other elements from a separate group. Therefore, three samples were randomly chosen from two different classes., for example, xi, $x_j \in$Class 1 and $x_k \in$ Class 2, then c = (x_i, x_j, x_k) can be regarded as a triple wise relative constraint, with the additional meaning that d $(x_i, x_j) <$ d (x_i, x_k). This constraint format in form of the 3 elements $[x_1, x_2, x_3]$ is convenient for generation as well as theoretical discussion. However, the distance property is symmetric d $(x_1, x_2) =$ d (x_2, x_1). That means that the same inequality d $(x_1, x_2) <$ d (x_2, x_3) can be written in 3 additional ways: d $(x_1, x_2) <$ d (x_2, x_3), d $(x_2, x_1) <$ d (x_2, x_3), d $(x_1, x_2) <$ d (x_3, x_2), d $(x_2, x_1) <$ d (x_3, x_2). It is more efficient and convenient to verify all the constraints at the same time. That's why instead of storing each constraint in form of the 3 elements $[x_1, x_2, x_3]$ with importance of the order taken into account, we will store each constraint in form of 4 elements $[x_1, x_2, x_1, x_3]$. This recording would also be equivalent to the inequality d $(x_1, x_2) <$ d (x_1, x_3). But now we will use all 4 representations: $[x_1, x_2, x_1, x_3]$, $[x_2, x_1, x_1, x_3]$, $[x_1, x_2, x_3, x_1]$, $[x_2, x_1, x_3, x_1]$ to exam all the constraints at the same time as well as to avoid additional verifications.

Table 4.1 Presents four pre-processing steps used in creating constraints which are finally considered in an effort to improve the quality of the triple-wise relative constraints used in the proposed work. These pre-processing stages will ultimately improve the outcomes of CWHAC algorithms. Conflict Constraints Removal and Transitive Closure, as originally proposed by Zheng and Li [11], as well as new strategies of Violated Constraints Removal and Redundant Removal Constraints, are utilized in this study.

Table 4. 1. Pre-processing steps used for the final generation of triple-wise relative constraints of the proposed work.

Pre-processing Method	**Task**
Transitive Closure or Constraint Expansion	Constraints generated at initial steps added to the information which they represent in an explicit way contain some additional information that can be extracted based on the general distance rules. For example, let's assume we have constraints $c_1 = (x_i , x_j , x_k)$, $c_2 = (x_j, x_l, x_s)$, $c_3=(x_i ,x_o, x_m)$, $c_4 = (x_k , x_s , x_i)$, and $c_5= (x_m , x_k , x_j)$, then the constraints once expanded would be $c_{2'} = (x_i,x_l,x_s)$, $c_{3'}=(x_j , x_o , x_m)$, $c_{4'} = (x_k , x_s , x_j)$, and $c_{5'}=(x_m , x_k , x_j)$ using transitivity property of the distances. Using this additional inexplicit rule, we can improve overall accuracy. The Floyd-Warshall algorithm, considering the original constraint, is employed to lengthen the con0straint set, as well as to uncover its transitive closure.
Conflict Constraint Removal	It is possible that the available constraints are conflicting. As an example, $c_1 = (x_i , x_j , x_k)$ and c2 = (x_j,x_k,x_i) illustrates this possibility, whereas, $c_1 = (x_i,xj,x_k)$, $c_2 = (x_i,x_k,x_l)$ and $c_3 = (x_i,x_l,x_j)$ create a cycle of merge directions. When attempting to detect a usable merging pair of clusters, the clustering algorithm may be unsuccessful, and, as a result, obstacles may develop should there be conflicts within the constraint set. To resolve this problem, conflicting constraints were iteratively and randomly removed, until conflicts were satisfied. Furthermore, the conflict Constraint can be appeared during merging the clusters when constraints are extended. for example, element x_5 is present which can be merged to the same cluster as x_1 and x_2, while

	constraint d $(x_5, x_6) < d (x_6, x_7)$ is also available which indicates that the new constraint d $(x_1, x_6) < d (x_6, x_7)$ can be created. Therefore, it is necessary to apply this pre-processing to verify the following rule: x_1 and x_6 to be taken from the same class and x_6 and x_7 from different class.
Redundant Removal Constraints	When certain elements x_i and xj, are merged into single clusters, the distance between them is reduced to 0, as $d(xi,x_j) = 0$, and, therefore, it is not required to re-inspect for constraints such as constraint $[x_i,x_j,x_k]$ with rule $d(x_i,x_j) < d(x_i,x_k)$. This is redundant, and removal is necessary. Moreover, the expansion constraints may be repeating. For example, $c_{1=}(x_i,x_j,x_k)$ and $c_2 = (x_j,x_i,x_k)$. The redundant or unnecessary constraints can lead to failure to identify a valid merging pair of clusters. To remove redundant constraints, we filter this constraint out. This is another reason to use this method; to process non-unique constraints. Repeated constraints may appear after merging which will affect unique constraints. As an example, if the merging of the two elements x_4 and x_2 leads to them being made into one group, and $c_2 = (x_1, x_4, x_3)$ and $c_1 = (x_1 , x_2 , x_3)$ are handed the original constraint alongside the rules $d(x_1, x_2) < d (x_1, x_3)$,$d (x_1, x_4) < d (x_1, x_3)$, these constraints would be identical; therefore, one of them needs to be removed.
Violated Constraint Removal	Random selection of constraints allows for the possibility of constraints to meet conditions; however, this is not true for useful constraints if they do not affect or update the distance matrix. Therefore, utilising these constraints provides no advantage compared with using these constraints first within the algorithm, especially if only a small percentage of the amount of constraints are chosen. For example, use of the constraint d $(x_i, x_j) < d (x_i, x_k)$, where $d (x_i, x_j) = 2$, $d (x_i,x_k)= 4)$, does not impact the outcomes. The distance would not be updated because the elements used in the original distance matrix (for elements d $(x_i, x_j) < d (x_i, x_k)$ would match the rules and conditions of input. When users select a large number of constraints, additional problems arise. As most of these constraints will have probably already satisfied the conditions, they may not be useful.

	Violation of selected constraints may occur after completion of the merge. This occurs because elements taken from different classes within the same constraints are able to merge, though other elements nearby do not have constraints attached. For example, if the elements 1, 2, and 3 derive from class 1, the elements 4, 5, and 6 derive from class 2, and elements 7, 8, and 9 derive from class 3. Also, if the constraints x_1, x_2, and x_4, as well as the constraints x_1, x_3, and x_4, after merging elements 3 and 5 from different classes are utilised (because the distance between two elements at that point is minimal and there are no constraints attached to them, the new constraint is obtained (x_1, x_5, and x_4). With elements 3 and 5 in the same cluster, all elements would share all the conditions for merging. Following the merge, any constraints that were violated should be filtered and checked.

4.3.3 Constrained Ward's Hierarchical Agglomerative Clustering (CWHAC) Algorithm.

The proposed CWHAC algorithm takes a set of data S, the number of clusters k and lists triple-wise relative constraints as input. The main steps of generating triple-wise relative constraints are presented in Algorithm 4.1.

The proposed CWHAC algorithm is based on the traditional HAC algorithm beginning with each data object in a separate cluster. Two clusters progressively merge after the algorithm starts from a trivial partition composed solely of singletons. The constraints are incorporated into the Ward's hierarchical clustering algorithm method to update the distance matrix, and the constraint issues are addressed. The CWHAC algorithm is based on the ultra-metric transformation of dissimilarity matrix, which exploits the triple-wise relative constraints as background knowledge to create a new metric for data similarity. The CWHAC algorithm can be classed as the 'hybrid' (also known as search and distance based) method. When the constraints are given, the IPoptim or UltraTran [11], methods seek to update dissimilarity metrics to fulfill both the transitive property of the distance and the constraints provided at hand. Meanwhile, the traditional Ward clustering algorithm function serves to merge clusters utilizing an objective function with minimum distance (which was presented in Equation 2.8 in Chapter 2) and alongside distance, within search-based methods; nonetheless, if they are found to violate constraints, the step of merging clusters with minimum distance within current use can then be skipped. Thus, the algorithm would be adapted by filtering out the violated constraints mentioned above and would instead seek for others. The clusters are updated during the execution of the algorithms (e.g., to begin with, when eliminating duplicates, a number of elements are classed into one group; notably, the elements that are allocated to the same clusters are substituted for their centroid, thus leading to them being signified by their centroid). Therefore, in certain stage if two elements have been previously merged into a single cluster, it is essential that the constraints are updated). This is because the new interesting dependencies may be appearing due to elements are regrouped. The proposed algorithm (CWHAC) is shown in algorithm 4.2, below.

Algorithm 4.1: Generating triple-wise relative constraints

```
Function Generate Constraints (S, $C_s$, K)
 Input: K – number of constraints to generate, $C_s$ – correct class assignment, S--
initial dataset.
Output:  $C_O$- list of triple-wise relative constraints.
  1. Calculate total number of possible constraints of required format N. Required
     format is S= [$S_i$, $S_j$, $S_k$] where $S_i$ and $S_j$ are from the same class ($C_s$ [$S_i$] == $C_s$
     [$S_j$]), and $S_k$ is from a different class ($C_s$ [$S_i$]! = $C_s$ [$S_k$]).
  2. Calculated distance matrix D where d [$S_i$, $S_j$] = distance between points $S_i$ and
     $S_j$.
  3. E= {} – empty set of the numbers corresponding to the constraints, which nee
     d to be excluded from the analysis.
  4. K_left = K.
  5. Generate K _ left numbers from range between 1 and N excluding numbers fr
     -om E.
  6. Based on the rule assigning each number between 1 and total number of const
     -raints to the unique constraint [$S_i$, $S_j$, $S_k$].
  7. Convert each constraint of format [$S_i$, $S_j$, $S_k$] to the format [$S_i$, $S_j$, $S_i$,$S_k$].
```

```
8. Examine all constraints for the condition   d [$S_i$, $S_j$] < d [$S_i$, $S_k$], disregard all
   constrains where this condition have already been satisfied. Add numbers cor
   -responding to these constraints to E. Add the remaining constraints into gene
   -rated list CO.
9. K _ left as K minus size of the constraints in the list ($C_O$) after removal.
10.      Repeat step (5), (6), (7) and (8) until either K_left = 0 or all the possible
   constraints from the list 1: N were tested.
11.      Return $C_O$.
```

Algorithm 4.2: Constrained Ward's Hierarchical Agglomerative Clustering (CWHAC)

```
Input: set S = {$S_1$, $S_2$, ..., $S_N$} of N data instances.
Set:  $C_O$= { $c_{o_1}$,..., $c_{o_m}$} of triple-wise relative constraints, k - number of clusters.
Output: Hierarchical cluster dendrogram.
1. Presume initial clusters are demonstrated by: S = {$S_1$,$S_2$,...,$S_N$}, with each
   element made up of an individual data entity; the original cluster amount
   being represented by N and the amount of clusters being required to produce
   being K, Set K = N.
2. The distance between $X_{[j]}$ and $X_{[i]}$ is represented by $d_{ij}$ element of calculated
   matrix D.
3. Repeat
4. Utilising either the UltraTran or IPoptim method, adapt the distance on the
   grounds of the given constraints. Produce updated distance matrix D'.
5. Find $S_i$ and $S_j$ such that D'[$S_i$,$S_j$] is the smallest distance and none of the
   constraints from $C_O$ will be violated if $S_i$ is combined with $S_j$.
6. Utilising Equation (2.8), merge clusters $S_j$ and $S_i$ generating $S_{ij}$=$S_i$∪$S_j$, a novel
   cluster.
7. Set the centroid of the new cluster to the cluster's centre of gravity. Remove
   references to the old clusters and their centroids.
8. Perform constraints updated according to the merged cluster.
9. Until convergence:  decrease K by 1, otherwise go back to Step 3.
```

4.4 Experiments

4.4.1 Experimental Methodology and Evaluation Measures

For evaluating the effectiveness and efficiency of the proposed CWHAC algorithm, the seven datasets from the UCI repository were used. These datasets have different numbers of classes, fields, instances, and dimensions [155] and they relate to different application domains. The specifications of the datasets used are summarized in Appendix-A.

Based our previous research results (see Chapter 3) the algorithm presented here presumes particular distance metric methods and linkage criterion strategy [19] for the different datasets. Four distances metrics were used with seven detests, within proposed algorithm, which are, Mahalanobis distance with Ionosphere dataset, Manhattan distance with BCWD, BCWO, Zoo and Dermatology datasets, Euclidean distance with Iris and Canberra distance with the Wine dataset; even though our previous research showed that these distances were the best for the respective datasets, distance measures other than the above could also be integrated into the algorithm for any dataset. To identify the two closest groups for merging, the Ward-linkage criterion was utilized within the proposed CWHAC algorithm.

The proposed algorithm was implemented using R and C ++ programming languages. In order to achieve improvements in the capability of the C++ programming language in data communication, the Rcpp Armadillo and Rcpp libraries were also utilized. As the algorithms create matrices of high complexity of order O (n^2) and O (n^3) (where n is the number of data points) , experiments were conducted within a Linux Ubuntu 14.04 LTS Intel (R) environment, using a high-performance computer having 20 Cores of Quad 2.80 GHz processors and 64 GB of RAM. The algorithms were executed with varying proportions of constraints for all datasets, ranging from 10 to 60%, which were pre-processed in an effort to remove any unsatisfactory or invalid constraints. In order to reduce the duration for the proposed algorithm to function, the three optimization approaches previously stated were employed across all sets of constraints. The proposed algorithm was applied 10 times for each experimental setting, and the mean of the results are obtained for reporting purposes in this thesis.

The experimental results were evaluated using three measures. Firstly, the run time concerning a proposed algorithm is computed (in seconds) for measuring its efficiency. Secondly, to prove the electiveness of the clustering performance, the F-score is calculated based on matches/mismatches between the clusters obtained by a proposed algorithm in comparison with known class labels. Thirdly, a one-tailed Mann-Whitney test [141] is implemented for demonstrating the significance of the performance improvement of the proposed CWHAC algorithm. The performance of the novel constraint pre-processing methods used within the proposed CWHAC algorithm (with p-value < 0.01) were also evaluated with varying amount of constraints (10%, 30%, 40%, and 60%) and when applied to different datasets. The algorithms were then compared by considering the optimization techniques (such as IPoptim and UltraTran) and the number of constraints, used.

A comparison were made of the performance of the proposed algorithm (CWHAC) when using different parameters (i.e. using two different techniques of fitting an ultra-metric and using different amounts of triple-wise relative constraints) with the results of the previous approach presented in Chapter-3, which does not use the proposed methods [19].

4.5 Result and Discussion

This section presents the experimental results obtained when evaluating the performance of the proposed CWHAC algorithm. The proposed algorithm is evaluated for its effectiveness and efficiency in comparison of existing Ward's Hierarchical Clustering with constraints [19] (CWHAC-before) ("CWHAC-before" refers to the results of before applying the new proposed methods of effectiveness and efficiency. The details discussions of the comparisons are carried out in the subsequent subsections.

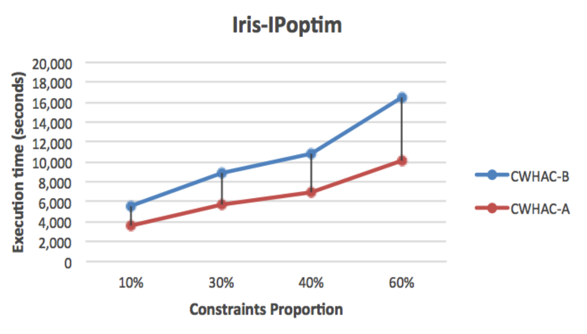
4.5.1 Efficiency Evaluation

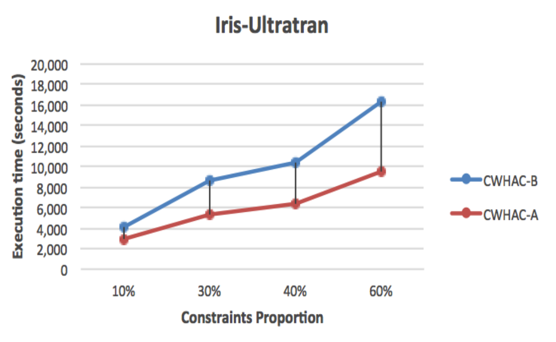
In order to prove the efficiency of the proposed algorithm, the experimental results were computed before and after adopting the principles of speed improvement of generated constraint techniques for the proposed CWHAC, as explained in Section 4.3.1. Evaluation is based on ultra-metric transformation of the dissimilarity matrix. Using the ultra-metric distance matrices, the use of two optimization techniques are evaluated, namely the constrained optimization technique, IPoptim and the transitive dissimilarity, based optimization technique, UltraTran.

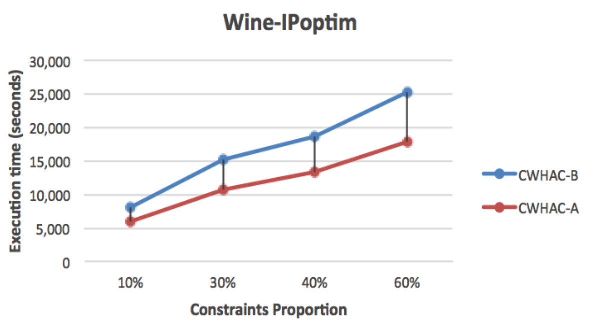
Table 4.2 presents the comparative results of clustering algorithm execution time, both before and after applying proposed methods for reducing the time required for generating constraint to the proposed CWHAC algorithm with using two constrained optimization techniques (IPoptim and UltraTran). And tested on the seven datasets used for experiments, corresponding to varying constraints. The best results are highlighted in bold in Table 4.2. The variation of execution time with increase proportion of constraints for each dataset are illustrated in the graphs of Figure. 4.1.

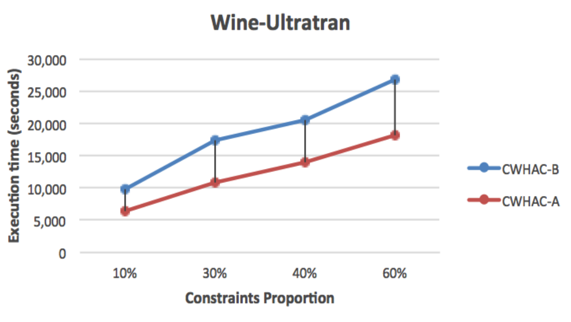
Table 4. 2. Results of using constraint optimizations – CWHAC IPoptim (CWHAC-IP algorithm and CWHAC UltraTran (CWHAC-UT) algorithm execution time (seconds) before and after with varying amount of constraints.

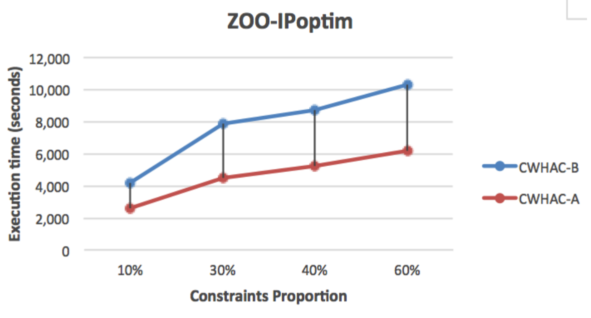
Dataset	Characteristics	Method	Constraints %	10%	30%	40%	60%
Iris	#Instances 150	CWHAC-IP	Before	5583	8863	10735	16408
	#Attributes 4		After	3591	5724	6938	10157
	#Classes3	CWHAC-UT	Before	4039	8571	10322	16259
			After	**2936**	**5377**	**6418**	**9531**
Wine	#Instances 178	CWHAC-IP	Before	7958	15275	18551	25310
	#Attributes 13		After	**5928**	**10592**	**13432**	**17915**
	#Classes4	CWHAC-UT	Before	9805	17365	20586	26831
			After	6418	10841	13958	18247
Zoo	#Instances 101	CWHAC-IP	Before	4178	7829	8752	10249
	#Attributes 17		After	2597	4471	5182	6148
	#Classes7	CWHAC-UT	Before	3864	7249	7915	9583
			After	**2249**	**3847**	**4305**	**5693**
Ionosphere	#Instances 351	CWHAC-IP	Before	131720	182947	211524	250192
	#Attributes 34		After	36801	79593	108397	148725
	# Classes2	CWHAC-UT	Before	102563	152697	182511	228437
			After	**31930**	**59277**	**79413**	**136491**
Dermatology	#Instances 366	CWHAC-IP	Before	147529	198531	226385	264922
	#Attributes 33		After	52469	85274	129146	162507
	# Classes6	CWHAC-UT	Before	110538	164942	186271	229759
			After	**33915**	**61241**	**81309**	**138205**
BCWO	#Instances 683	CWHAC-IP	Before	214793	369148	426281	571496
	#Attributes 10		After	82581	206936	257419	362875
	# Classes2	CWHAC-UT	Before	185278	338511	397295	549789
			After	**80146**	**186137**	**238685**	**350637**
BCWD	#Instances 569	CWHAC-IP	Before	169457	327589	379415	547592
	#Attributes 32		After	**68395**	**175529**	**220846**	**349570**
	# Classes 2	CWHAC-UT	Before	179268	335108	397184	557182
			After	77305	185216	236685	354168

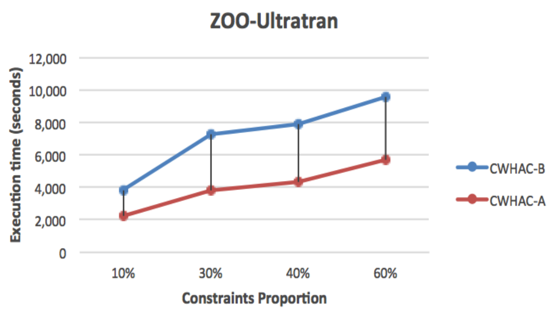

Iris-IPoptim
Execution time (seconds)
20,000
18,000
16,000
14,000
12,000
10,000
8,000
6,000
4,000
2,000
0
10%
30%
40%
60%
Constraints Proportion
CWHAC-B
CWHAC-A

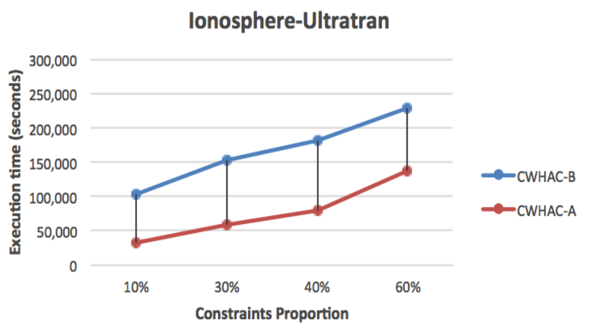

Iris-Ultratran
Execution time (seconds)
20,000
18,000
16,000
14,000
12,000
10,000
8,000
6,000
4,000
2,000
0
10%
30%
40%
60%
Constraints Proportion
CWHAC-B
CWHAC-A

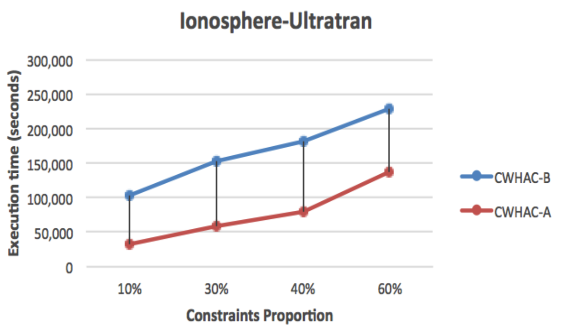

Wine-IPoptim
Execution time (seconds)
30,000
25,000
20,000
15,000
10,000
5,000
0
10%
30%
40%
60%
Constraints Proportion
CWHAC-B
CWHAC-A


Wine-Ultratran
Execution time (seconds)
30,000
25,000
20,000
15,000
10,000
5,000
0
10%
30%
40%
60%
Constraints Proportion
CWHAC-B
CWHAC-A


ZOO-IPoptim
Execution time (seconds)
12,000
10,000
8,000
6,000
4,000
2,000
0
10%
30%
40%
60%
Constraints Proportion
CWHAC-B
CWHAC-A


ZOO-Ultratran
Execution time (seconds)
12,000
10,000
8,000
6,000
4,000
2,000
0
10%
30%
40%
60%
Constraints Proportion
CWHAC-B
CWHAC-A


Ionosphere-Ultratran
Execution time (seconds)
300,000
250,000
200,000
150,000
100,000
50,000
0
10%
30%
40%
60%
Constraints Proportion
CWHAC-B
CWHAC-A


Ionosphere-Ultratran
Execution time (seconds)
300,000
250,000
200,000
150,000
100,000
50,000
0
10%
30%
40%
60%
Constraints Proportion
CWHAC-B
CWHAC-A

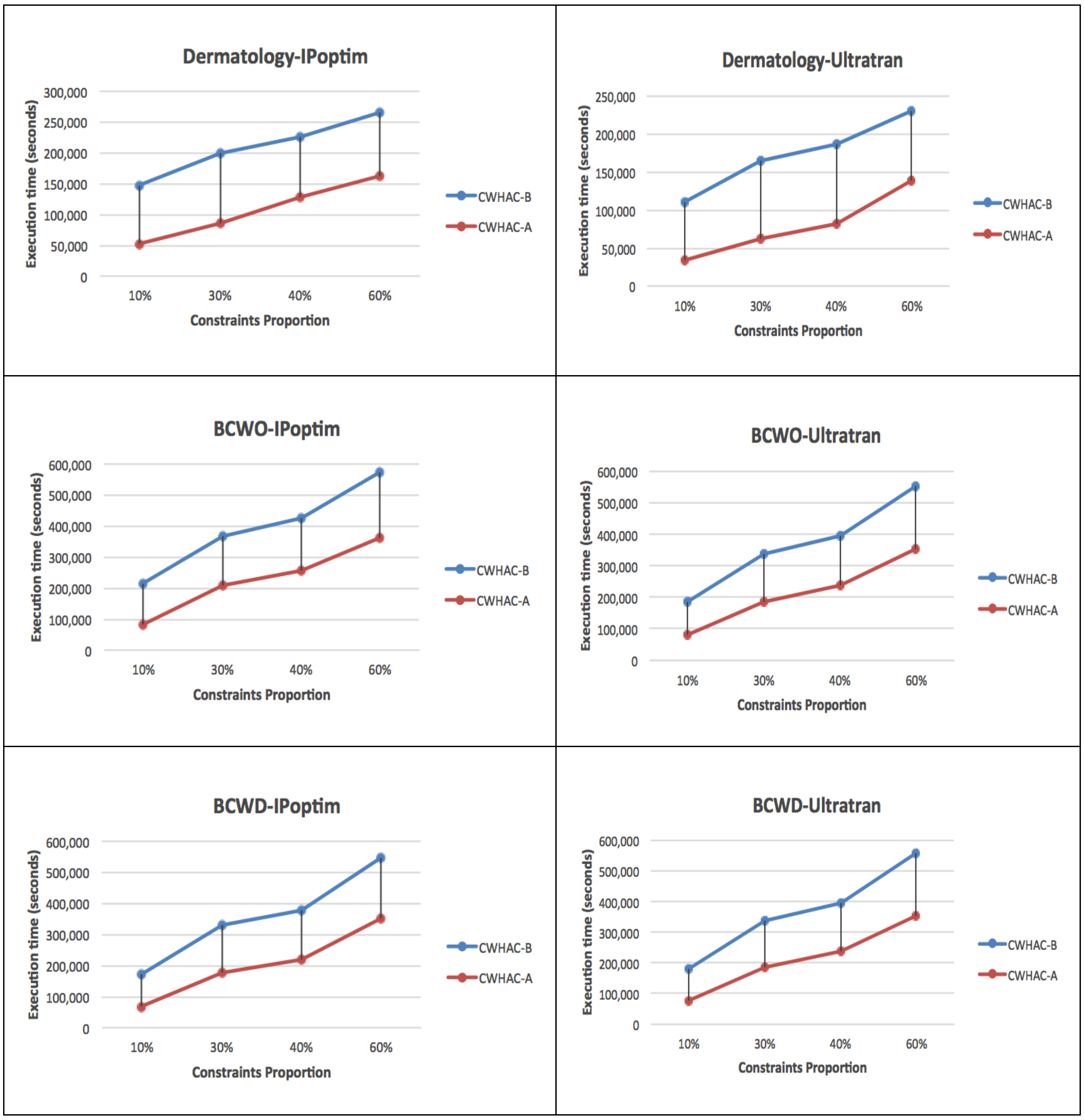


Figure 4. 1. Running time in seconds for CWHAC- IPoptim and CWHAC- UltraTran with varying constraint proportions – before (CWHAC-B) and after (CWHAC-A) using constraint optimization approaches.

From Table 4.2 and Figure 4.1, it can be observed that after applying the proposed methods for minimizing the time required for generating constraints, the execution time in seconds is reduced by a significant amount for all datasets, as indicated in Figure.4.1. The CWHAC-UltraTran method performs better than the CWHAC-IPoptim method for most datasets. Out of the seven datasets used for testing the execution time improvement using CWHAC-UltraTran method is found to be better in five datasets, in particular for

Iris, Zoo, Dermatology, Ionosphere and BCWO where the improvements are statistically significant. It is also showed that the proposed algorithm (CWHAC-A) requires more time for calculating when the data size increases. For example, when compared to Iris with 150 instances, BCWO with 683 instances tends to has higher computational cost.

Table 4.3 presents the percentage change in the time required for execution, measured in seconds, prior to and post the application of the proposed methods (computational optimisation principles of generated constraints) within CWHAC algorithm using IPoptim and UltraTran methods with different proportions of constraints. And the cases that have significant percentage change of the time required for execution are highlighted in bold. From Table 4.3, it can be noted that there is a significant percentage improvement in the time required for execution when the proposed method is applied to CWHAC-IPoptim (CWHAC-IP) algorithm and CWHAC-UltraTran (CWHAC- UT) algorithm. An analysis of the results shows that CWHAC-UT performance is better with various amounts of constraints when compared to CWHAC-IP. Nevertheless, it can be noted that the CWHAC-IP algorithm reaches a substantial change in percentage for the low amount of the constraints (10%) with Iris, Ionosphere, BCWO and BCWD. It is also shown from Table 4.3 that there is more significant percentage change of in results of execution time prior to and post the adopting of the proposed methods with low percentage of constraints for big size datasets (such as Ionosphere, Dermatology, BCWO and BCWD) compared to small size datasets (such as Zoo, Iris and Wine) .

Table 4. 3. Percentage change in results of execution time in seconds (before and after using constraint optimization approaches) for CWHAC- IPoptim (CWHAC- IP) and CWHAC- UltraTran (CWHAC- UT).

Dataset	**Method/Constraints**	**10%**	**30%**	**40%**	**60%**
Zoo	CWHAC- IP	37.84%	42.89%	40.79%	40.01%
	CWHAC- UT	41.8%	46.93%	45.61%	40.59%
Iris	CWHAC- IP	**35.68%**	35.42%	35.37%	38.1%
	CWHAC- UT	27.31%	37.27%	37.82%	**41.38%**
Wine	CWHAC- IP	25.51%	30.66%	27.59%	29.22%
	CWHAC- UT	**34.54%**	**37.57%**	**32.2%**	31.99%
Ionosphere	CWHAC- IP	**72.06%**	56.49%	48.75%	40.56%
	CWHAC- UT	68.87%	61.18%	56.49%	40.25%
Dermatology	CWHAC- IP	64.43%	57.05%	42.95%	38.66%
	CWHAC- UT	**69.32%**	**62.87%**	**56.35%**	39.85%
BCWO	CWHAC- IP	**61.55%**	43.94%	39.61%	36.5%
	CWHAC- UT	56.74%	45.01%	39.92%	36.22%

BCWD	CWHAC- IP	**59.64%**	46.42%	41.79%	36.16%
	CWHAC- UT	56.88%	44.73%	40.41%	36.44%

Percentage change at the execution time:
CWHAC- IP= (CWHAC- IP before - CWHAC- IP after/ CWHAC- IP before) *100
CWHAC- UT= (CWHAC- UT before - CWHAC- UT after/ CWHAC- UT before) *100

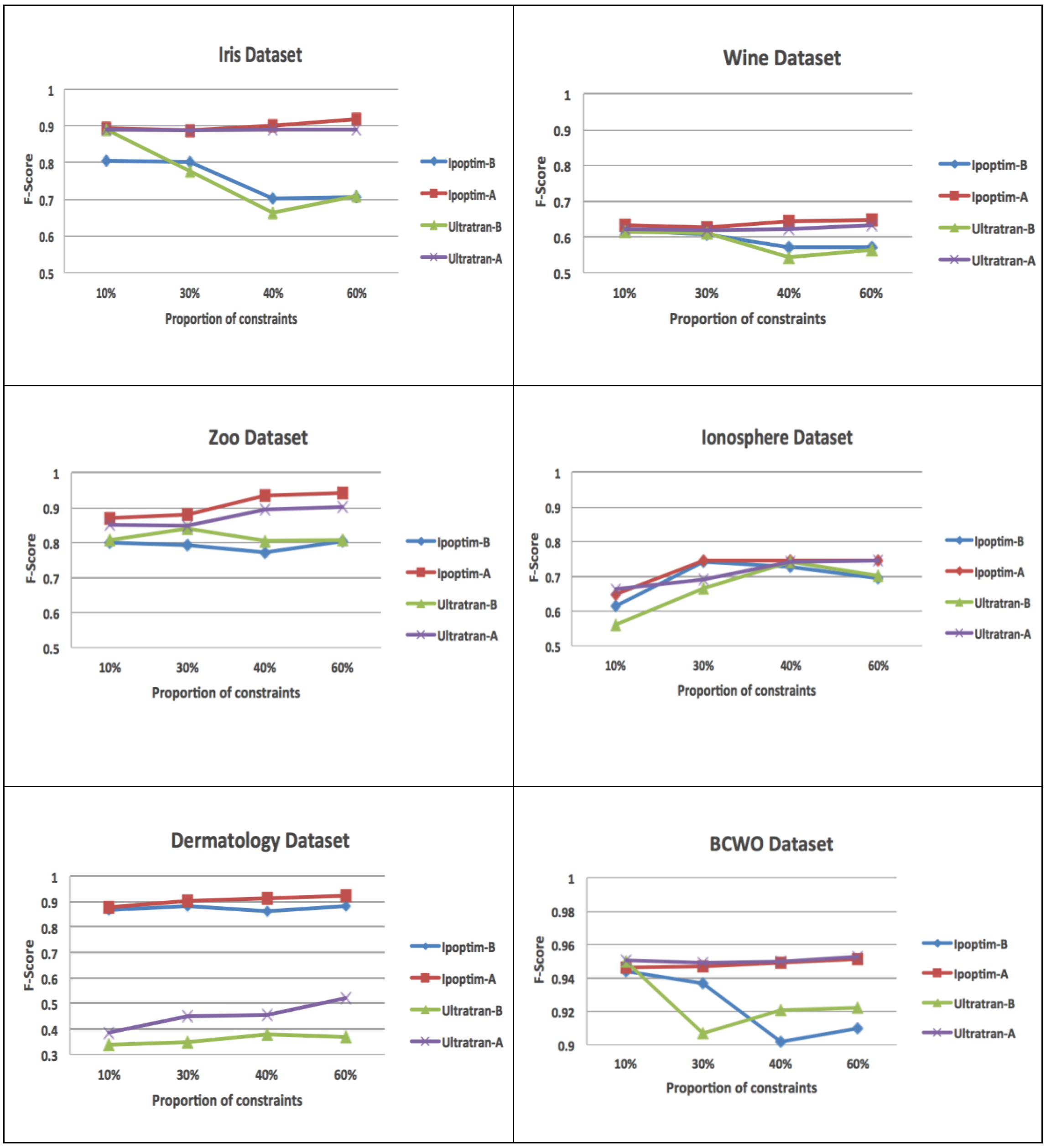
From the comparative results presented in Table 4.2, 4.3 and Figure. 4.1, followings points can be summarized.

- Using the computational optimization principles of generated constraints method, the execution time of CWAAC algorithm reduces substantially. In all datasets, the execution time values of the proposed algorithm (CWHAC-after) are consistently lower compared to CWHAC-before. The percentage of performance improvement is significant, as shown in Table 4.3.
- The proposed algorithm reports a significant reduction in clustering performance on all datasets including large datasets.
- Both methods used, namely IPoptim and UltraTran, demonstrate a good performance in reducing time. It can be observed that for most of the cases, UltraTran outperforms the IPoptim in terms of the efficient clustering performance.
- The average execution time of the clustering algorithms increases with an increasing number of constraints. This is also expected in the case of the proposed algorithm, as it needs additional execution time for tracking forbidden merging of the clusters due to constraint violations.

4.5.2 Effectiveness Evaluation

In order to prove the effectiveness of the proposed algorithm, we computed the experimental results before and after adopting the novel pre-processing methods (see Section 4.3.2), in terms of F-Score. Here, the results 'before' applying the new pre-processing methods of generated constraints refers to the results of existing Ward's Hierarchical Clustering with constraints [19] (CWHAC-before).

The experiments were conducted using UltraTran and IPoptim methods with CWHAC algorithms. The average of 10 executions were carried out for each dataset using IPoptim and UltraTran methods, by varying different proportion of constraints. Figure. 4.2 highlights the comparative results of before and after applying the proposed method of pre-processing and using UltraTran and IPoptim methods on the basis of seven datasets used in this work corresponding to varying constraints in terms of average F-score values. The results of IPoptim- Before (IPoptim-B), UltraTran-Before (UltraTran-B), IPoptim-After (IPoptim-A), and UltraTran-After (UltraTran-A) refer to existing algorithm (CWHAC-before) [19] and the proposed algorithm (CWHAC-after), respectively. The constraints are varied from 10% to 60% as shown in Figure. 4.2.



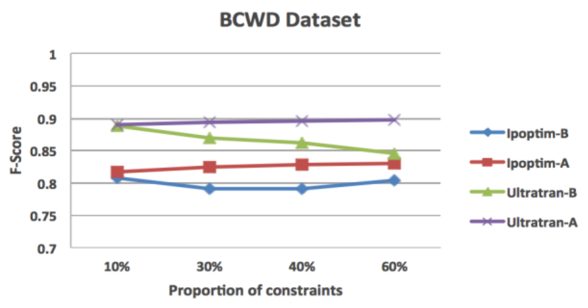


Figure 4. 2. The effectiveness of CWHAC performance with by varying constraint proportions.

It can be clearly noticed from Figure 4.2 that the proposed algorithm (CWHAC) with adopting new pre-processing methods of generated constraints is capable to integrate domain knowledge into the WHC successfully and improves its performance. The algorithm performance improves when an increasing number of constraints. CWHAC algorithm performance with IPoptim-A and UltraTran-A method appears to show significant improvement, compared with CWHAC with IPoptim-B and UltraTran-B, particularly with high amounts of constraints (40% and 60 %) in most of datasets.

From the comparative results presented in Figure. 4.2, the following points can be concluded.

- By adopting new pre-processing methods of generated constraints, the proposed algorithm becomes more effective with increasing number of constraints, when compared with results of existing Ward's Hierarchical Clustering algorithm with constraints (without adopting the new pre-processing methods) [19] . It is not monotonically increasing with the number of constraints. As shown in Figure. 4.2, the clustering performance without the new pre-processing methods is degraded for most of the datasets with high amounts of constraints compared to low amount of constraints.
- Both the methods used, namely IPoptim and UltraTran, show a high impact on the effectiveness of the proposed algorithm with different constraints. It can be observed that most of the cases, IPoptim outperforms the UltraTran in terms of clustering performance.

To further investigate the effectiveness of the CWHAC algorithm, we conducted a Mann-Whitney test [141] to detect statistical significance in the differences of the algorithm's performance considering p < 0.01. The Mann-Whitney test is applied to the average of the F-score values for each algorithm to compare the CWHAC algorithm performance before and after improving the impact of the constraints. A comparative analysis is performed on the basis of the two constrained optimization methods, namely, IPoptim and UltraTran. The tested combinations include CWHAC-IPoptim-Before, CWHAC-IPoptim-After, CWHAC-UltraTran-Before, CWHAC-UltraTran-After, with varying amounts of constraints in terms of F-score values. For each constraint, a test is conducted which is a test of the CWHAC-IP-B versus CWHAC-IP-A, and CWHAC-Ul-A versus CWHAC-Ul-A using the average of the F-score values for each dataset. The outcome of the Mann-Whitney test is presented in Table 4.4 and all significant improvements are highlighted. The significant improvement of p-value is p < 0.01.

Table 4.4 shows that there is a significant improvement with Iris, particularly CWHAC algorithm with IPoptim from using 10% to 60% constraints. The CWHAC algorithm performance with both IPoptim and UltraTran significantly improves results in most of the datasets, when using high amounts of constraints (40% and 60%). In addition, it was noticed that the CWHAC algorithm with both IPoptim and UltraTran methods appears to show insignificant improvement when using 10% to 60% constraints for the BCWD dataset.

Table 4. 4. Results of P-values obtained using Mann-Whitney test[141](test the IPoptim-B versus IPoptim-A, and UltraTran-B versus UltraTran-A within CWHAC algorithms).

Dataset	**Methods**	**Constraints Proportion**			
		10%	**30%**	**40%**	**60%**
Iris	CWHAC- IPoptim –B Vs. CWHAC- IPoptim –A	**0.00018**	**0.00029**	**0.00009**	**0.00009**
	CWHAC- UltraTran-B Vs. CWHAC- UltraTran-A	0.42465	0.03754	**0.00058**	**0.0057**
Wine	CWHAC- IPoptim –B Vs. CWHAC- IPoptim –A	0.23576	0.04457	**0.00009**	**0.00018**
	CWHAC- UltraTran-B Vs. CWHAC- UltraTran-A	0.0268	0.23576	**0.00009**	**0.00009**
Zoo	CWHAC- IPoptim –B Vs. CWHAC- IPoptim –A	0.01287	**0.00453**	**0.0005**	**0.00289**
	CWHAC- UltraTran-B Vs. CWHAC- UltraTran-A	**0.00368**	0.48405	**0.00034**	**0.00126**
Ionosphere	CWHAC- IPoptim –B Vs. CWHAC- IPoptim –A	**0.00866**	0.31207	0.05592	**0.00022**
	CWHAC- UltraTran-B Vs. CWHAC- UltraTran-A	0.09853	0.11314	0.23576	**0.00009**
Dermatology	CWHAC- IPoptim –B Vs. CWHAC- IPoptim –A	0.04457	0.03515	**0.00009**	0.01578

	CWHAC- UltraTran-B Vs. CWHAC- UltraTran-A	**0.00695**	**0.00111**	0.01578	**0.00029**
BCWO	CWHAC- IPoptim –B Vs. CWHAC- IPoptim –A	0.33724	0.15386	**0.00009**	**0.00009**
	CWHAC- UltraTran-B Vs. CWHAC- UltraTran-A	0.40905	0.00866	**0.00866**	0.04457
BCWD	CWHAC- IPoptim –B Vs. CWHAC- IPoptim –A	0.39743	0.05155	0.05155	0.07078
	CWHAC- UltraTran-B Vs. CWHAC- UltraTran-A	0.12100	0.02068	0.01578	0.06057

Significant improvement $p < 0.01$

4.6 Conclusion

This chapter has presented the CWHAC which is a new take on Ward's hierarchical agglomerative clustering method that is founded on the triple-wise relative constraints. The CWHAC efficiency and effectiveness have been improved after adopting new methods. Such methods ensure that the proposed algorithm can successfully incorporate domain knowledge-based triple-wise relative constraint in CWHAC algorithm successfully while enhancing its performance by decreasing the amount of violated and redundant constraints as well as the enhancing computational complexity of CWHAC algorithm through three optimization principles for decreasing the time required to generate constraints. The proposed algorithm was experimented different parameters' variations, including the number of constraints, the distance function used and constrained optimization methods which are IPoptim and UltraTran. The results of the algorithm findings were examined concerning clustering quality accuracy as well as runtime performance. The experiment results showed that IPoptim as well as UltraTran methods can effectively perform in CWHAC to reduce the time complexity and enhance the clustering quality. Moreover, while the UltraTran method can outperform the IPoptim method regarding clustering efficiency, the IPoptim method can outperform the UltraTran method regarding clustering effectiveness. The important improvements were validated using the Mann-Whitney test [141] before and after the new pre-processing method regarding generated constraint techniques was implemented. It should be noted that Although there have been improvements in the efficiency of the CWHAC algorithm, a significant amount of time is required with a large number of constraints, especially for large datasets. Hence, the next chapter will conduct studies using alternative methodologies including a hybrid approach for examining the accelerating agglomerative clustering problem to provide an initial partition that has a sufficient amount of cluster sample. This process can ensure that the efficiency of cluster merging that begins with the initial partition instead of from a trivial partition that only includes individual clusters.

CHAPTER 5

Constrained Ward's Hierarchical Clustering Method Using the Two Initial Cluster Setting Strategies:Agglomerative and Intelligent K-means

5.1 Introduction

The preceding chapters, chapter 3 and 4, presented a detailed account of the growth evident in the domain of ssHC with knowledge-based constraints and proposed a novel approach for the advancement of the state-of-the-art based on triple-wise relative constraints and their efficient selection, in Chapter-4. This chapter presents an enhanced framework of Constraint Ward's Hierarchical Clustering (CWHC) algorithm employing a novel strategy for the initialization of the clusters. In Chapter-4 a strategy that combines the conventional Agglomerative method with the Constrained Ward's Hierarchical Clustering algorithm, named as CWHAC was designed and formulated and its performance efficiency and effectiveness is analyzed in detail. It is noted that the Agglomerative approach itself provides a basis for the initialization of the clusters. In this chapter a further innovative hybrid semi-supervised clustering method employing triple-wise relative constraints that gives competitive performance and less computation time is proposed. It is termed as the Constrained Ward's Hierarchical Clustering founded on intelligent K-Means (ik-means) (CWHC-IKM) that is an amalgamation of the benefits of Ward's Hierarchical clustering under knowledge-based constraints along with the effective cluster initialization that can be enabled by ik-means. The hierarchical clustering quality is enhanced using the existing triple-wise relative constraints as proposed in Chapter-4 and the cluster initialization using the ik-means algorithm is shown to drastically reduce the convergence time further. It gives way to the initial partitioning of Ward's hierarchical clustering, which lets the cluster merging to initiate from this partition rather than from a trivial partition made of only a singleton. It is proved that this approach saves time and makes the overall clustering algorithm more efficient.

This chapter is sectioned into six parts. Apart from this section with provided an overview of the contribution to be made in this chapter, section 5.2 presents the literature review, research background and motivation, followed by section 5.3 that details the framework of the suggested approach based on ik-means and a revisit to the agglomerative approach presented in chapter-4. Section, 5.4, provides an account of the experiments conducted. The penultimate section, 5.5, compares and analyses the outcomes of the empirical results of the execution of both CWHAC and CWHC-IKM algorithms, followed by a brief conclusion in section 5.6.

5.2 Background and Motivation

The clustering can be conducted whether using a PC method or hierarchical clustering method (as discussed in chapter 2). However, when comparing these methods, each method may have a positive feature that does not exist in the other that may be handled by hybrid approaches between the two. The PC algorithms consume less computation time. A majority of them function in linear time, in other words with time complexity O(n); however, hierarchical algorithms give a comparatively better clustering quality at higher efficiency [173], [174]. Hasan and Duan [175] propose that hierarchical clustering is comparatively more detailed and informative than PC such as k-means as it determines the entire hierarchy of clusters. The data is made easily comprehensible with the use of dendrogram to nest and present the clusters. Nevertheless, despite the merits of agglomerative hierarchical structure, it has its share of limitations, for example the overall quality declines with the addition of data [174]. One more disadvantage is that that computations are much more time consuming and require more memory than non-hierarchical options [13], [14]. In the case of the traditional HAC, it is imperative to calculate all pairwise distances with quadratic complexity. Some distance matrix functions are expensive especially for high dimensional data. Accurate measurement of the pairwise distance of data points in close vicinity is very important due to the nature of HAC algorithms to combine nearest data points or clusters into tree nodes, nevertheless, calculating distances between remote points is a futile exercise and should be avoided [176]. Thus, it is wise to partition the dataset in order to bypass a full distance matrix computation [177].

It can be deduced from above discussion that a single clustering approach (partitional or hierarchical) is insufficient for effective clustering data [174]. A promising direction to speed up the procedure of clustering, while making the data clusters more meaningful is to hybridize both paritional and hierarchical clustering approaches [172]. A majority of the hybrid methods are designed for unsupervised hierarchical clustering. Several studies used the unsupervised PC algorithm, k-means clustering algorithm, to divide the input data into m sub-clusters and then applied unsupervised hierarchical clustering to create a hierarchical structure based on *m* sub-clusters [14], [174], [178], [179]. De Amorim, Makarenkov and Mirkin [21] propose a new algorithm to conduct efficient unsupervised hierarchical clustering with the help of intelligent k-means (ik-means) initialisation [18], [40] and is termed as the A-ward hierarchical clustering algorithm that allows the cluster merging to begin from the initial partition. It was also proved that A-ward algorithm can drastically reduce the time needed by a conventional Ward hierarchical clustering algorithm while ensuring no adverse changes to cluster recover ability [21]. A-Ward algorithm is able to partition the data into a number of clusters without handle prior knowledge. However, it may result in retrieving unrelated information for the user as it is an unsupervised algorithm that does not employ knowledge-based constraints for clustering.

A few limited numbers of studies have also been conducted on ssHC approaches [13], [171]. A hierarchical clustering algorithm based upon the K-means with ML and CL constraints (HCAKC) with better clustering quality and lower computational complexity was proposed by Hang et al. [171]. The ideal number of clusters in HCAKC algorithm is ascertained by calculating the average Improved Silhouette (IS) of the dataset to detect the ideal number of clusters, followed by determining the initial clusters by running K-means. Subsequently then the final clusters are obtained by using the hierarchical agglomerative algorithm with pair-wise ML and CL constraints on the initial clusters. The constraint violation is monitored and handled by modifying the similarity metric using a penalty factor. Nevertheless, the HCAKC algorithm may generate varying clustering solutions depending on the initial clusters owing to the complicated prediction of right centroid through K-means clustering.

The HAC technique with constraints developed by Tamura, Obara, and Miyamoto [13] has the capability to handle huge quantities of data through a two-stage clustering method.

A one-pass COP k-means++ algorithm is used in the first stage (was developed using two methods: k-means++ and COP k-means) followed by the hierarchical agglomerative algorithm in the second stage to ensure that the output of the merge process is in the form of a dendrogram. With regard to pair-wise constraints at both stages, the ML and CL constraints are used specifically in the first stage. Despite the fact that CL constraints are addressed in the first stage their presence can be seen in the second stage and is therefore handled by a penalty term. A random selection of object pairs in CL was conducted with the objects being selected from different clusters of the data set. The number in CL varies from 0 to 50 for artificial data and 0 to 500 for Shuttle data. The preliminary tests on data sets revealed that ML is not useful when compared to CL. The proposed algorithm was evaluated using two data sets, one having real data and the other having artificial data [180]. It was then ascertained that the proposed method depends heavily on the initial values, to enhance the clustering quality. The one-pass k-means++ is comparatively more effective than one-pass k-means in the first stage and ML is comparatively inefficient with regard to CL. Tamura, Obara, and Miyamoto [13] study that the pair-wise constraints with the help of hybrid methods (mix of hierarchical and partitional clustering). However, the other researchers proposed that the pair-wise constraints are not conducive to hierarchical clustering [5], [11].

Even though optimization methodologies were proposed in Chapter-4 for reducing the time consumed in generating constraints within CWHAC algorithm, it still is a time-consuming process to reach convergence for large datasets. This is because it starts from a trivial partition made of only singletons. The agglomerative hierarchical approach, which is of high computational complexity, takes a considerable time to converge. It starts each cluster as a singleton, and then recursively combines the two clusters with the least distance, to result in a singular large cluster [44]. Hence, the aim of this research is to develop an alternative approach to effectively speed up the hierarchical clustering so that it becomes capable of handling large data using semi-supervision or constraints. To the best of our knowledge, this is the first attempt to that propose the use of knowledge-based triple-wise relative constraints with the hybrid approach of hierarchical clustering described above. The new hybrid algorithm proposed in this study is termed as the CWH-IKM algorithm. It is capable of learning from ik-means and triple-wise relative approach.

Along with a new algorithm, this study aims to present a semi-supervised clustering framework capable of learning from either of two approaches which are agglomerative (Chapter-4) and ik-means approaches. The performance of the two approaches are compared in terms of time and clustering quality (using f-score). Even though it has not been attempted before, we would present a detailed comparison of the performance of the CWHC algorithm with regard to both methods for cluster initialisation, i.e. based on agglomerative (CWHAC) and based on ik-means (CWHC-IKM). In particular, their effectiveness and efficiency of clustering performance with different constraints are compared and analysed.

5.3 Proposed Approach - Constraint Ward's Hierarchical Clustering (CWHC) Algorithm

The main aim of extending and developing the Ward hierarchical clustering algorithm as CWHC algorithm is to find a better clustering solution using triple-wise relative constrains rules. The proposed algorithm depends on the ultra-metric transformation of the dissimilarity matrix, which uses triple-wise relative constraints to generate a new metric for data similarity. The proposed algorithm can be classed as a 'hybrid' approach between search-based and distance-based methods. When the constraints are presented, the IPoptim or UltraTran methods aim to update the dissimilarity metrics to cater to the transitive property of the distance and the present constraints. At the same time, the conventional Ward clustering algorithm function combines clusters using Equation 2.8. Within search-based methods, the merging clusters with minimum distance can be skipped if constraints are violated. The violated constraints would be filtered by using pre-processing methods for constraints mentioned in the previous chapter (see chapter 4 in Section 4.3.2).

The CWHC algorithm takes a set of datasets S, the number of clusters *k* and lists triple-wise relative constraints as input. The details of algorithm used for generating triple-wise relative constraints were provided in Chapter 4 (algorithm 4.1). The CWHC algorithm can produce a variety of clustering solutions that are generated using different types of initial setup methods. The proposed algorithm applies two types of such methods as described below and shown in Figure 5.1.

✓ The CWHAC method is based on a traditional HAC algorithm proposed in our preceding chapter (see chapter 4 (algorithm 4.2)) and published by [20]. It involves three phases:

- In the first step, it starts to compute the initial distance between $X_{[i]}$ and $X_{[j]}$ which is represented by d_{ij} element of calculated matrix D and followed by presenting every object in data as a cluster.
- In the second step, the constraints are given, the IPoptim or UltraTran methods seek to update dissimilarity metrics D to fulfill Constrained ultra-metric distance matrix D'. And then the smallest distance between clusters is merged using the Ward's method (Equation 2.8 in Chapter 2) as per the constrained ultra-metric distance and their centroids are set up based on the new cluster generated to the cluster's centre of gravity. And remove references to the old clusters and their centroids.
- In the third step, the constraints are updated as per the merged cluster. All the steps (2 and 3) are repeated till convergence: reduce number of clusters K by 1.

✓ The CWHC-IKM is developed as a new hybrid semi-supervised clustering algorithm in this chapter. It combines the advantages of the partitioning and hierarchical clustering algorithms with the existing constraints. The CWHC-IKM algorithm passes through two stages in the clustering process as described below.

- The first stage starts to compute the initial distance between X[i] and X[j] which is represented by d_{ij} element of calculated matrix D, and followed by applying ik-means method for generating an initial partition K* (k* is greater than the true number or the desired number of clusters) with their centroids CK*. The triple-wise relative constraints are given, the application of IPoptim or UltraTran methods on the initial partition to update dissimilarity metrics D to fulfill constrained ultra-metric distance matrix D'.
- In the second stage, Ward's method (Equation 2.8 in Chapter 2) is employed to the initial partition (after updating distance based on constraints) for the merging process. The initial partition has two levels, which are used to obtain the hierarchical cluster dendrogram. The Ward's method is applied at the first level to group the objects within each cluster into an individual detailed tree T1, T2…Ti. At the second level, each Ti

is treated as a cluster and merged by Ward's method into a one tree T during the search for the minimum distance between clusters and the closest clusters are merged with updating their centroids to set the centroid of the new cluster to the cluster's centre of gravity. This step is repeated until convergence (the number of clusters K= 1).

The ik-means algorithm is a smart initialization method aimed to automatically detecting the actual number of clusters and the primary centroids for K-Means. However, the cluster objects that depict the different concepts are not fully represented by the distance function and are assigned to the cluster of the nearest center, which may occur close to cluster borders. As shown in Figure5.2, both elements are close to each other but belong to the different clusters. Hence, the distance of the initial partition (obtained by ik-means) is updated according to the given triple-wise relative constraints prior to merging. This is followed by employing Ward's method on initial partition, which forms $K*$, used to cluster the k* centroids into a tree-like dendrogram structure along with the objects within each of the clusters into one detailed trees (k). This minimizes the time required by the merging clusters to generate a tree-like dendrogram from $K*$ until a single cluster of size of N data (K) is found.

The effectiveness and efficiency of CWHAC method was improved by prosing to use novel techniques for improving the execution time of generated constraints and addressing the problem of satisfiable constraint selection as discussed in Chapter 4. Therefore, these techniques are also adopted into CWHC-IKM method.

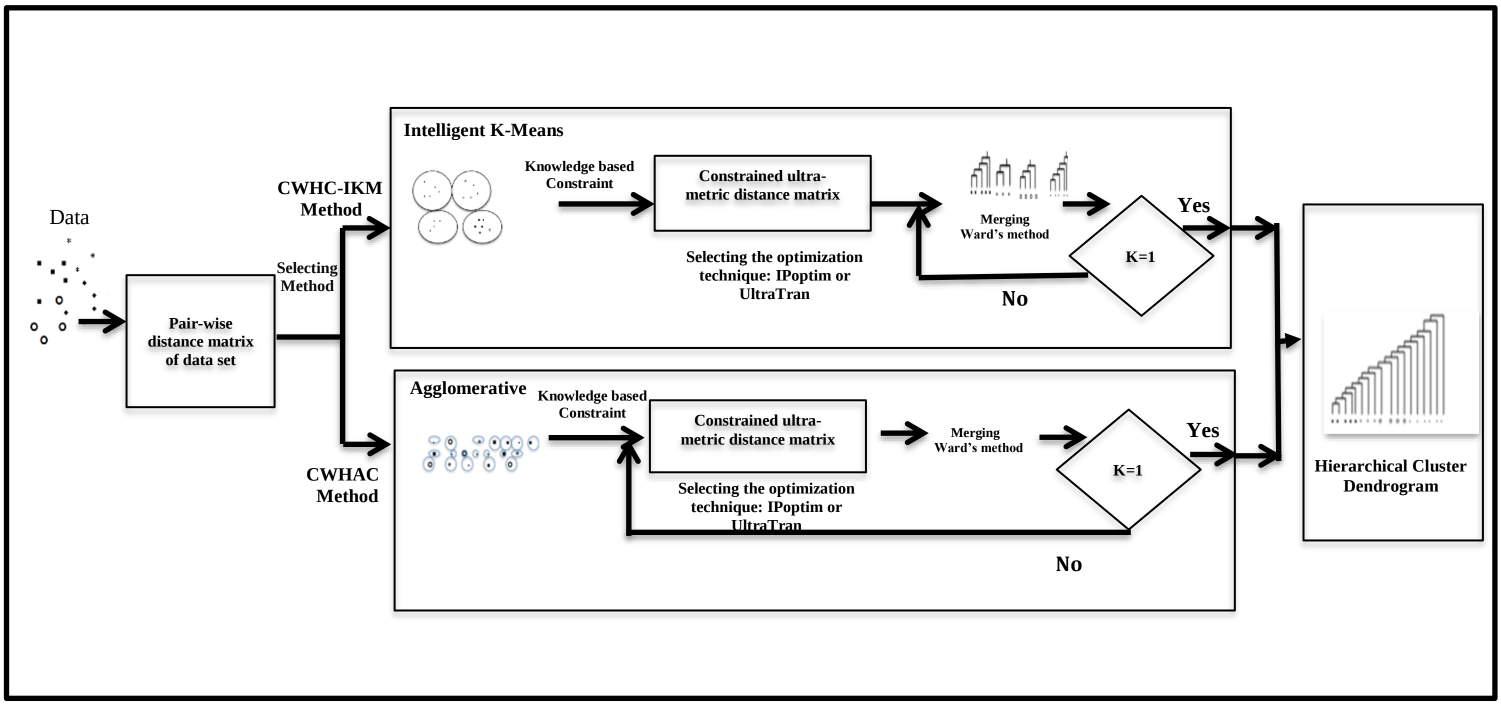

Data
Pair-wise distance matrix of data set
Selecting Method
CWHC-IKM Method
CWHAC Method
Intelligent K-Means
Knowledge based Constraint
Constrained ultra-metric distance matrix
Selecting the optimization technique: IPoptim or UltraTran
Merging Ward's method
K=1
Yes
No
Agglomerative
Knowledge based Constraint
Constrained ultra-metric distance matrix
Selecting the optimization technique: IPoptim or UltraTran
Merging Ward's method
K=1
Yes
No
Hierarchical Cluster Dendrogram

Figure 5. 1. Framework of Constrained Ward's Hierarchical Clustering algorithm (CWHC) based on different Initialization methods. Initial objects within proposed algorithm are clustered by either hierarchical agglomerative method or intelligent k-means method.

5.4 Experiment Setup

This study uses two constrained optimization methods, namely IPoptim and UltraTran to analyze the effects of CWHC algorithm with different approaches of the initial clustering setting (CWHAC and CWHC-IKM) and how it is affected by the various degrees of constraints 10%, 30%, 40%, and 60 %. All the constraints were pre-processed to eliminate all unsatisfactory and invalid constraints and the three optimization approaches have been used across the constraints to reduce the execution time of the proposed algorithm, the details of which have been discussed in Chapter 4 (see Chapter 4 in Section 4.3.1 and Section4.3.2).

A set of three distance metrics have been adopted within the proposed algorithm (based on results in Chapter 3 and published in Aljohani et al., [19] with the experiments conducted on eight popular UCI datasets [155] (the characteristics of the datasets are provided in Appendix-A), namely: Euclidean distance with Iris dataset; Mahalanobis distance with Ionosphere dataset; and Manhattan distance with BCWO, BCWD, Dermatology, Mammographic and Banknote Authentication datasets. The maximum, minimum, average of the standard deviation of 10 runs were computed for each dataset with CWHAC and CWHC-IKM methods using IPoptim and UltraTran methods, by varying different proportions (percentages) of constraint.

The proposed algorithm (CWHC-IKM) has been compared with CWHAC of Chapter-4 for its effectiveness and efficiency under different parameters, such as the constrained optimization methods, the number of constraints, and the computational complexity until convergence. Furthermore, CWHAC and CWHC-IKM have been compared against two unsupervised HC: unsupervised Ward's Hierarchical Agglomerative Clustering (WHAC) and unsupervised Ward's Hierarchical Agglomerative Clustering based ik-means (A-Ward) [21] .

The proposed algorithm has been implemented in R and C++ programming. The Rcpp Armadillo and Rcpp library were used to communicate a C++ programming method with R. The experiments were carried out in Linux Ubuntu 14.04 LTS Intel (R) environment, using a high-performance machine having 20 Cores Quad 2.80 GHz processor and 64 GB of RAM.

5.4.1 Performance Measures

The performance effectiveness and efficiency of the algorithms has been evaluated by analyzing the experimental results through using different measures. Both internal and external indices were used to evaluate the effectiveness of the algorithms. The external measure, the F-score is the primary criteria used in this study to analyze the level of agreement achieved between the output of the proposed clustering methods and the correct class labels of the dataset available as ground-truth. It is used as an evaluation metric of the performance of the proposed CWHAC and CWHC-IKM methods over all the nodes in the dendrogram. The empirical results of F-score value used for the evaluation of proposed methods are the minimum, maximum and average values of the respective standard deviations obtained.

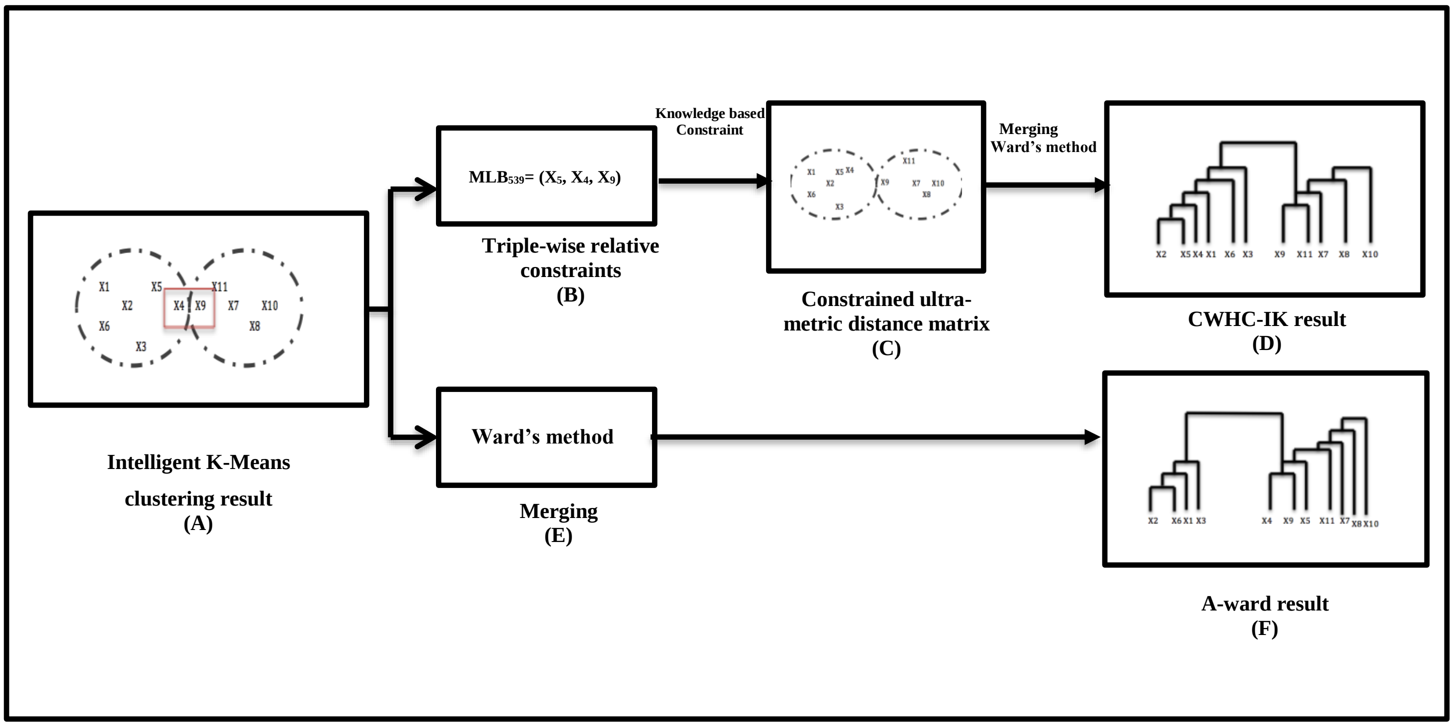

Intelligent K-Means clustering result (A)
X1
X5
X11
X2
X4
X9
X7
X10
X6
X8
X3
MLB539= (X5, X4, X9)
Triple-wise relative constraints (B)
Knowledge based Constraint
Constrained ultra-metric distance matrix (C)
Merging Ward's method
CWHC-IK result (D)
X2 X5 X4 X1 X6 X3 X9 X11 X7 X8 X10
Ward's method
Merging (E)
A-ward result (F)
X2 X6 X1 X3 X4 X9 X5 X11 X7 X8 X10

Figure 5. 2. An illustrative example of comparing the Ward's Hierarchical Clustering Algorithm Based on ik-means with constraints (CWHC-IKM) and without constraints A-Ward). The initial clusters by ik-means are shown in (A). The triple-wise relative constraints are given in (B). (C) Shows the constrained ultra-metric distance matrix and (D) is the result of the corresponding hierarchical clustering with constraints (CWHC-IK). By combining both (A) and (E), the hierarchical clustering based on ik-means without constraints (A-Ward) is illustrated.

The main reason for applying internal evaluation measures in this study is due to the information intrinsic to the data that studies the goodness of a clustering structure without external information. In the absence of the required partition, the quality of a partition can be calculated by measuring how thoroughly each instance is associated with the cluster and how well-separated a cluster is from other clusters [181], [182]. Calinski-Harabasz (CH) index[134] is an internal measure (see Chapter 2 in Section 2.6, Equation 2.27) which is applied to the best F-score results with five of the eight datasets (namely: Iris, Zoo, Dermatology, BCWO, and BCWD) to further evaluate cluster quality in terms of compactness and well-separateness. The maximum value for CH indicates a suitable partition for the data set.

Furthermore, the efficiencies of CWHAC and CWHC-IKM methods have been evaluated by determining the run-time (in seconds). Four of the eight datasets (have a varying size of instances, a number of dimensions and number of classes) have been experimented with different amount of constraints (ranging from 10% to 60%) to investigate the time needed to run the proposed methods (CWHAC and CWHC-IKM). Nevertheless, the run-time of algorithms with only 60% of constraints has been experimented with for the rest of the datasets. Finally, the non-parametric Mann-Whitney test [141] is used for determining whether the improvement is statistically significant for both algorithms, considering (with p-value < 0.01). The test has been applied to compare CWHC-IKM against an agglomerative approach (CWHAC).

The resulting clusters for CWHC algorithm with varying strategies, namely CWHAC and CWHC-IKM are presented in the form of dendrograms. The datasets which demonstrated the best three clustering results, out of the eight datasets, were selected (Iris, Zoo, and Dermatology) and used in a further experimental study with a varying number of dimensions and number of classes. Furthermore, the dendrogram with the best clustering results using 60 % of constraints is presented and visualized.

5.5 Results and Discussion

This section presents the experimental results of the performance evaluation of the two CWHC algorithms, CWHAC and CWHC-IKM and discusses the effectiveness and efficiency of each method. Additionally, the performance of unsupervised Ward's Hierarchical Clustering (WHC) and A-Ward [21] have been compared with the proposed methods with constraints, with amount of constraints ranging from 10 % to 60%. Details of the analysis and explanations are provided in the following sub-sections.

5.5.1 Effectiveness Measurement and Validation

A. External Clustering Results-Based F-score

Figure 5.3 illustrates the experimental results obtained when using CWHAC and CWHC-IKM with Ipoptim (CWHAC-IP, CWHC-IKM-IP) and UltraTran (CWHAC-UT, CWHC-IKM-UT) methods on the eight datasets. F-score (minimum, maximum and average of Standard Deviation) measure is being used with varying percentages of constraints. The

F-score obtained by the unsupervised WHC and A-Ward methods are also indicated in comparison to the results of the above four methods.

It can be observed from Figure 5.3 that average values of CWHC-IKM method performs better clustering results than CWHAC in five out of eight datasets (Iris, Zoo, Dermatology, BCWO, and Banknote authentication). This is significantly demonstrated with complex dataset such as Dermatology which contains six groups with 33 features. Figure 3.5.c shows that using UltraTran within CWHC-IKM produces a significant improvement in clustering process compared to using within CWHAC. The CWHC-IKM method can partition data into initial subgroups that help to be more organized for data that contains large number of classes. This initial partition makes more control to assign the points in their correct group by using constrained optimization method (UltraTran).

In a few cases, the minimum CWHAC results are shown to unexpectedly exceed those of CWHAC-IKM. Simultaneously, the corresponding maximum CWHC-IKM results are found higher than maximum CWHAC findings, demonstrating the superior performance of CWHC-IKM. This is illustrated in Figure 5.3 wherein diverse results can be observed for different datasets with different percentage of constraints, such as IPoptim with CWHC-IKM and CWHAC is superior for Zoo with 40% and 60% of constraints, Dermatology dataset with 10 % of constraints, and the Banknote authentication with 30%, 40% and 60% of constraints. However, the findings of UltraTran with CWHC-IKM and the CWHAC demonstrate better performance with the following: the BCWD dataset with 10%, 30%, 40% and 60% of constraints, BWCO with 30% constraints and Mamography database with 10%, 30%, 40% and 60% of constraints, and Iris with 10%.

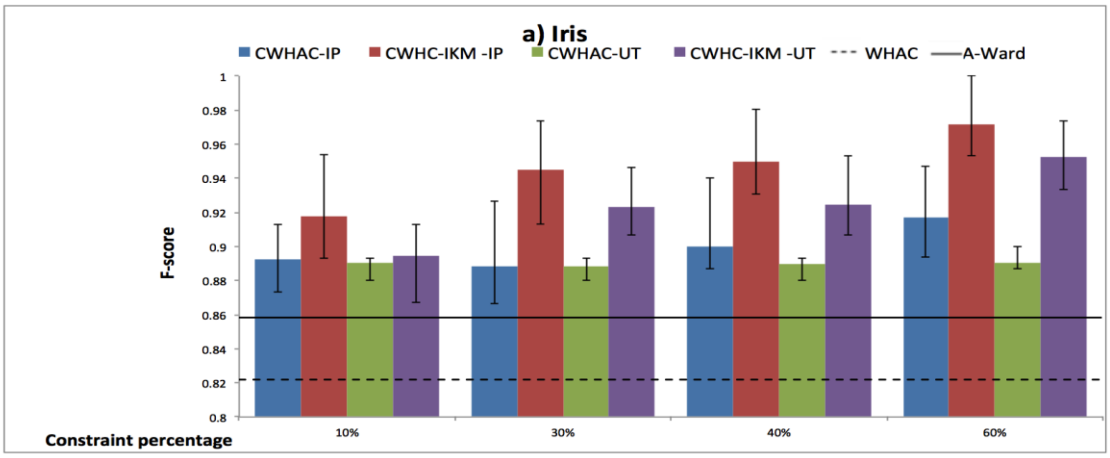
As discussed above, CWHC-IKM significantly outperforms CWHAC in the majority of datasets, however, it fails to demonstrate this trend in all the constraints settings in certain datasets. This is evident in CWHAC combined using both constrained optimization methods (UltraTran and IPoptim) that produces a better result for BCWO at 10 % with UltraTran and with IPoptim at 10% and 30% respectively; for BCWD with IPoptim 10% and for Banknote authentication with 40%. Overall, the performance of CWHC-IKM is poor compared with CWHAC in a few cases, when using low percentages of constraints with UltraTran and using both low and high percentages with IPoptim.

When making a comparison between two constrained optimization methods within proposed methods, the IPoptim method with either CWHAC or CWHC-IKM performs better than the UltraTran technique for most of the datasets used. Conversely, the opposite observation is found on BCWD and Mammographic such that the UltraTran method performs demonstrably better in terms of cluster quality with both methods (CWHAC and CWHC-IKM). Furthermore, the two methods of constrained optimization (IPoptim and UltraTran) demonstrate identical responses in relation to the two proposed methods. In other words, when the best result of one of the proposed approaches with IPoptim or UltraTran outputs that can be seen the same behavior in the same constrained optimization method with another approach in seven of eight datasets. However, on Ionosphere, it is observed that CWHAC with IPoptim demonstrates superior performance in comparison to Ultra Tan. CWHC-IKM with IPoptim demonstrates poor performance when compared with UltraTran.

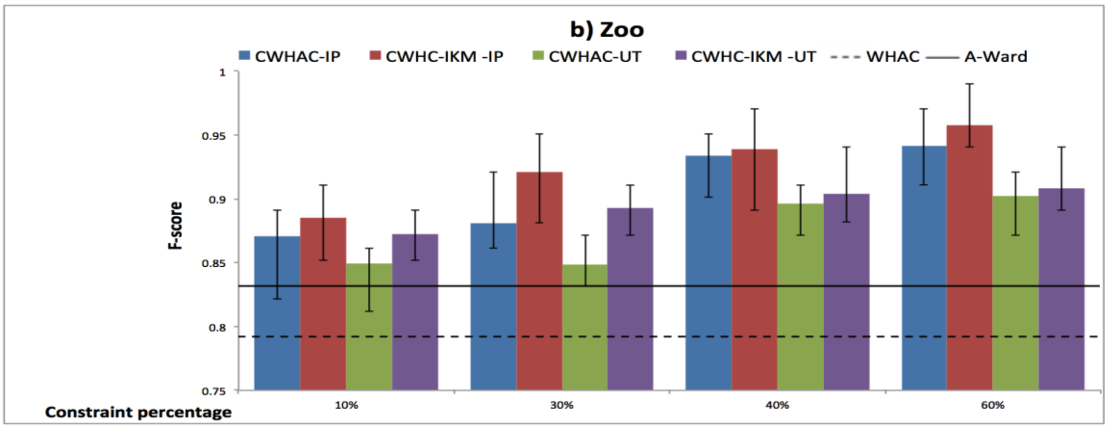
To analyze the stability in the performance of the CWHC algorithm using the CWHC-IKM and CWHAC methods, we executed the algorithm 10 times and averaged the results calculating and considering the standard deviation of the results. Figure 5.3, reports results of standard deviation that are shown in the table below the graphs. It can be observed that CWHAC demonstrates greater stability than CWHC-IKM for all the datasets. Moreover, whilst both CWHAC and CWHC-IKM possess more stability when combined with the UltraTran method than with IPoptim method in most of the datasets. However, the dermatology dataset produces entirely contrary results for different amount of constraints. Likewise, the mammographic dataset produces similarly opposing results for most of the constraint percentages, as does the BCWO, with 10% and 40% constraint percentages.

Figure 5.3 also reveals that the introduction of constraints in the proposed semi-supervised hierarchical clustering algorithms (CWHAC and CWHC-IKM) results in the outperformance of clustering results. They are obtainable from hierarchical clustering with constraints (A-ward and WHAC). This is shown even when having the number of constraints limited to 10%. It is interesting to note that it is not necessary to base the superior results of A-ward algorithm over WHC algorithm to expect this same behavior to be true for the proposed semi-supervised algorithms. For example, WHC on BCWO demonstrates superior performance to A-ward, while CWHC-IKM outperforms CWHAC.

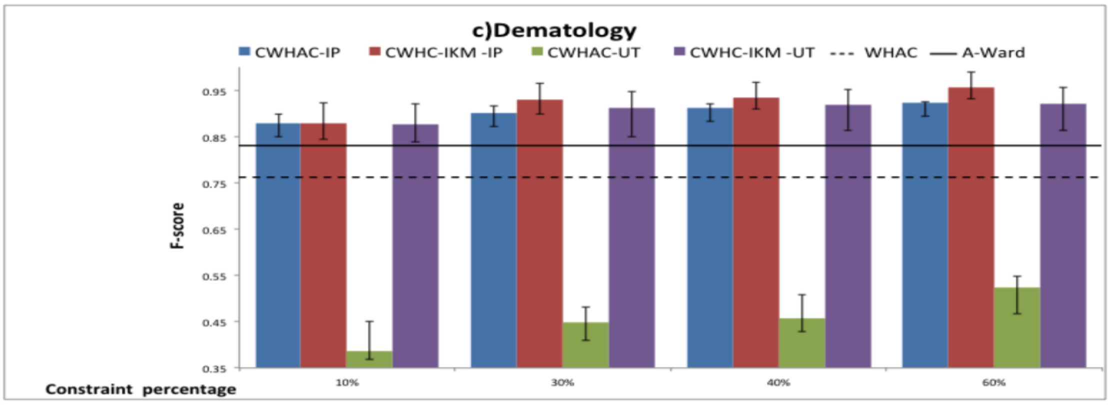
Conversely, on BCWD, A-ward performs better than WHC, whereas CWHC-IKM outperforms CWHAC. Finally, the proposed methods for addressing the problem of satisfiable constraints, which were discussed in the previous chapter (see Chapter 4), highlighting their positive impact on the performance of the CWHAC method are also proved to be successful when used within CWHC-IKM. In particular, the solution has a significant positive impact when an increasing amount of constraints are used within CWHC-IKM.



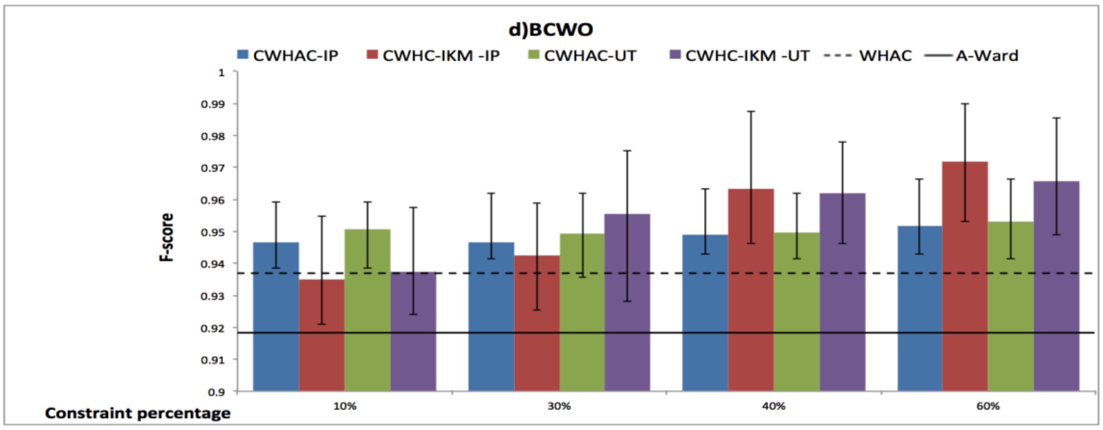
		10%	30%	40%	60%
🟦	CWHAC-IP (SD)	0.0187	0.0182	0.0169	0.0166
🟥	CWHC-IKM -IP (SD)	0.0198	0.0175	0.0191	0.0186
🟩	CWHAC-UT (SD)	0.0053	0.0049	0.0044	0.0061
🟪	CWHC-IKM -UT (SD)	0.0192	0.0168	0.0154	0.0159



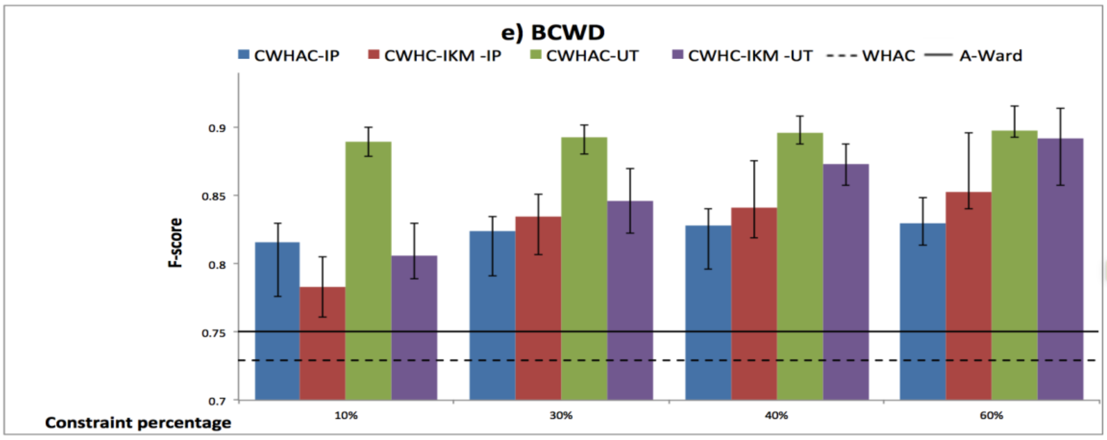
		10%	30%	40%	60%
🟦	CWHAC-IP (SD)	0.0185	0.0172	0.0177	0.0179
🟥	CWHC-IKM -IP (SD)	0.0281	0.0259	0.0238	0.0217
🟩	CWHAC-UT (SD)	0.0152	0.0154	0.0149	0.0136
🟪	CWHC-IKM -UT (SD)	0.019	0.0187	0.0175	0.0183



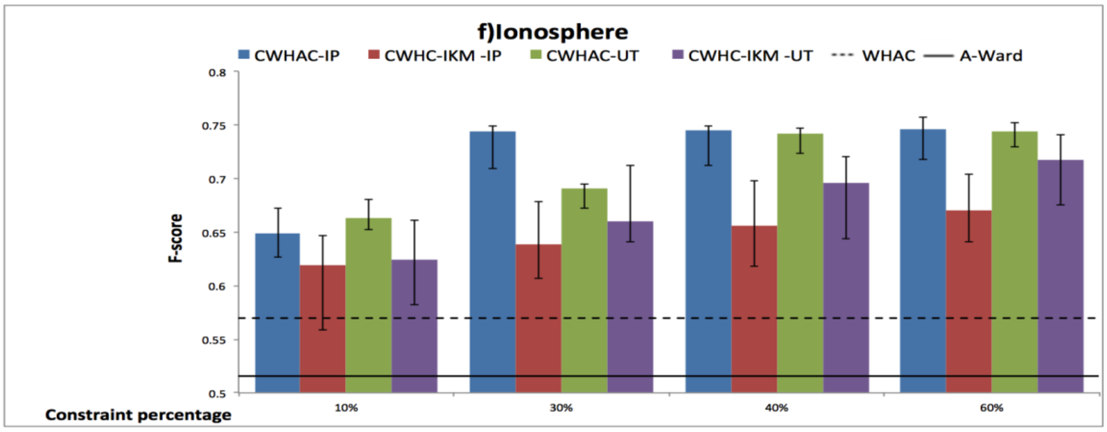
🟦 CWHAC-IP (SD)	0.0016	0.0167	0.00151	0.0098
🟥 CWHC-IKM -IP (SD)	0.0291	0.0282	0.0256	0.0239
🟩 CWHAC-UT (SD)	0.0255	0.0289	0.0292	0.0285
🟪 CWHC-IKM -UT (SD)	0.0379	0.0332	0.0326	0.0319



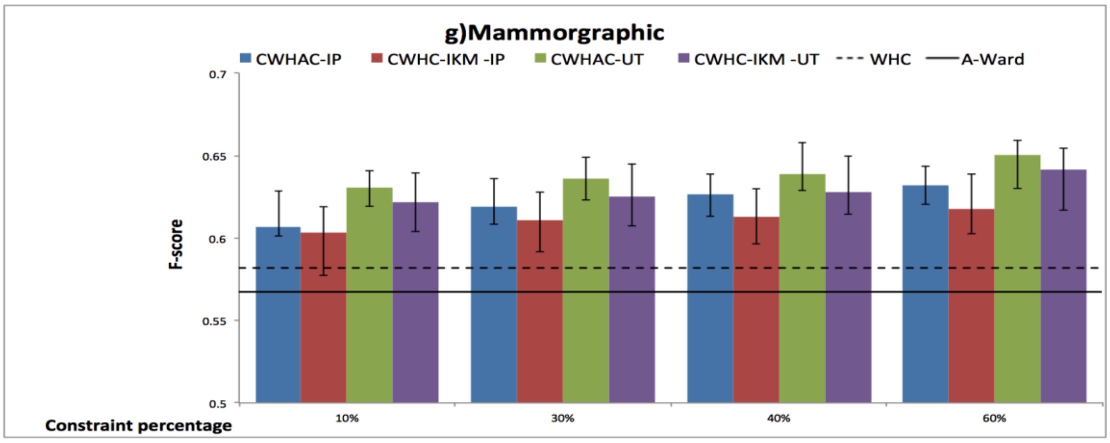
🟦 CWHAC-IP (SD)	0.0097	0.0071	0.0067	0.0084
🟥 CWHC-IKM -IP (SD)	0.0162	0.0167	0.0165	0.0168
🟩 CWHAC-UT (SD)	0.0082	0.0059	0.0071	0.0067
🟪 CWHC-IKM -UT (SD)	0.0165	0.0162	0.0157	0.0159



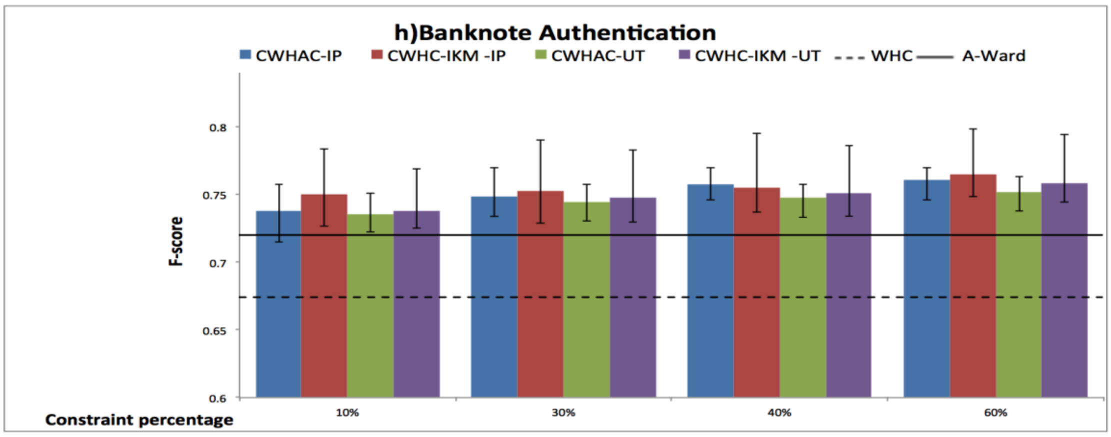
🟦 CWHAC-IP (SD)	0.0182	0.0165	0.0167	0.0162
🟥 CWHC-IKM -IP (SD)	0.0219	0.0196	0.0191	0.0194
🟩 CWHAC-UT (SD)	0.0105	0.0088	0.0082	0.0085
🟪 CWHC-IKM -UT (SD)	0.0189	0.0174	0.0158	0.0153



	10%	30%	40%	60%
🟦 CWHAC-IP (SD)	0.0177	0.0118	0.011	0.0146
🟥 CWHC-IKM -IP (SD)	0.0305	0.0286	0.0281	0.0278
🟩 CWHAC-UT (SD)	0.0124	0.0098	0.0083	0.0095
🟪 CWHC-IKM -UT (SD)	0.0257	0.0265	0.0269	0.0273



	10%	30%	40%	60%
🟦 CWHAC-IP (SD)	0.11	0.0117	0.0108	0.0103
🟥 CWHC-IKM -IP (SD)	0.0193	0.0168	0.0166	0.0172
🟩 CWHAC-UT (SD)	0.0113	0.0115	0.0124	0.0129
🟪 CWHC-IKM -UT (SD)	0.0179	0.0182	0.0168	0.0165



	10%	30%	40%	60%
🟦 CWHAC-IP (SD)	0.0165	0.0123	0.0117	0.0113
🟥 CWHC-IKM -IP (SD)	0.0229	0.0215	0.0208	0.0196
🟩 CWHAC-UT (SD)	0.0136	0.0117	0.0087	0.0108
🟪 CWHC-IKM -UT (SD)	0.0198	0.019	0.0187	0.0183

Figure 5. 3.Results of the clustering quality in terms of the external measure (F-score) corresponding to different constraint percentages.

B. Internal Clustering Results - Based on Calinski-Harabasz Index

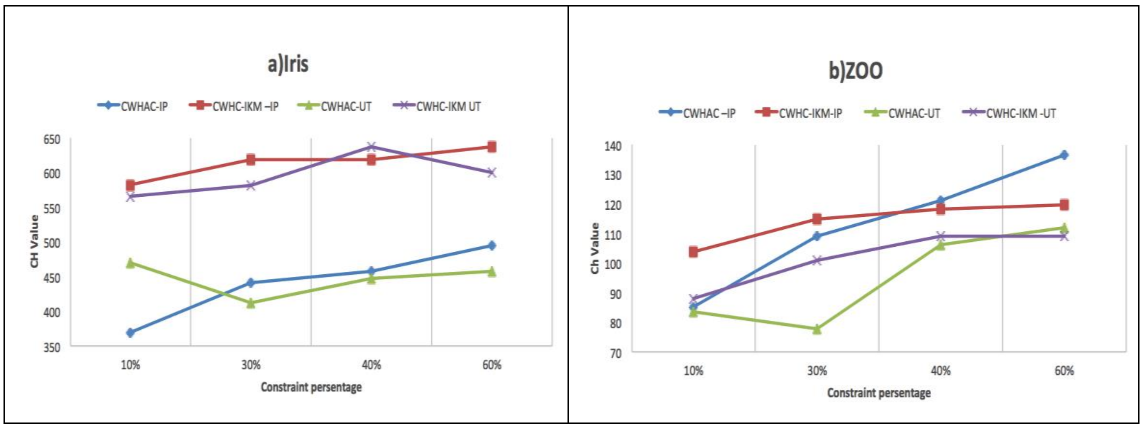
The Calinski-Harabasz (CH) index originating from the corresponding maximum values of the F-score for different datasets was employed to evaluate the internal clustering results of the proposed algorithms. We run the proposed algorithms 10 times and measure the performance in terms of F-score. We selected the best/ most accurate cluster result of 10 runs and then evaluated that cluster by internal measures (CH) to measure the separations and compactness. Figure 5.4 presents the comparative CH values which depict the clustering performance for the proposed methods (CWHAC and CWHC-IKM) when used for clustering five datasets. A high value of CH-index indicates good clustering performance which are well-separated and compact clusters. The following conclusions can be drawn from an examination of Figure. 5.4.

- The CH result of CWHC-IKM with IPoptim produces more compact and well-separated clusters compared with CH results of other methods in majority of the datasets.
- Interestingly, the CWHC-IKM with IPoptim method for the Zoo dataset demonstrates improved performance in class-label agreement but reduces performance in terms of cluster quality when used with high percentages of constraints (40% and 60%). The best demonstrable performance in relation to these constraints is demonstrated with the IPoptim within CWHAC method.

There is both a match and consistency between the results of using internal measures and external measures when using the IPoptim method. With an increase in the proportion of constraints, there is an increase in values of F-score and CH for CWHC-IKM and CWHAC using IPoptim method in all datasets. However, this observation is reflected when using UltraTran, as there is an increase in class-label agreement although the cluster quality deteriorates drastically when increasing the proportion of constraints in some cases. For example, the UltraTran produces more compact and well-separated clusters with CWHAC (for Zoo and BCWD datasets when using 10% constraints as compared to using 30% constraints and for Iris at 10% constraints when compared with other amounts of constraints. Further, this occurs with CWHC-IKM as well (for Iris and 40% constraints as compared with 60% constraints, and the same behavior with dermatology, but with 10% constraints compared with 30 % constraints).

The CH measure outperforms other internal measures, as per Riyaz and Rashid[183], Arbelaitz et al. [132], and Milligan and Cooper [184]. However, the results in Figure 5.4 and those from research by Riyaz and Rashid[183] suggest that the internal measure (CH) is less precise in relation to determining true label clusters in comparison to the external index (F-score). The CH depends on a clustering structure which fails to account for the presence of objects in the groups. Thus, the CH index can demonstrate high-performance levels with objects which are in proximity to each other in the group and the individual group is far from another group. Whilst the presence of these objects in this group cannot be deemed correct. Alternatively, a contrary result might arise wherein objects are present in the correct group, yet they are not compact enough and have separated clusters which result in poor CH results.

As per Riyaz and Rashid [183], whilst the external indices perform better than the internal ones, they require additional external information that is rarely available for real datasets. Conversely, in the current study, validating experimental findings with external measures is comparatively easy, since the class-label for each dataset in the study is already known. To conclude, the direction and outcomes from the current study are dependent upon the external measure.



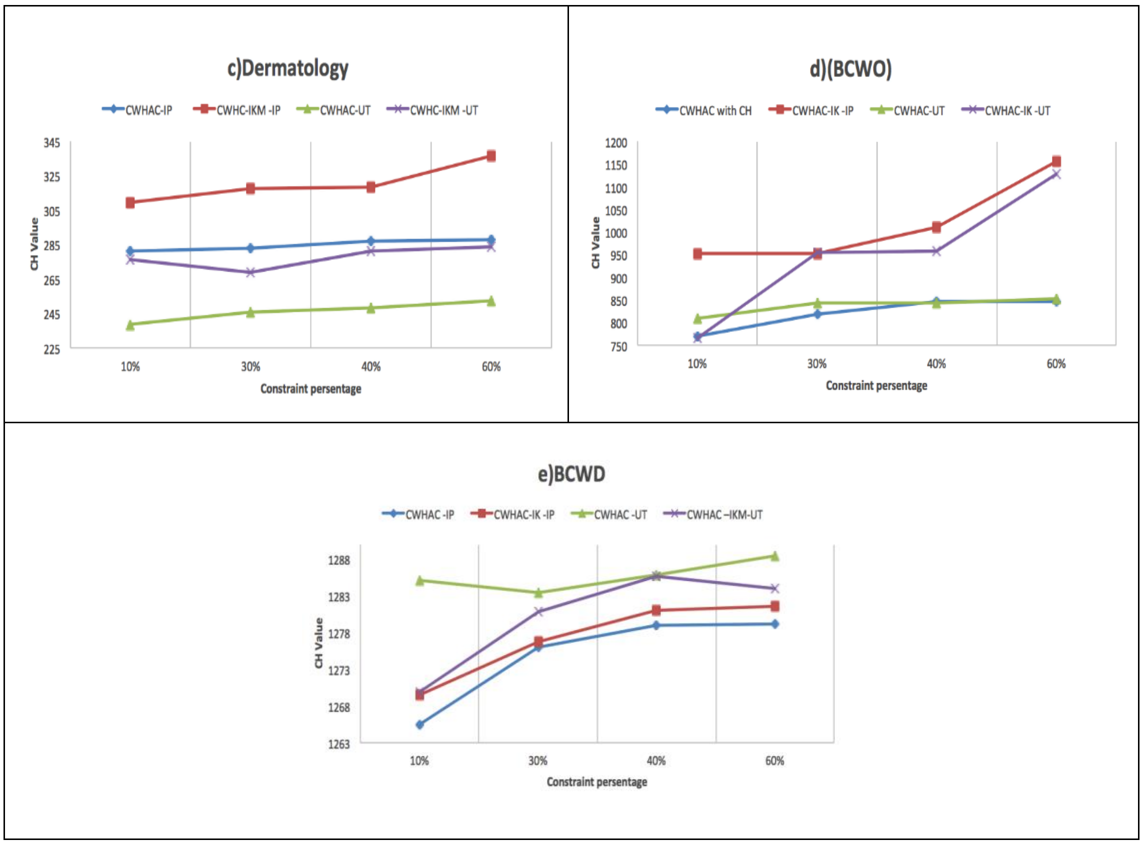


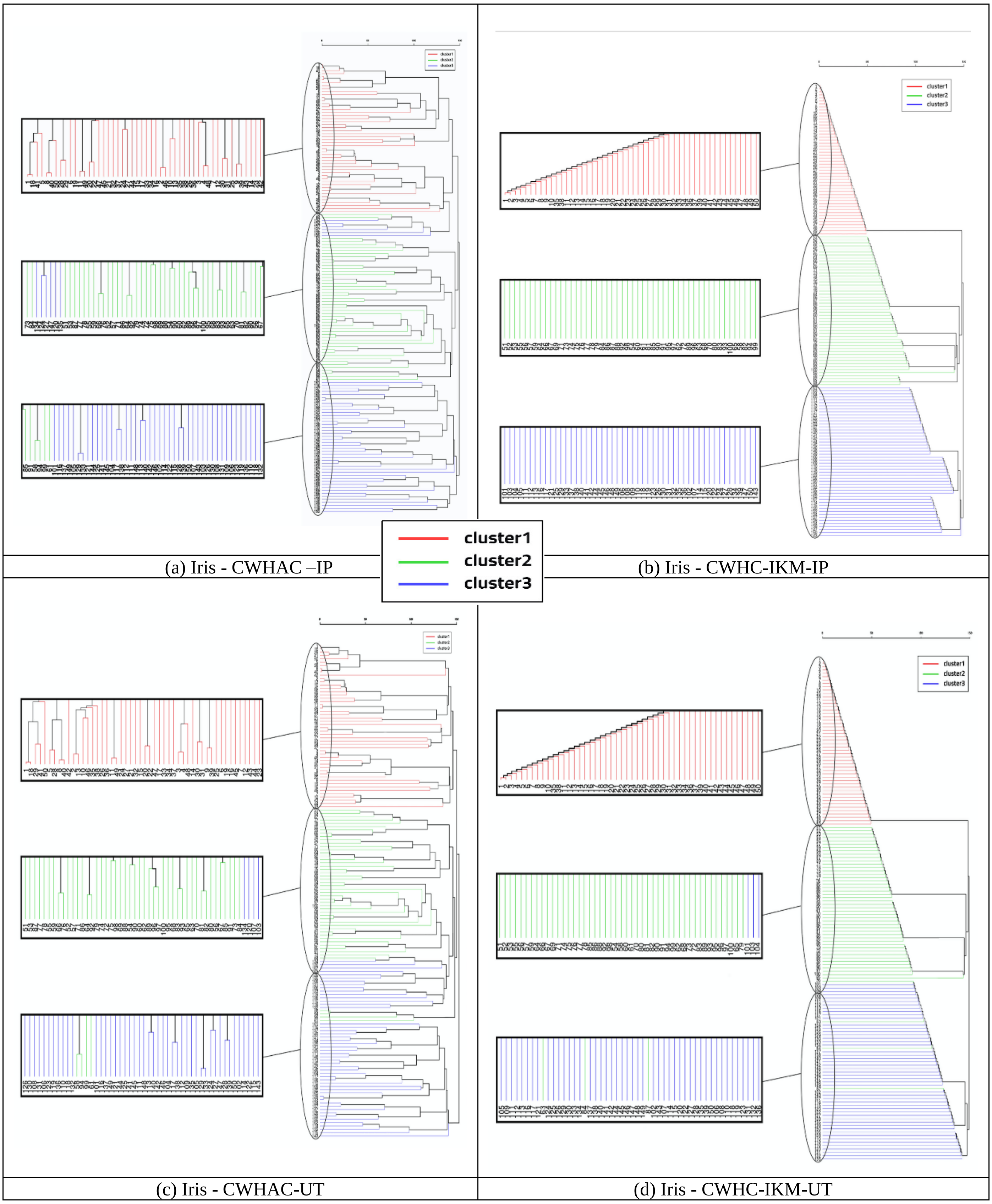
Figure 5. 4. Results of the clustering quality in terms of the dependence of Internal measure (CH) to different proportions of constraints.

B. Visualizing the Clustering Results for CWHAC and CWHC-IKM

To extend the analysis, the dendrogram hierarchical clustering has been employed to visualize cluster results obtained by UltraTran and IPoptim within CWHAC and CWHC-IKM using three datasets, namely, Iris, Zoo, and Dermatology, as per Figure 5.5 (see Appendix -B for the full-size visualization of all dendrograms). These datasets are chosen in accordance with features present in various dimensions and due to generating the most effective clustering results relative to other datasets. Different colours are employed to denote clusters in the dendrogram in order to assist readers in the identification process.

It is visually apparent in Figure 5.5 that the CWHC-IKM exposes the different structure of dendrogram clustering compared to the structure displayed through the CWHAC method. This is caused by differences in the initializing of the clustering between the methods involved. Specifically, the CWHC-IKM algorithm establishes dendrogram clustering groups, whilst CWHAC constructs individual samples. It is also observed that

CWHC-IKM-IP demonstrates better performance for the datasets of Zoo, Iris and Dermatology in comparison to the CWHAC method. For example, in Iris with CWH C-IKM-IP (Figure 5.5, a-d) , all CWHC-IKM-IP dendrogram points are correctly allocated to the appropriate classes. Yet misclustering occurs with CWHAC-IKM-UT, CWHAC-UT, and CWHAC-UT, which consequently lowers levels of accuracy in the findings. Certain points in class (2) and (3) are wrongly assigned to class (1) in CWHAC-UT, whilst in CWHAC-IP certain points from class (3) have become merged with points in class (3) whereas the contrary occurred for CWHC-IKM-UT, where some points from class (2) are incorrectly placed with class (3). In addition, Figure 5.5 contains data pertaining to dendrogram clustering for other datasets (Dermatology and Zoo).

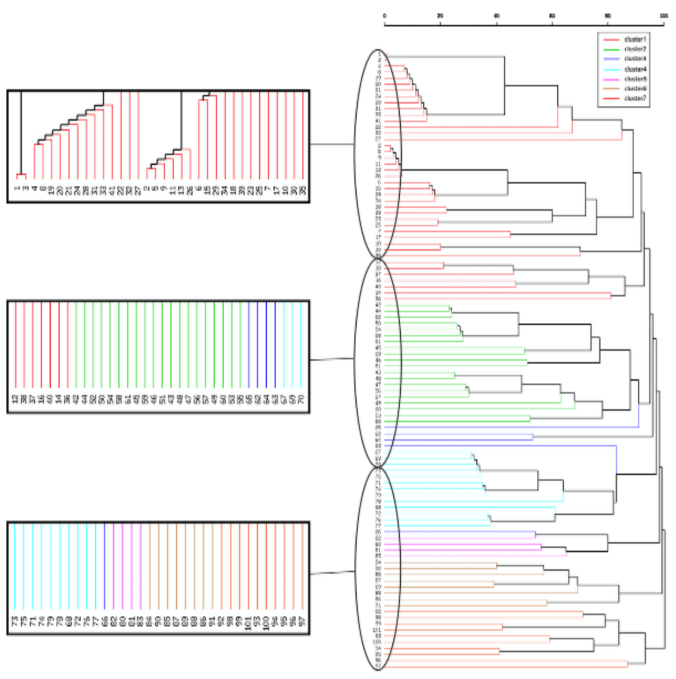


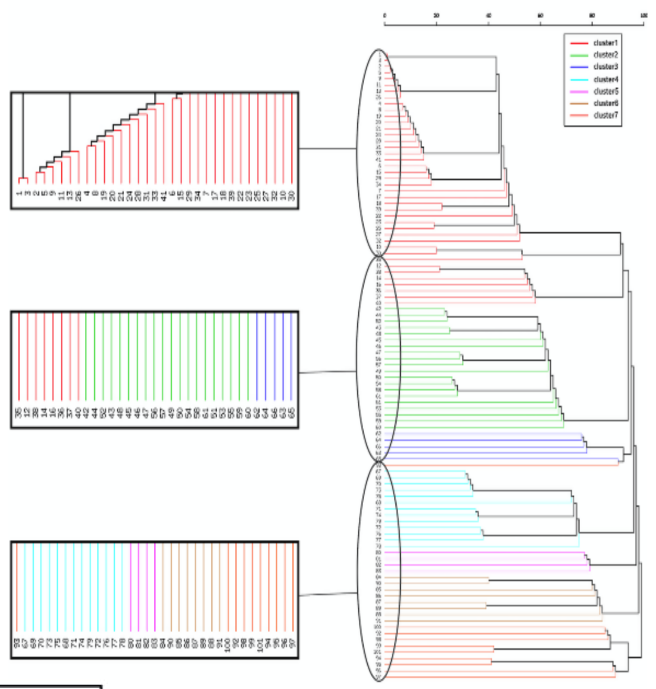
(a) Iris - CWHAC –IP

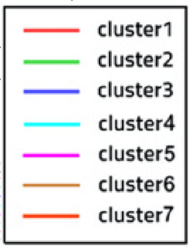
(b) Iris - CWHC-IKM-IP

(c) Iris - CWHAC-UT

(d) Iris - CWHC-IKM-UT

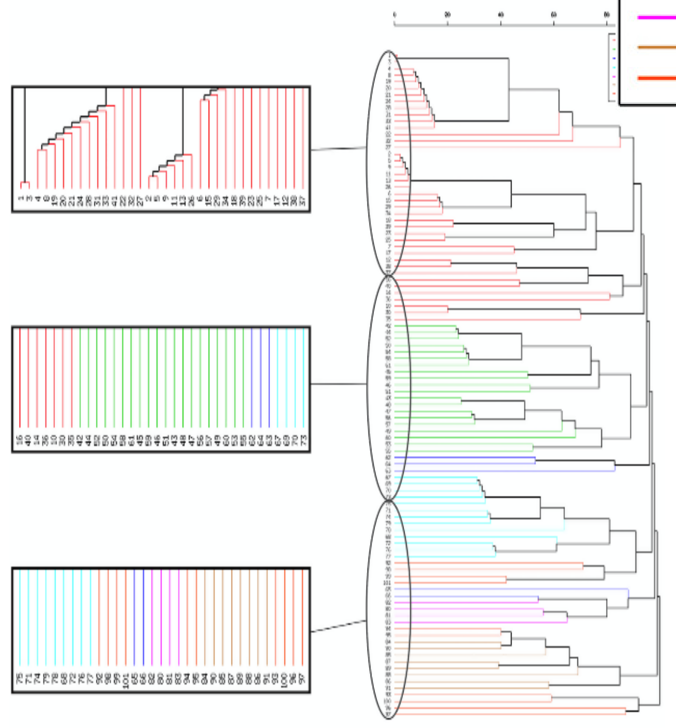


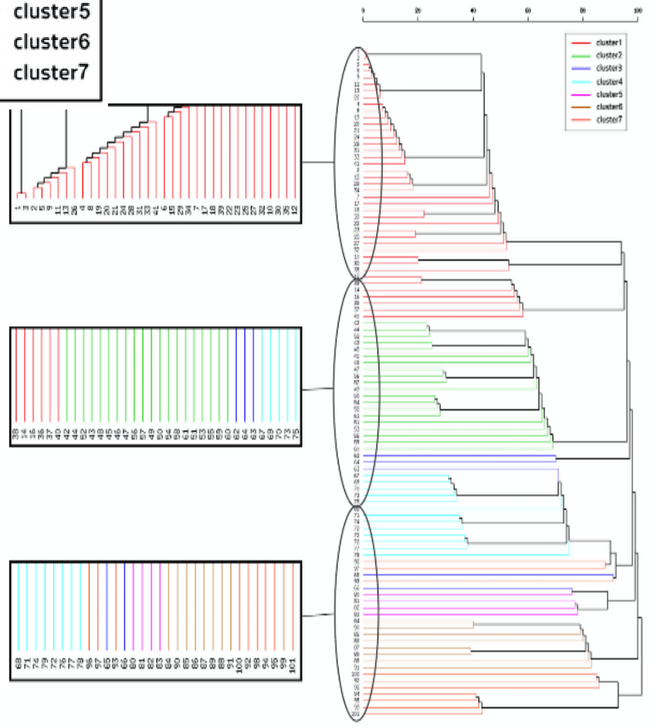




(e) Zoo - CWHAC-IP

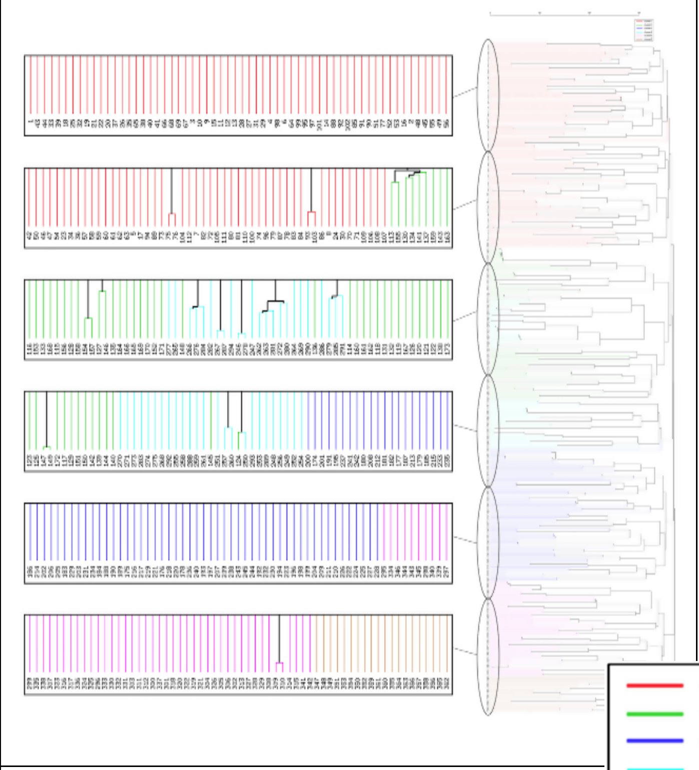
(f) Zoo - CWHC-IKM-IP



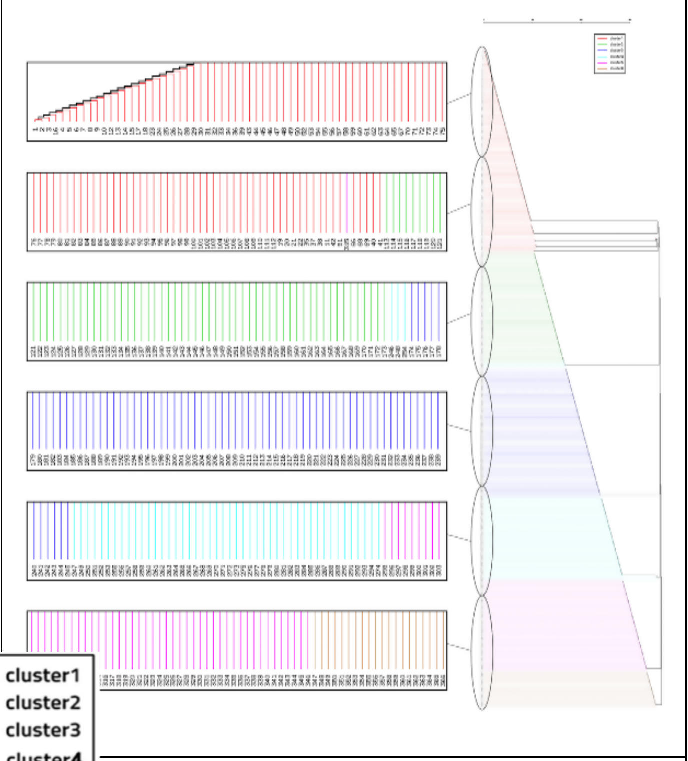


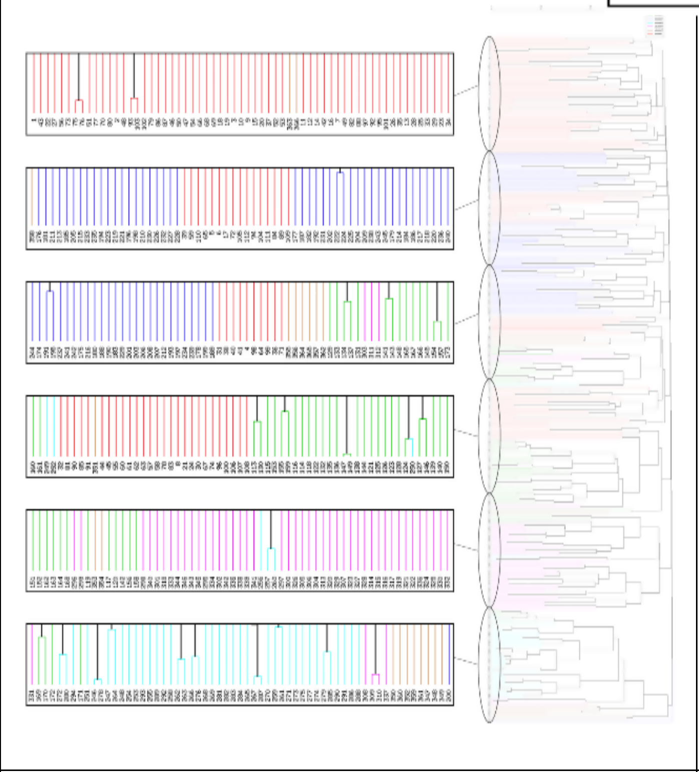
(g) Zoo - CWHAC-UT

(h) Zoo - CWHC-IKM-UT



(i) Dermatology - CWHAC-IP


(j) Dermatology - CWHC-IKM-IP


(k) Dermatology - CWHAC-UT

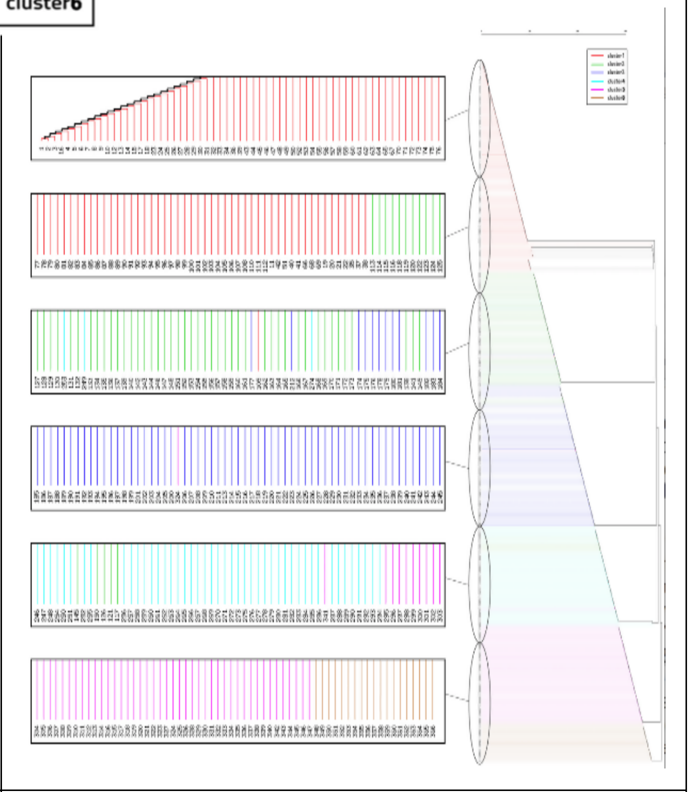

(l) Dermatology - CWHC-IKM-UT

Figure 5. 5. Dendrogram charts illustrating CWHAC and CWH-IKM clustering results for Iris, Zoo, and Dermatology datasets.

5.5.2 Efficiency Measurement and Validation

To evaluate the clustering efficiency of the CWHAC and CWHC-IKM methods evaluations founded upon the two constrained optimization techniques (IPoptim and UltraTran) for four datasets (Iris, Dermatology, BCWD and Banknote) is carried out. These are presented in Figure 5.6 and Table 5.1, wherein the comparative results for the datasets are shown to correlate with the various constraints as manifested by execution times as measured in seconds. Table 5.2 presents results of execution time in seconds for eight datasets with 60% of constraints. In both Table 5.1 and 5.2, the optimum findings have been highlighted.

Figure 5.6 reveals that there are significant increases in execution time when increasing proportions of constraints within each dataset using both CWHC-IKM and CWHAC techniques. Furthermore, CWHC-IKM with UltraTran reports minimum time as compared with other approaches with a different percentage of constraints in most of the datasets. Nevertheless, CWHC-IKM with BCWD dataset increases the efficiency with IPoptim.

When comparing the behavior towards CWHC-IKM and CWHAC of the constrained optimization techniques (IPoptim and UltraTran), the execution time for both techniques within CWHC-IKM is observed to be slightly improved in comparison to CWHAC in most datasets. There is a significant improvement in performance CWHAC with UltraTran when compared with IPoptim, such as with Ionospehere, Zoo, and Dermatology, as per Table 5.2. Additionally, since the execution time is decreased once the ik-means is applied using both IPoptim and UltraTran, the UltraTran clearly outperforms the IPoptim in relation to clustering efficiency.

Table 5.1. Illustrates to what extent the decrease in execution time in seconds with the proposed CWHC-IKM method using both IPoptim and UltraTran over the corresponding CWHAC method for different proportions of constraints. It is can be observed from Table 5.1 that the execution time in seconds decreases when using CWHC-IKM method with both IPoptim and UltraTran. It shows significant decrease for CWHC-IKM-IP with Iris (10% and 40%), BCWO (10% and 30%) and Dermatology when using 10% to 60% of

constraints, while for CWHC-IKM-UT with BCWD at low amount of constraints (10% and 30%).

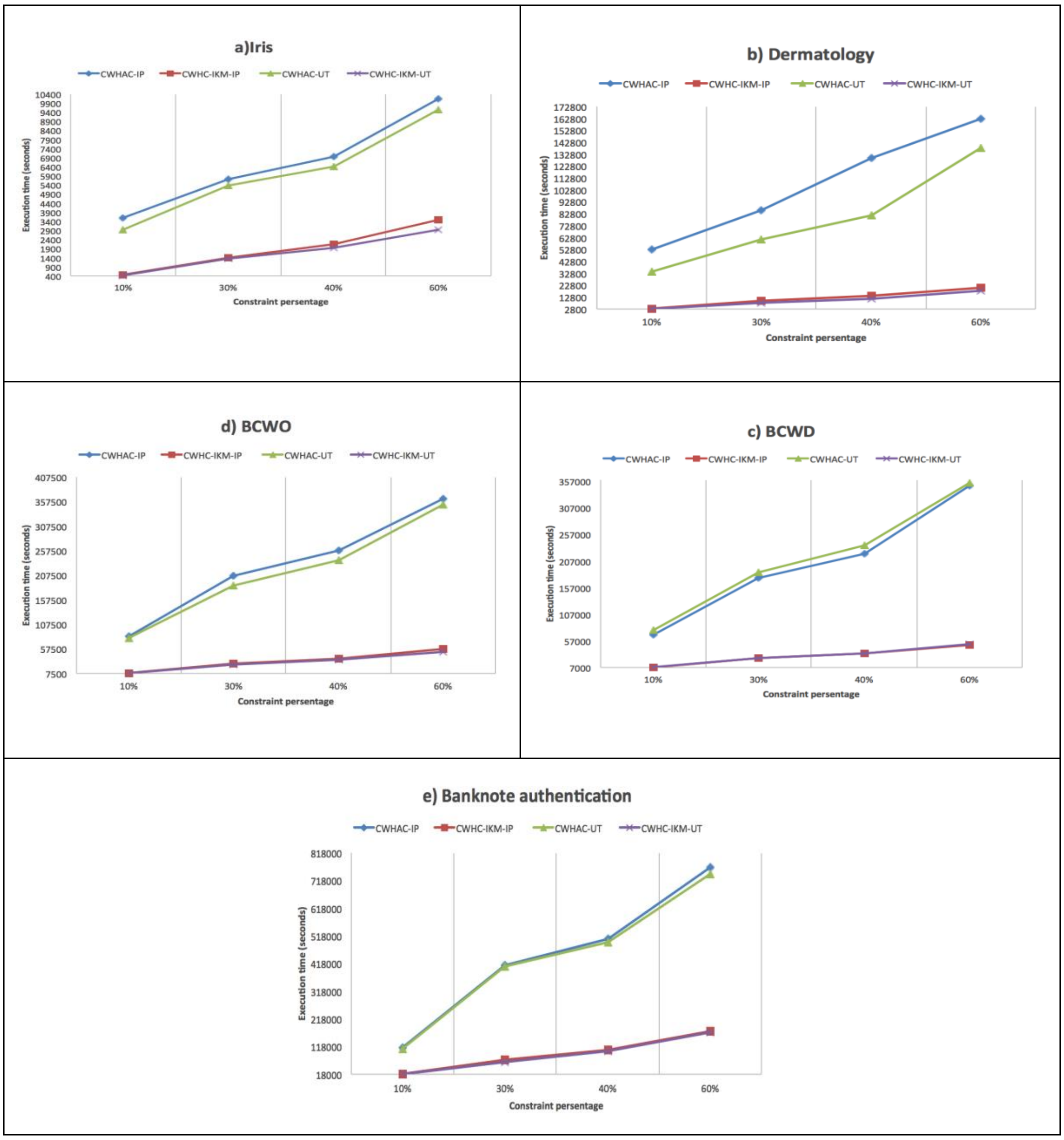


Figure 5. 6. Running time in seconds for CWHAC-IP, CWHC-IKM-IP, CWHAC-UT and CWHC-IKM-UT with varying amount of constraints.

Table 5. 1. The difference in results of execution time in seconds between the CWHAC IP and CWHC-IKM IP; and between the CWHAC IP and CWHC-IKM IP.

Dataset	**Method**	**Constraints**			
		10%	**30%**	**40%**	**60%**
Iris	CWHC - IPoptim	**3125**	4317	**4811**	6685
	CWHC- UltraTran	2508	4049	4494	6572
Dermatology	CWHC - IPoptim	**49261**	**75448**	**115452**	**141691**
	CWHC- UltraTran	31062	52775	69941	120353
BCWO	CWHC - IPoptim	**74029**	**178767**	220364	304851
	CWHC- UltraTran	71949	160304	203771	298166
BCWD	CWHC - IPoptim	61256	151455	188137	299815
	CWHC- UltraTran	**69453**	**160357**	203104	302745
Banknote authentication	CWHC - IPoptim	95893	344502	403342	591494
	CWHC - UltraTran	92725	344256	395076	574516

CWHC – IPoptim (IP) = CWHAC IP - CWHC-IKM IP
CWHC – UltraTran (UT) = CWHAC UT - CWHC-IKM UT

Table 5. 2. Results of execution time in seconds for CWHAC-IP, CWHC-IKM-IP, CWHAC-UT and CWHC-IKM-UT with 60% of constraints for eight datasets.

Dataset	**Method**	**Time (Second)**	**Dataset**	**Method**	**Time (Second)**
Iris	CWHAC-IP	10157	BCWO	CWHAC-IP	362875
	CWHC-IKM -IP	3472		CWHC-IKM -IP	58024
	CWHAC-UT	9531		CWHAC-UT	350637
	CWHC-IKM-UT	**2959**		CWHC-IKM-UT	**52471**
ZOO	CWHAC-IP	6148	Ionosphere	CWHAC-IP	148725
	CWHC-IKM -IP	1935		CWHC-IKM -IP	17513
	CWHAC-UT	5693		CWHAC-UT	136491
	CWHC-IKM-UT	**1759**		CWHC-IKM-UT	**14507**
Dermatology	CWHAC-IP	162507	Mammographic	CWHAC-IP	480795
	CWHC-IKM -IP	20816		CWHC-IKM -IP	76319
	CWHAC-UT	138205		CWHAC-UT	472061
	CWHC-IKM-UT	**17852**		CWHC-IKM-UT	**75802**
BCWD	CWHAC-IP	349570	Banknote authentication	CWHAC-IP	765318
	CWHC-IKM -IP	**49755**		CWHC-IKM -IP	173824
	CWHAC-UT	354168		CWHAC-UT	742049
	CWHC-IKM-UT	51423		CWHC-IKM-UT	**167533**

From the comparative results presented in the above paragraphs, followings points can be summarized.

- Significant improvement in clustering performance for the majority of datasets is noted when using the CWHC-IKM.
- In most of the cases, IPoptim with two methods (CWHC-IKM and CWHAC) outperforms UltraTran in terms of the F-score and CH measures.
- CWHAC algorithm is more stable than CWHC-IKM. For the majority of datasets, UltraTran proved stability and efficiency as compared to IPoptim.

- Typically, when one constrained optimization method generated enhanced results within CWHC-IKM, a comparable performence is identified in CWHAC in relation to most of the datasets used.
- Semi-supervised HC algorithms with constraints (CWHC-IKM and CWHAC) perform better than HC without constraints (WHC and A-Ward) in cluster quality results. In all datasets, the F-score values of the algorithms using constraints are consistently higher than those of algorithms without constraints.
- There are consistency and agreement between the results of the CH values and accuracy values with respect to CWHC-IKM with IPoptim, which show better performance for both class-agreement and cluster quality, compared with other methods in five of the datasets.
- There is an improvement of the F-score and CH values for CWHC-IKM and CWHAC using IPoptim optimization methods, when there is an increase in the proportion of constraints.
- The new hybrid method (CWHC-IKM) substantially improves the clustering time of traditional HAC method with constraints. The best performance is found with the CWHC-IKM with UltraTran, compared with other methods, in relation to most of the datasets.

To further investigate the effectiveness of CWHC algorithm based on two approaches of the initial clustering setting, the Mann-Whitney test [141] is utilized to compare the CWHC algorithm with ik-means approach (CWHC-IKM) with another agglomerative approach (CWHAC) to detect statistical significance in the differences in the performance of the algorithm considering an α of 0.01 ($p < 0.01$). A comparative analysis is performed on the basis of both of the constraint optimization methods, namely, IPoptim and UltraTran. The tested combination involved IPoptim within CWHC-IKM (CWHC-IKM-IP), IPoptim within CWHAC (CWHAC-IP), UltraTran within CWHC-IKM (CWHC-IKM-UT), and UltraTran within CWHAC (CWHAC-UT), with varying numbers of constraints in terms of F-score values. For each constraint, a test is conducted, a test of the CWHC-IKM-IP versus CWHAC-IP, and CWHC-IKM-UT versus CWHAC-UT using F-score values. The outcome of the Mann-Whitney test is presented in Table 5.3, and all significant improvements are highlighted.

It can be noticed from Table 5.3 that here is significant improvement for CWHC-IKM with both IPoptim and UltraTran, particularly when using large amount of constraints, which are shown with Iris and Dermatology. A similar result is found with Zoo at 30% of constraints and with BCWO at 60%. As discussed above, CWHAC outperforms CWHC-IK on three datasets, but it does not show significant improvement on Mammographic, unlike on Ionosphere with different percentages of constraints and BCWD with low amount of constraints (10 % and 30%).

Table 5. 3 Results of P-values obtained using Mann-Whitney test [141] (CWHAC-IP versus CWHC-IKM-IP, and CWHAC-UT versus CWHC-IKM-UT) with varying amount of constraints.

Dataset	**Methods**	**Constraints Percentage**			
		10%	**30%**	**40%**	**60%**
Iris	CWHAC-IP Vs. CWHC-IKM-IP	0.01044	**0.00029**	**0.00029**	**0.00009**
	CWHAC-UT Vs. CWHC-IKM-UT	0.23576	**0.00009**	**0.00009**	**0.00009**
Zoo	CWHAC-IP Vs. CWHC-IKM-IP	0.17361	**0.00368**	0.35197	0.03754
	CWHAC-UT Vs. CWHC-IKM-UT	0.01876	**0.00029**	0.26109	0.45620
Dermatology	CWHAC-IP Vs. CWHC-IKM-IP	0.39743	0.05155	0.23576	**0.00009**
	CWHAC-UT Vs. CWHC-IKM-UT	**0.00009**	**0.00009**	**0.00009**	**0.00009**
Ionosphere	CWHAC-IP Vs. CWHC-IKM-IP	0.01876	**0.00009**	**0.00009**	**0.00009**
	CWHAC-UT Vs. CWHC-IKM-UT	**0.00233**	0.01287	**0.00009**	**0.00776**
BCWO	CWHAC-IP Vs. CWHC-IKM-IP	0.05155	0.36393	0.02275	**0.00570**
	CWHAC-UT Vs. CWHC-IKM-UT	0.05155	0.39743	0.07780	0.03216
BCWD	CWHAC-IP Vs. CWHC-IKM-IP	**0.00139**	0.09342	0.09342	0.05155
	CWHAC-UT Vs. CWHC-IKM-UT	**0.00009**	**0.00009**	0.01287	0.10565
Mammographic	CWHAC-IP Vs. CWHC-IKM-IP	0.45620	0.07078	0.03216	0.01287
	CWHAC-UT Vs. CWHC-IKM-UT	0.03216	0.05155	0.03754	0.01578
Banknote	CWHAC-IP Vs. CWHC-IKM-IP	0.48405	0.28434	0.36393	0.39743
	CWHAC-UT Vs. CWHC-IKM-UT	0.13567	0.33724	0.46812	0.17361

Significant improvement p < 0.01

5.6 Conclusion

This chapter has presented a novel Constrained Ward's Hierarchical Clustering (CWHC) algorithm. Using a hybrid approach named CWHC-IKM algorithm, the data is partitioned into numerous sub-clusters at the first level and then constantly merged with the sub-clusters using constrained Ward hierarchical clustering in the second level. The CWHC-IKM algorithm cluster result has been analyzed and compared, both in terms of effectiveness and efficiency, with CWHAC algorithm based on agglomerative approach that was presented in chapter-4. Thus, this chapter constitutes an appraisal of the CWHC) algorithm using two types of initializing clustering strategies namely agglomerative strategy presented in chapter-4 and ik-means strategy presented in this chapter. More specifically, the CWHC algorithms with two approaches of initializing clustering have been evaluated when employing parameter-based constrained optimization methods (IPoptim or UltraTran) to identify the most appropriate constraints parameter for each algorithm according to the varying number of constraints proposed. The performance of the proposed approaches were analyzed in combination with both the IPoptim and the UltraTran via the application of internal and external metrics, using of F-score and CH-index, to ascertain in detail the clustering quality of the proposed methods and determining the execution time to evaluate their efficiencies. Hierarchical clustering within the structure of dendrogram has been presented with two constrained optimization methods to illustrate how instances can be clustered using each approach. To conclude, the Mann-Whitney test was employed to verify the improvements in CWH-IKM in comparison to CWHAC. Results of this experiment showed that CWH-IKM is substantially faster and more quite accurate then CWHAC, highly recommended for clustering larger datasets with higher percentage of constraints.

CHAPTER 6

Learning Feature Weights for Semi-Supervised Hierarchical Clustering Methods

6.1 Introduction

The preceding chapter proposed the novel CWHC-IKM for cluster initialisation. The fundamental idea around developing this approach was to find a hidden structure within the data, through using partly constrained patterns. However, in general, some features contained within the data may not be important or irrelevant for the purpose of clustering. Thus, this chapter further investigates the CWHC-IKM algorithm setting a degree of importance of each feature for clustering purposes, by applying a novel feature weighting technique. Hence, it proposes a novel framework for Semi-supervised Ward Hierarchical Clustering algorithms with weighted features, which will be known as the Constrained Weighted Ward Hierarchical Clustering algorithm based on Intelligent K-means (CWWHC-IKM).

This chapter is structured as follows. Section, 6.2, presents the literature review, background and motivation behind the novel idea presented within this chapter. This is followed by section 6.3 which details the proposed weighted constrained method. Section, 6.4, presents the experimental results in evaluating the algorithm's performance. The penultimate section, 6.5, compares and analyses the outcomes of the empirical execution for the proposed method. Finally, the conclusion is provided in section 6.6.

6.2 Related Work

Data can inform the basis of all hypotheses, as the number of features is a critical factor for hypothesis of space [24]; however, findings suggest that classes can be predicted by patterns or functions of a hypothesis, which is based on specific data, so that the 'curse of dimensionality' occurs when the number of features contributes to exponential increases in hypothesis space as a linear increase [185], [186]. Specific algorithms might experience too much difficulty in dealing with a large number of features, which is an issue for many data sets, [21] [126], because algorithms can find the best hypothesis when the hypothesis space is smaller, as this is easier for computation when the number of features is smaller [126]. These findings report the difficulties experienced by algorithms when dealing with cluster data that has similar information within features, as normally they evaluate one feature at a time.

When relevant features need to be found simultaneously, clustering becomes more challenging [123] and similarity measures are calculated by all available features used by clustering algorithms, where equal importance is given to all features. However, clustering outcomes could be misguided by the presence of noise in data that suggests some features are irrelevant [128].

Feature weighting techniques are shown by a review of the literature on this subject to solve the selection of important attributes problem, such as the Synclus algorithm that uses k-means clustering method, as an early development proposed by DeSarbo[187]. Initial weights begin this two-stage algorithm, and subsequently initial partitioning is applied to these weights, and for the next cycle, optimisation is achieved by updating weights. To achieve an optimal set of weights, this process iterates until a successful end, but this is described as an algorithm that is computationally expensive [128].

Variable weights are calculated automatically by W-k-means algorithms that are proposed by Huang and his colleagues [188], where higher intra-cluster similarities are assigned to variables assigned with higher weights. The original k-means clustering objective function is revised during the clustering process, when weights are optimised automatically, and real-world data sets and simulated data sets are used to test the proposed algorithm. Original data is transformed when weights are used with a beta user-defined parameter, but there is not detailed information provided about this parameter in terms of its functionality or importance [128].

To resolve the variable weighting problem, unsupervised hierarchical clustering methods are used in a method proposed by De Soete [124], [189], who studied additive tree fitting and ultra-metrics in an attempt to discover optimal variable weights. This method is not able to deal with large data sets, because of the complexity of computations for the hierarchical clustering methods, but the Polak-Ribiere optimisation procedure was later proposed by Makarenkov and Legendre [190], to extend k-means clustering, but this also slowed the speed of the algorithm.

De Amorim [17] and De Amorim, Makarenkov and Mirkin [21] introduced unsupervised hierarchical clustering algorithms based on feature weights approaches, which automatically calculated each feature weight in a data set that represent the degree of relevance. Feature weights are generated due to the use of Wardp and WardpB. These methods could be applied for bioinformatics and malware taxonomy that require that the relationship between taxons is demonstrated, and when irrelevant features are common, so that it can be used in various fields of study immediately. Chapter 2 discusses weighted Ward techniques.

Coarse feature weighting and fuzzy clustering can be performed by semi-supervised clustering and an attribute discrimination (S-SCAD) algorithm with instance level constraints proposed by Frigui and Mahdi[191]. This S-SCAD algorithm uses the least number of constraints when instances should not or should reside in the same cluster to provide guidance for clustering processes, as this learns with partial supervision and feature relevance weights that are cluster-dependent and optimal. Therefore, for the relevance weight of each feature subset and the optimal partition, one objective function is minimised by the S-SCAD algorithm and completed iteratively by updating feature weights and prototype parameters for each iteration.

Cluster specific feature weights are calculated by constrained Minkowski weighted k-means (CMWK-means) proposed by De Amorim [192], where pair-wise CL and ML rules are the basis for generating cluster constraints, and to select the correct Minkowski exponent, 20% of labelled data is used by the algorithm. The study reports that constrained clustering rules used had insignificant impact on correctly clustered entity and experiments were carried out on data sets with noise features added.

It was apparent that the majority of studies included feature weighting methods having done unsupervised HC [17], [21], [124], [189], [190] which also included ssPC methods [191], [192]. Nevertheless, a literature review regarding this subject indicates that no studies have been undertaken on the ssHC techniques on the basis of feature weighting methods.

6.3 Proposed Methods

This study addresses the complexity of clustering by eliminating the effect of irrelevant or noisy features within the use of a semi-supervised clustering algorithm. Thus, the Constrained Ward Hierarchical Clustering algorithm based on intelligent K-means (CWHC-IKM) is further extended using feature weight methods. Hence, a novel framework called Constrained Weighted Ward Hierarchical Clustering algorithm based on intelligent k-means (CWWHC-IKM) is introduced which combines the semi-supervised clustering technique with a method for learning feature weights. This feature weight learning technique is used to automatically compute the weight of each feature, allowing a feature to have different degrees of relevance at different clusters thanks to the procedure of the Wardp and WardpB techniques (the details of which have been discussed in Chapter 2). Figure 6.1 depicts the suggested method for the CWWHC-IKM framework.

As a result of our findings from previous experiments that the effectiveness of the IPoptim method outperformed the UltraTran method, only the IPoptim method was used within the proposed method (CWWHC-IKM) for the Constrained ultra- metric distance matrix. The CWWHC-IKM passes through two steps during the clustering procedure as described below.

- The first step involves computing the initial distance between X[j] and X[i] which is represented by dij element of calculated matrix D, then it applies the ik-means algorithm in order to create an initial partition k* is greater than the true number or the desired number of clusters) with their centroids C_{K*} and subsequently applying triple-wise relative constraints as background knowledge on the initial partition to generate a new metric similarity (Constrained ultra- metric distance matrix D') .
- There are two stages to the second step in order to obtain hierarchical cluster dendrogram. In the initial stage, it is necessary to initialize the feature weights for Set wkv =1/V (where V is the number of features), and subsequently applying Ward's techniques based on features weight

calculating utilizing Equations 2.22 or 2.23 (found in Chapter 2) to the initial partition K*(after updating the distance based on constraints) for the merging procedure, thereby grouping the objects in each cluster into an individually detailed tree with updating their centroid to set the centroid of the new cluster to the cluster's centre of gravity as well as their feature weights are updated by applying Equation 2.24 (see Chapter 2) during the merging of clusters. Following this, the subgroups (represented by the sub-tree T1, T2…Ti) are merged and reduced until the true number of cluster is obtained (K*=K, where K is true number of cluster). At the second stage, each Ti is considered to be a cluster and is merged with another by Ward's method with feature weights into (utilizing Equations 2.22 or 2.23 (found in Chapter 2)) into one tree, T, during finding the minimum distance between clusters. During merging process of clusters, their feature weights would be updated by applying Equation 2.24. The second stage step is repeated till convergence (number of clusters K = 1 which is a single cluster of size of N data).

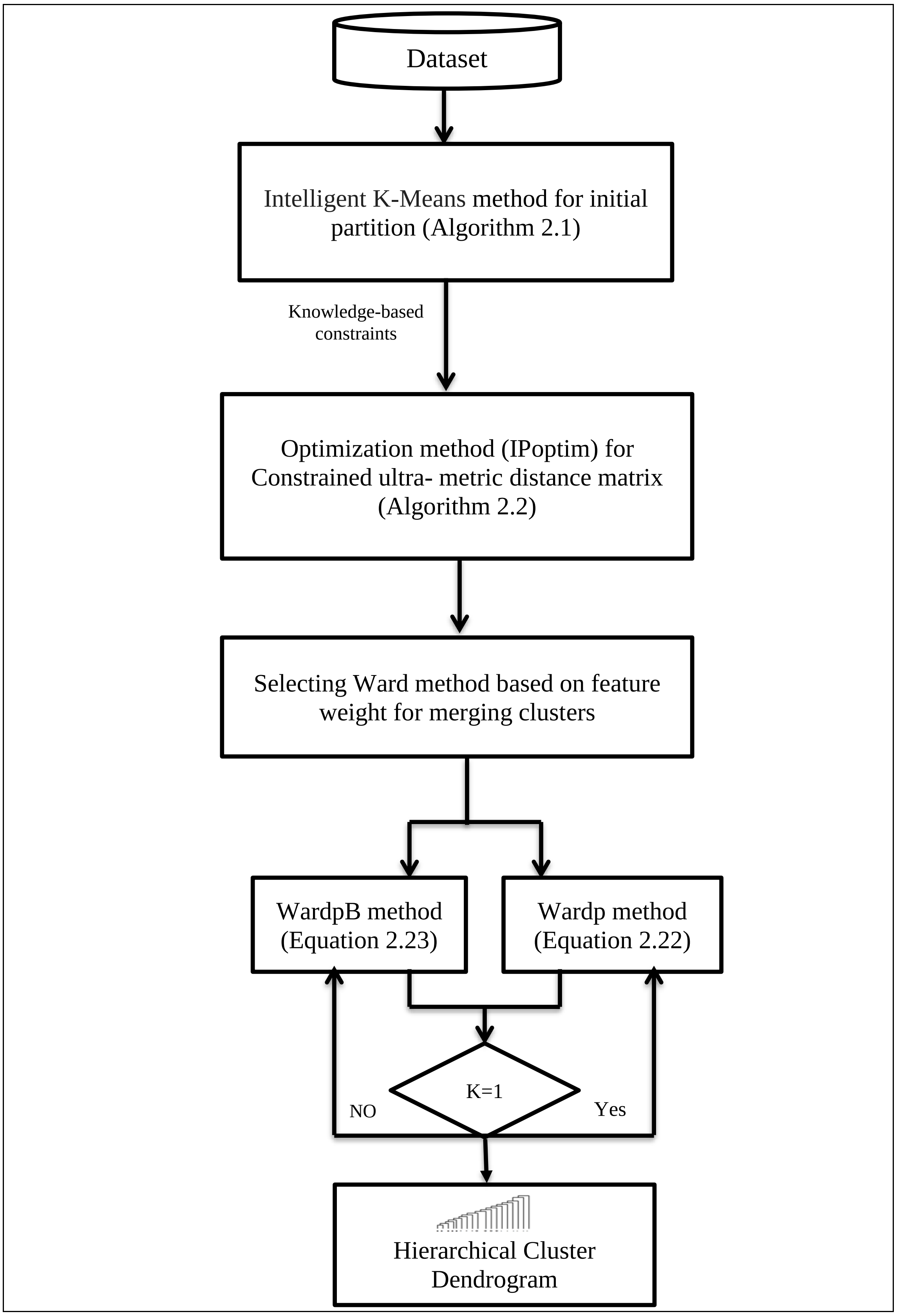

Dataset
Intelligent K-Means method for initial partition (Algorithm 2.1)
Knowledge-based constraints
Optimization method (IPoptim) for Constrained ultra- metric distance matrix (Algorithm 2.2)
Selecting Ward method based on feature weight for merging clusters
WardpB method (Equation 2.23)
Wardp method (Equation 2.22)
K=1
NO
Yes
Hierarchical Cluster Dendrogram

Figure 6. 1.Framework for Constrained Weighted Ward's Hierarchical Clustering based on intelligent K-means (CWWHC-IKM).

6.4 Experiment Methodology and Setup

The experiments were conducted on two datasets selected from UCI [155]. Appendix-A summarizes the specifications of the datasets that were used. This study followed method of previous studies [17], [21] [41], [126], [192] to add 'noise features' to the datasets (i.e. adding additional features as noise), rather than adding noise to the values of features. Two new datasets are created by adding about 50 percent of extra features as noise to the original datasets, containing uniformly distributed noise. These new datasets are used for the purpose of demonstrating the positive impact of the proposed feature weighting approach.

The intention of adopting the above approach is to make the cluster structure related challenge for data sets (like Iris and Zoo) more complex in the presence of unrelated features while yet maintaining the actual data patterns/structure. Cluster structure in a data set is often confined to a subset of features rather than the entire feature set. Adding noise values into original features of data can obscure the detection of the cluster structure. Hence, our aims is to maintain the actual data structure but 'weakening' the connotation of similarity between patterns by adding extra unrelated features in order to be able to discover the effectiveness of the proposed approach. Proposed approach based on feature weighting methods can serve to distinguish that these new features are just distractions and should be ignored in the clustering procedure.

The final clustering of the proposed technique (CWWHC-IKM) is dependent on the exponent parameters p, B which were applied. The experiments of the proposed techniques are applied differently according to the feature weighting methods (Wardp and WardpB) for the purpose of discovering the best parameters. The proposed technique which uses the Wardp criterion is dependent on a user-defined single parameter p, which indicates the rate of weight and impact distance. Therefore, these experiments are undertaken by the Wardp technique in the following way. We conducted experiments for each dataset (with and without noise) with values of P from 1 to 5 with a progress step of 0.1. We then selected the optimal for 10 % and 60 % of constraints.

However, the application of the WardpB criterion depends on two user-defined parameters p and B, which indicate the rate of impact distance and weight, respectively. Since the WardpB technique involves two parameters, it is necessary to conduct experiments for each dataset (with and without noise) with values of p from 1 to 5 with a progress step of 0.1 and with setting up B from 1 to 5 with a progress step of 0.1, after having selected the optimal parameters (p,B) for 10 % and 60 % of constraints .

Since the construction of the constraints is random, it was necessary to conduct 10 runs of the proposed technique with the optimal parameters (p and B) and to obtain the results by averaging the results of each experiment.

The proposed method is conducted by applying low and high proportions of constraints representing 10 % and 60 % respectively. This study applied only the values of p and B within the intervals of 1 and 5 for simulation, as findings of previous studies report that when the value of p is greater than 5, it has not shown best partitions [21], [41], [192]. Consequently, it is not necessary to apply values over 5 for p and B.

The proposed methods (Constrained Weighted Wardp Hierarchical Clustering based on intelligent K-means- IPoptim method (CWWp HC-IKM- IP), Constrained Weighted WardpB Hierarchical Clustering based on intelligent K-means-IPoptim method (CWWpB HC-IKM-IP)) are compared with previous algorithm (CWHC-IK-IP,see Chapter 5).

The F-score is used to analyze the agreement level achieved amongst the output of the proposed clustering methods and the correct class labels of the dataset. The empirical results of F-score value for the proposed methods are averaged in terms of their respective standard deviations.

6.5 Results and Discussion

The experimental results are used to demonstrate the clustering performance of the proposed CWWHC-IKM algorithm with constrained optimization method (IPoptim) and to identify insignificant (or noisy) variables from the given data sets. The results are presented in two sections below; the first is section A, in which the results of the algorithms are presented when using data from the original datasets without noise, the second is section B where the datasets contain random noise features.

For reporting purposes, the values of the p and B exponents were adjusted to obtain the optimal accuracy of cluster recovery. The average standard deviations of 10 runs were calculated for each dataset with each method (clean and noisy), and the values of the p and B exponents were adjusted to obtain the best F-score accuracy. Tables 6.1 and 6.2 indicate the results of the proposed method with various feature weight learning methods with deponents on the values of the p and B exponents. The best results are highlighted in bold.

A) Results on the Dataset with no Noise Features Added

Table 6.1 summarises the findings of our experiments for proposed methods with datasets without noise. The peak values of P, B exponent for the proposed method are presented. As shown in Table 6.1, the CWWp HC-IKM-IP and CWWpB HC-IKM-IP methods with the Iris dataset produce better results at 10% and 60% correspondingly, when compared to the other variants of the techniques. It is evident that the CWWpHC-IKM-IP approach attains the peak accuracy with 10% of 93.67 %, which utilises a similar exponents parameter for weight and distances (p=1.3). While 60% attains the peak accuracy 97.53% when the CWWpB HC-IKM-IP method are used, implying that it utilised a different exponent constraint for weight and distances (p=1; B=3.8).

Furthermore, although the superior performance when incorporating the feature weighted method at 10% of constraints for the Zoo dataset, represented in the CWWpB HC-IKM-IP technique, there is a variation in performance for 60% of constraints, which performs a better score in the absence of the feature weighting method, represented in the CWHC-IK-IP technique. As shown in table 6.1 the CWHC-IKM-IP approach attains peak accuracy with 60 % of 95.74%. While with 10% attains the peak accuracy 91.38 % , when the CWWpB HC-IKM-IP method is used, implying that it utilized a different exponent parameter for weight and distances (p=3.2; B=4.5).

Table 6. 1 Average accuracy F-score for comparing suggested methods with/without weighted feature methods being considered in the two datasets (minus noise features).

		Constraint%	10%		60%
Dataset	Method	Parameter	F-score	Parameter	F-score
Iris	CWHC-IK-IP	-	91.74±0.0198	-	97.11±0.0186
	CWWp HC-IKM- IP	p=1.3	**93.67±0.0236**	p=1.7	95.34±0.0173
	CWWpB HC-IKM- IP	p=1.5, B=1.3	92.08±0.0295	p=1, B=3.8	**97.53±0.0224**
	CWHC-IK-IP	-	88.52±0.0281	-	**95.74±0.0217**
Zoo	CWWp HC-IKM- IP	p = 3.7	86.34±0.0254	p=1.2	91.49±0.0193
	CWWpB HC-IKM- IP	p=3.2, B= 4.5	**91.38±0.0335**	p=4.2, B=1.8	93.56±0.0284

The p parameter for CWWp HC-IKM- IP
The p and B parameters for CWWpB HC-IKM- IP

B) Results on the Dataset with Noise Features Added

Table 6.2 summarises the findings of our experiments for proposed methods with datasets to which 50% of noise features were added. The peak values of P, B exponent for the proposed method are presented.

It shows that the feature weights methods are instrumental for partitioning the right cluster into the right group when incorporating noise in both datasets with noise (Iris and Zoo). For Iris, the most improved results are obtained at 10% (88.67) and 60 % (91.29) of constraints when adopting CWWpB-HC-IKM-IP. It requires the different value parameter for weights and distance (at p=1.1, B=1.8; and p=2.3, B=1.4 correspondingly). The CWWpB HC-IKM- IP method for the Zoo dataset also produces superior accuracy in both low and high number of constraints which obtains nearly 83.66 and 87.81 at p=4.8, B=4.9 with 10%; and p=2.1, B=1 with 60% respectively.

Table 6. 2. Averaged accuracy F-score for comparing proposed algorithms with/ without weighted feature methods being considered in the two datasets (with added noise features).

		Constraint%	10%		60%
Dataset	Method	Parameter	F-score	Parameter	F-score
Iris with feature noise	CWHC-IK-IP	-	82.59±0.0247	-	88.34±0.0196
	CWWp HC-IKM- IP	p= 1.7	85.48±0.0214	p=3	90.68±0.0174
	CWWpB HC-IKM- IP	p=1.1, B=1.8	**88.67±0.0225**	p=2.3, B=1.4	**91.29±0.0207**
Zoo with feature noise	CWHC-IK-IP	-	76.63±0.0235	-	82.96±0.0221
	CWWp HC-IKM- IP	p=4.5	83.45 ±0.0238	p=4.1	85.32±0.0206
	CWWpB HC-IKM- IP	p=4.8, B=4.9	**83.66±0.0253**	p=2.1, B=1	**87.81±0.0231**

The p parameter for CWWp HC-IKM- IP
The p and B parameters for CWWpB HC-IKM- IP

6.6 Conclusion

This chapter has proposed a novel approach for learning feature weights in constrained HC algorithms by incorporating feature weight determining techniques within semi-supervised clustering. The proposed approach is named as CWWHC-IKM. The proposed method automatically derives the weight of each feature while allowing a feature to hold varying degrees of relevance at different clusters owing to the use of the weighted Ward's techniques (Wardp and WardpB). This chapter also has aided in identifying optimum exponent result for the approaches in the context of the proposed method for a particular dataset. In Wardp, the rate of impact distance and weight is denoted by the sole parameter, p, which is user-defined, whereas in WardpB, the rate of impact distance and weight are denoted by parameters, p and B, respectively and are also user-defined. Overall, the experiments confirm the superiority of employing feature-weighting in the proposed method, especially when incorporating noise features (unrelated features) into the dataset. Thus, the proposed CWWHC-IKM method is designed to resolve challenges of dealing with noise in data and other unrelated features during the clustering process.

CHAPTER 7

Conclusion and Future Work

There is growing attention for semi-supervised clustering algorithms among data mining and machine learning communities. Even so, the existing body of literature and research works have revealed that significant progress has been made mostly in the area of semi-supervised PC [1], [5]. For this reason, the present study incorporates approaches and procedures for designing and developing a novel (HC) algorithm with existing knowledge-based constraints. Such existing knowledge has been employed as a type of triple-wise relative constraints that fit with a hierarchical structure. The techniques and approaches that employed into the development are linkage measures of the agglomerative method, distance metrics, ik-means method, devised new methods for constraint pre-processing and its time complexity and learning feature weights owing to the use of Wardp and WardpB producing novel ssHC algorithms capable of addressing different concerns. Every single approach was assigned an objective as highlighted in Section 1.5, and deployed and presented in Chapters 3, 4, 5, and 6, which make up the big portion of this study.

Chapter 3 explored a new technique regarding a ssHAC algorithm. It is the integration of the knowledge-based triple-wise relative constraints process into the linkage measure of agglomerative HC methods. A closer look at the existing body of research works revealed that studies on methods utilizing agglomerative HC with triple-wise relative constraints do not exist. A significant number of the existing research works on linkage measures of agglomerative HC are based on unsupervised HC algorithms and ssHAC algorithms with pair-wise constraints. Furthermore, the proposed algorithm (ssHAC) is designed and implemented in the context of a new framework that has been designed based on the six popular linkages of an agglomerative hierarchical procedure with the different ten distance metrics. The proposed framework with varying factors (distance metric, linkage, the two constrained optimization procedures utilizing varying levels of constraints) was assessed to ascertain the manner in which a suitable combination of a distance metric, linkage technique and the number of constraints on the functioning of the proposed algorithm (ssHAC) would be selected.

Identifying and tweaking constraints would be challenging given it is difficult to identify a functional constraint example without pre-processing. An improper selection of constraints might cause degradation of the resulting clustering instead of enhancing it. Therefore, the pertinent question is: is it possible to successfully integrate the knowledge to a HC process via triple-wise relative constraints? The answer to this question is considered in Chapter 4. The proposed methods were found to successfully improve the effectiveness and efficiency of CWHAC. Experimental findings suggested that the CWHAC can resolve the problems associated with triple-wise relative constraints within CWHAC algorithms by diminishing the amount of redundant and violated constraints, together with problems associated with the computational complexity of CWHAC algorithm by suggesting the new three-optimization protocol for reducing the time-consuming process involved in generating constraints.

Chapter 5 explored the hybrid approach, which is an alternate approach for accelerating the HAC equipped to handle large data with semi-supervision of constraints. This is the first study to proposed knowledge-based triple-wise relative constraints in the context of the hybrid technique of HC. The innovative hybrid algorithm recommended in this study is known as CWHC-IKM. It has the ability to learn from ik-means and the triple-wise relative constraint mechanism. CWH-IKM algorithm is a result of a combination of the benefits of Ward’s hierarchical clustering under knowledge-based constraints and ik-means. The quality of HC is improved through the existing triple-wise relative constraints and the ik-means algorithm, the initialization cluster technique for HC quality substantially decreases the time it takes to converge. Additionally, it paves the way for the initial partition of Ward’s hierarchical clustering, which facilitates cluster merging to originate from this partition, as opposed to from a trivial partition composed solely of individual clusters. Moreover, the CWH-IKM cluster result was then analyzed and compared for effectiveness and efficiency with CWHAC. It could be said that this chapter appraises CWHC algorithm through the utilization of two forms of initializing clustering techniques—ik-means and agglomerative—which are considered individually and each of these methods was developed separately to learn from each type. The new hybrid technique (CWH-IKM) was found to be substantially faster, quite accurate and highly dependable for clustering larger datasets with higher constraints when compared with employing the agglomerative technique.

Chapter 6 describes an innovative method (CWWHC-IKM) for learning feature Weight in the Constrained HC algorithm given in has been designed to combine feature weighs techniques with semi-supervised clustering. The proposed method automatically derives the weight of each feature while allowing them to show varying levels of relevance as different clusters owing to the use of the Wardp and WardpB approaches. This chapter also made a significant contribution toward identifying the most viable exponent result for these methods with respect to the proposed technique for a specific dataset. As for Wardp, the rate of impact distance and weight is denoted by the single parameter, p, which is set by the user, whereas for WardpB, the rate of impact distance and weight are indicated by p and B, respectively and are set by the user. Findings suggested that the CWWHC-IKM can tackle the noise issues or irrelevant features during clustering.

7.1 Limitations and Opportunities for Future Work

The findings of this study present several opportunities for future research. Even so, it is not without limitations. This section considers some limitations of the presented work as well as its significance for future works.

Given several researchers claim that knowledge-based triple-wise relative constraint is more appropriate when considering the structure if HC compared to pair-wise constraint, it was employed in this study as the primary approach to prior knowledge to develop ssHC. However, not a single research study exists providing a comprehensive analysis and comparison of employing these two forms of constraints with HC. Consequently, future research efforts should be focused on comprehending/comparing the effectiveness and efficiency or time complexity of the two types of constraints in the context of HC (on similar parameters, which is employing the same hierarchical learning algorithm for the learning of the two forms of constraints), which has yet to be studied.

The study was successful in satisfying all constraints by suggesting novel methods for pre-processing for resolving the problem of non-satisfaction or ineffective constraints, which in turn results in the improvement of the performance of the proposed algorithm when increasing the number of constraints. However, variations exist in the performance of the proposed algorithm during different runs. This can be attributed to randomly selecting constraints. While the random selection of constraints presents greater opportunities for in-depth research to discover various types of knowledge in the form of constraints. Another reason for the variation in results perhaps during different operations is that some random constraint parameters are more useful for clustering algorithms than others. According to Davidson, Wagstaff, and Basu, [164], a good constraint set parameters ought to be coherent and must have high informativeness. Consequently, selecting the potent and mode beneficial constraints of varying datasets for ssHC ought to be studied further in our futures research, as it would be of significant benefit for both researchers and practitioners.

A novel approach (CWWHC-IKM) has been proposed in chapter 6 in order to learn the feature weights in constrained HC algorithms by incorporating feature weight determining techniques within semi-supervised clustering. The proposed CWWHC-IKM method resolved challenges of dealing with noise in data and other irrelevant features during the clustering process. Unfortunately, details of which features played a more significant role in ensuring clustering accuracy was not recorded in these investigations. Such information would have led to a better understanding of the impact and the practical relevance of the research conducted in this thesis. Further research in this direction is recommended in the future.

References

[1] E. Bair, “Semi-supervised clustering methods,” Wiley Interdisciplinary Reviews: Computational Statistics, vol. 5, no. 5. pp. 349–361, 2013.

[2] D. Dinler and M. K. Tural, “A survey of constrained clustering,” in Unsupervised Learning Algorithms, pp. 207–235, 2016.

[3] A. K. H. Tung, J. Han, L. V. S. Lakshmanan, and R. T. Ng, “Constraint-based clustering in large databases,” in International Conference on Database Theory,vol.pp. 405-419, 2001 , Springer, Berlin, Heidelberg.

[4] Y. Zhao and G. Karypis, “Evaluation of hierarchical clustering algorithms for document datasets,” Proceedings of the eleventh international conference on Information and knowledge management - CIKM 02, 2002.

[5] X. Ma and S. Dhavala, “Hierarchical Clustering with Prior Knowledge,” Jun. 2018.

[6] P.-N. M. S. U. Tan, M. U. of M. Steinbach, and V. U. of M. Kumar, “Cluster Analysis: Basic concepts and algorithms,” Introduction to data mining, pp. 487–568, 2006.

[7] H. Frigui and R. Krishnapuram, “Clustering by competitive agglomeration,” Pattern Recognition, vol. 30, no. 7, pp. 1109–1119, 1997.

[8] R. O. Duda, P. E. Hart, and D. G. Stork, “Pattern classification,” New York John Wiley, Sect., 2001.

[9] J. Sander, X. Qin, Z. Lu, N. Niu, and A. Kovarsky, “Automatic extraction of clusters from hierarchical clustering representations,” in Pacific-Asia Conference on Knowledge Discovery and Data Mining, vol. 2637, pp. 75-872003, Springer, Berlin, Heidelberg.

[10] K.L. Wagstaff and C. Cardie, “Clustering with Instance-level Constraints,” in: Proceedings of the 17th International Conference on Machine Learning, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc, pp. 1103–1110, 2000.

[11] L. Zheng and T. Li, “Semi-supervised Hierarchical Clustering,” in

IEEE 11th International Conference on Data Mining, pp. 982–991, 2011.

[12] B. Chen, P. C. Tai, R. Harrison, and Y. Pan, “Novel hybrid hierarchical-K-means clustering method (H-K-means) for microarray analysis,” in IEEE Computational Systems Bioinformatics Conference, Workshops and Poster Abstracts, pp. 105–108, 2005 .

[13] Y. Tamura, N. Obara, and S. Miyamoto, “A method of two-stage clustering with constraints using agglomerative hierarchical algorithm and one-pass k-means++,” in Knowledge and Systems Engineering, vol. 245, pp. 9–19, 2014.

[14] C. R. Lin and M. S. Chen, “Combining partitional and hierarchical algorithms for robust and efficient data clustering with cohesion self-merging,” Transactions on Knowledge and Data Engineering, vol. 17, no. 2, pp. 145–159, 2005.

[15] M. F. Balcan , Y. Liang and P. Gupta, “Robust hierarchical clustering,” vol. 15, no. 1, pp. 3831-3871, 2010.

[16] J. A. S. Almeida, L. M. S. Barbosa, A. A. C. C. Pais, and S. J. Formosinho, “Improving hierarchical cluster analysis: A new method with outlier detection and automatic clustering,” Chemometrics and Intelligent Laboratory Systems., vol. 87, no. 2, pp. 208–217, 2007.

[17] R. C. De Amorim, “Feature Relevance in Ward’s Hierarchical Clustering Using the L p Norm,” J. Classif., vol. 32, no. 1, pp. 46–62, 2015.

[18] B. G. Mirkin, Clustering for data mining : a data recovery approach. Chapman & Hall/CRC, 2005.

[19] A. Aljohani, D. T. C. Lai, P. C. Bell, and E. A. Edirisinghe, “A Comparison of Distance Metrics in Semi-supervised Hierarchical Clustering Methods,” in International Conference on Intelligent Computing, vol. 10363 LNAI, pp. 719–731, 2017.

[20] A. A. Aljohani, E. A. Edirisinghe, and D. T. C. Lai, “An Effective and Efficient Constrained Ward’s Hierarchical Agglomerative Clustering Method,”pp. 590–611, 2019 .

[21] R. C. De Amorim, V. Makarenkov, and B. Mirkin, “A-Wardpβ:

Effective hierarchical clustering using the Minkowski metric and a fast k-means initialisation," Information Sciences., vol. 370–371, pp. 343–354, 2016.

[22] M. E. A. Brea, "Constrained Clustering Algorithms: Practical Issues and Applications", PhD Thesis, 2013.

[23] D. T. C. Lai, "An Exploration of Improvements to Semi-supervised Fuzzy c-Means Clustering for Real-World Biomedical Data," PhD Thesis, University of Nottingham, 2014.

[24] T. Mitchell, "Machine Learning," in Computer, pp. 870–877, 1997 .

[25] A. Albalate and W. Minker, "Semi-Supervised and Unsupervised Machine Learning: Novel Strategies". John Wiley & Sons, Ltd, 2013.

[26] R. Capaldo and F. Collovà, "Clustering : A survey," 2008. [Online]. Available:https://www.slideshare.net/rcapaldo/cluster-analysis-presentation. [Accessed: 05-Nov-2019].

[27] P. Orbanz and J. M. Buhmann, "Nonparametric Bayesian image segmentation," Int. J. Comput. Vis., vol. 77, no. 1–3, pp. 25–45, 2008.

[28] S. Zeng, R. Huang, Z. Kang, and N. Sang, "Image segmentation using spectral clustering of Gaussian mixture models," Neurocomputing, vol. 144, pp. 346–356, Nov. 2014.

[29] K. Voevodski, M. F. Balcan, H. Röglin, S. H. Teng, and Y. Xia, "Active clustering of biological sequences," J. Mach. Learn. Res., vol. 13, pp. 203–225, 2012.

[30] I. C. McDowell, D. Manandhar, C. M. Vockley, A. K. Schmid, T. E. Reddy, and B. E. Engelhardt, "Clustering gene expression time series data using an infinite Gaussian process mixture model," PLoS Comput. Biol., vol. 14, no. 1, 2018.

[31] J. W. Van Dam and M. Van De Velden, "Online profiling and clustering of Facebook users," Decision Support Systems, vol. 70, pp. 60–72, 2015.

[32] J. Surma, Business Intelligence: Making Decisions Through Data Analytics. Business Expert Press, 2011.

[33] C. Anderson, D. Lee, and N. Dean, “Identifying clusters in Bayesian disease mapping,” Biostatistics, vol. 15, no. 3, pp. 457–469, 2014.

[34] J. Han, J. Pei and M. Kamber, “Data Mining: Concepts and Techniques”. Elsevier Inc., 2012.

[35] M. Yenugula, “Hierarchical Clustering and Interaction,” UC, San Diego, , PhD Thesis, 2016.

[36] J. Qi, Y. Yu, L. Wang, J. Liu, and Y. Wang, “An effective and efficient hierarchical K-means clustering algorithm,” Int. J. Distrib. Sens. Networks, vol. 13, no. 8, pp. 1–17, Aug. 2017.

[37] J. G. Dy, “Unsupervised Feature Selection,” in Liu, H. and Motoda, H. Computational Methods of Feature Selection., pp.19–34, 2007 .

[38] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, “An efficient k-means clustering algorithms: Analysis and implementation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, no. 7, pp. 881–892, 2002.

[39] F. Camastra and A. Verri, “A novel Kernel method for clustering,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, no. 5, pp. 801–805, 2005.

[40] M. M. T. Chiang and B. Mirkin, “Intelligent choice of the number of clusters in k-means clustering: An experimental study with different cluster spreads,” J. Classif., vol. 27, no. 1, pp. 3–40, 2010.

[41] R. C. De Amorim and B. Mirkin, “Minkowski metric, feature weighting and anomalous cluster initializing in K-Means clustering,” Pattern Recognit., vol. 45, no. 3, pp. 1061–1075, 2012.

[42] S. C. Johnson, “Hierarchical clustering schemes,” Psychometrika, vol. 32, no. 3, pp. 241–254, 1967.

[43] D. T. Pham and A. A. Afify, “Engineering Applications of Clustering Techniques,” in Intelligent Production Machines and Systems, Elsevier Ltd, 2006, pp. 326–331.

[44] A. K. Jain and R. C. Dubes, Algorithms for Clustering Data. Prentice Hall, Englewood Cliffs, 1988.

[45] P. J. Rousseuw and L. Kaufman, Finding Groups in Data: An Introduction to Cluster Analysis. Hoboken: Wiley Online Library, 1990.

[46] K. M. Rafsanjani, Z. A. Varzaneh, and N. E. Chukanlo, “A Survey Of Hierarchical Clustering Algorithms,” J. Math. Comput. Sci., vol. 05, no. 03, pp. 229–240, Oct. 2012.

[47] L. Feng, M. H. Qiu, Y. X. Wang, Q. L. Xiang, Y. F. Yang, and K. Liu, “A fast divisive clustering algorithm using an improved discrete particle swarm optimizer,” Pattern Recognit. Lett., vol. 31, no. 11, pp. 1216–1225, 2010.

[48] P. Wittek, “Quantum Machine Learning: What Quantum Computing Means to Data Mining,” 2014.

[49] G. Gan, C. Ma, and J. Wu, Data Clustering: Theory, Algorithms, and Applications, vol. 20, 2007.

[50] O. Yim and K. T. Ramdeen, “Hierarchical Cluster Analysis: Comparison of Three Linkage Measures,” in The Quantitative Methods for Psychology, vol. 11, no. 1, 2015, pp. 8–24.

[51] M. Mazzocchi, Statistics for Marketing and Consumer Research. Sage, 2008.

[52] G. N. Lance and W. T. Williams, “A General Theory of Classificatory Sorting Strategies: 1. Hierarchical Systems,” Comput. J., vol. 9, no. 4, pp. 373–380, 1967.

[53] P. H. Sneath, “The application of computers to taxonomy.,” J. Gen. Microbiol., vol. 17, no. 1, pp. 201–226, Aug. 1957.

[54] U. Maulik and S. Bandyopadhyay, “Performance evaluation of some clustering algorithms and validity indices,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 24, no. 12, pp. 1650–1654, 2002.

[55] V. Kumar, J. K. Chhabra, and D. Kumar, “Performance Evaluation of Distance Metrics in the Clustering Algorithms,” Infocomp J. Comput. Sci., vol. 13, no. 1, pp. 38–52, 2014.

[56] C. Sokal and R. Michener, “A statistical method for evaluating systematic relationships,” Univ. Kansas Sci. Bull., vol. 38, pp. 1409–

1438, 1958.

[57] S. Yashwant and S. L. Sananse, “Comparisons of Different Methods of Cluster Analysis with Application to Rainfall Data,” Int. J. Innov. Res. Sci. Eng. Technol., vol. 4, no. 1981, pp. 10861–10872, 2015.

[58] S. Miyamoto and A. Terami, “Semi-supervised agglomerative hierarchical clustering algorithms with pairwise constraints,” in IEEE World Congress on Computational Intelligence, WCCI, pp. 1-6 IEEE, 2010.

[59] J. H. Ward, “Hierarchical Grouping to Optimize an Objective Function,” J. Am. Stat. Assoc., vol. 58, no. 301, pp. 236–244, 1963.

[60] F. Murtagh and P. Legendre, “Ward’s Hierarchical Agglomerative Clustering Method: Which Algorithms Implement Ward’s Criterion?,” J. Classif., vol. 31, no. 3, pp. 274–295, 2014.

[61] G. M. Downs and J. M. Barnard, “Clustering Methods and Their Uses in Computational Chemistry,” in Reviews in Computational Chemistry, vol. 18, pp. 1–40, 2003.

[62] P. J. Ortega, B. M. D. R. Rojas, and S. M. J. García, “Research issues on k-means algorithm: An experimental trial using Matlab,” in CEUR Workshop Proceedings, vol. 534, pp. 83–96, 2009.

[63] T. Hertz, “Learning Distance Functions : Algorithms and Applications,” 2006.

[64] J. Irani, N. Pise, and M. Phatak, “Clustering Techniques and the Similarity Measures used in Clustering: A Survey,” Int. J. Comput. Appl., vol. 134, no. 7, pp. 9–14, 2016.

[65] V. R. Patel and R. G. Mehta, “Data clustering: Integrating different distance measures with modified k-means algorithm,” in Advances in Intelligent and Soft Computing, vol. 131 AISC, no. 2, pp. 691–700, 2012 .

[66] A. Huang, “Similarity measures for text document clustering,” in New Zealand Computer Science Research Student Conference, NZCSRSC 2008 - Proceedings, pp. 49–56, 2008.

[67] F. Cao, J. Liang, D. Li, L. Bai, and C. Dang, “A dissimilarity measure

for the k-Modes clustering algorithm," Knowledge-Based Syst., vol. 26, pp. 120–127, 2012.

[68] R. Xu and D. Wunsch, "Survey of clustering algorithms," IEEE Transactions on Neural Networks, vol. 16, no. 3. pp. 645–678, May-2005.

[69] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical pattern recognition: A review," IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 1, pp. 4–37, 2000.

[70] P. C. Mahalanobis, "On the generilised distance in statistics," Proceedings of the National Institute of Sciences of India. 1936.

[71] R. De Maesschalck, D. Jouan-Rimbaud, and D. L. Massart, "The Mahalanobis distance," Chemom. Intell. Lab. Syst., vol. 50, no. 1, pp. 1–18, 2000.

[72] L. M. Cherry, S. M. Case, J. G. Kunkel, J. S. Wyles, and A. C. Wilson, "Body shape metrics and organismal evolution," Evolution (N. Y)., vol. 36, no. 5, p. 914-933, 1982.

[73] Z. Prekopcsák and D. Lemire, "Time series classification by class-specific Mahalanobis distance measures," Adv. Data Anal. Classif., vol. 6, no. 3, pp. 185–200, 2012.

[74] S. Pandit and S. Gupta, "A Comparative Study on Distance Measuring Approaches for Clustering," Int. J. Res. Comput. Sci., vol. 2, no. 1, pp. 29–31, 2011.

[75] B. Larsen and C. Aone, "Fast and effective text mining using linear-time document clustering," in International Conference on Knowledge Discovery and Data Mining, pp. 16–22, 1999.

[76] B. R. A. Yates and R. B. Neto, "Modern Information Retrieval," 1999.

[77] B. Charulatha, P. Rodrigues, T. Chitralekha, and A. Rajaraman, "A Comparative study of different distance metrics that can be used in Fuzzy Clustering Algorithms," Int. J. Emerg. Trends Technol. Comput. Sci., 2013.

[78] M. H. Fulekar, Bioinformatics: Applications in Life and Environmental Sciences. 2009.

[79] R. Potolea, S. Cacoveanu, and C. Lemnaru, “Meta-learning framework for prediction strategy evaluation,” in International Conference on Enterprise Information Systems, vol. 73 LNBIP, pp. 280–295, 2011.

[80] A. Singh, A. Yadav, and A. Rana, “K-means with Three different Distance Metrics,” Int. J. Comput. Appl., vol. 67, no. 10, pp. 13–17, 2013.

[81] G. N. Lance and W. T. Williams, “Computer Programs for Hierarchical Polythetic Classification (‘Similarity Analyses’),” Comput. J., vol. 9, no. 1, pp. 60–64, 1966.

[82] J. R. Bray and J. T. Curtis, “An Ordination of the Upland Forest Communities of Southern Wisconsin,” Ecol. Monogr., vol. 27, no. 4, pp. 325–349, 1957.

[83] I. Davidson and S. Basu, “A Survey of Clustering with Instance Level Constraints,” ACM Trans. Knowl. Discov. Data, pp. 1–41, 2007.

[84] X. Zhu, Semi-Supervised Learning Literature Survey - TR 1530. University of Wisconsin-Madison Department of Computer Sciences, Sci. York, 2005.

[85] N. N. Pise and P. Kulkarni, “A survey of semi-supervised learning methods,” in Proceedings - International Conference on Computational Intelligence and Security, CIS, 2008, vol. 2, pp. 30–34.

[86] S. Basu, “Semi-supervised Clustering: Learning with Limited User Feedback,” PhD Thesis Austin, 2003.

[87] K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell, “Text classification from labeled and unlabeled documents using EM,” Mach. Learn., vol. 39, no. 2, pp. 103–134, 2000.

[88] Z. Lu and T. K. Leen, “Semi-supervised clustering with pairwise constraints: A discriminative approach,” in Journal of Machine Learning Research, vol. 2, pp. 299–306, 2007.

[89] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, “Constrained K-means Clustering with Background Knowledge,” in International Conference on Machine Learning ICML, vol. 1, pp. 577–584, 2001.

[90] V. Chatziafratis, R. Niazadeh, and M. Charikar, “Hierarchical clustering with structural constraints,” in 35th International Conference on Machine Learning, ICML,vol. 2, pp. 1213–1226, 2018.

[91] S. Zhang, H. S. Wong, and D. Xie, “Semi-supervised clustering with pairwise and size constraints,” in Proceedings of the International Joint Conference on Neural Networks, pp. 2450–2457, 2014.

[92] F. Gao, Z. Yue, Q. Xiong, J. Wang, E. Yang, and A. Hussain, “A Novel Semi-supervised Classification Method Based on Class Certainty of Samples,” in International Conference on Brain Inspired Cognitive Systems, pp. 315-324. , 2018.

[93] Z. H. Zhou and M. Li, “Semi-supervised learning by disagreement,” Knowl. Inf. Syst., vol. 24, no. 3, pp. 415–439, 2010.

[94] N. V. Chawla and G. Karakoulas, “Learning from labeled and unlabeled data: An empirical study across techniques and domains,” J. Artif. Intell. Res., vol. 23, pp. 331–366, 2005.

[95] J. N. Vittaut, M. R. Amini, and P. Gallinari, “Learning classification with both labeled and unlabeled data,” in European Conference on Machine Learning, vol. 2430, pp. 468–479, 2002.

[96] M. A. Rizoiu, “Semi-supervised structuring of complex data,” in IJCAI International Joint Conference on Artificial Intelligence, pp. 3239–3240,2013.

[97] A. K. Jain, “Data clustering: 50 years beyond K-means,” Pattern Recognit. Lett., vol. 31, no. 8, pp. 651–666, 2010.

[98] S. Basu, M. Bilenko, and R. J. Mooney, “A probabilistic framework for semi-supervised clustering,” in KDD- Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 59–68, 2004.

[99] K. L. Wagstaff, “Constrained Clustering,” in Encyclopedia of Machine Learning, Boston, MA: Springer US, pp. 220–221, 2011.

[100] C. Zhigang, L. Xuan and Y. Fan, “Constrained K-means with external information,” in Proceedings of the 8th International Conference on Computer Science and Education, ICCSE, 2013, pp. 490–493.

[101] A. Demiriz, K. P. Bennett, and M. J. Embrechts, "Semi-supervised clustering using genetic algorithms," in Intelligent Engineering Systems Through Artificial Neural Networks, vol. 9, pp. 809–814,1999.

[102] S. Basu, A. Banerjee, and R. Mooney, "Semi-supervised Clustering by Seeding," Proc. 19th Int. Conf. Mach. Learn., no. July, pp. 19–26, 2002.

[103] D. Cohn, R. Caruana, and A. Kachites McCallum, "Semi-supervised clustering with user feedback," in Constrained Clustering: Advances in Algorithms, Theory, and Applications, pp. 17–31, 2003.

[104] D. Klein, S. D. Kamvar, and C. D. Manning, "From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering," Int. Conf. Mach. Learn., pp. 307–314, 2002.

[105] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, "Distance metric learning with application to clustering with side-information," in Advances in Neural Information Processing Systems,pp. 521–528, 2003.

[106] H. Chang and D. Y. Yeung, "Locally linear metric adaptation with application to semi-supervised clustering and image retrieval," Pattern Recognit., vol. 39, no. 7, pp. 1253–1264, 2006.

[107] R. Filipovych, S. M. Resnick, and C. Davatzikos, "Semi-supervised cluster analysis of imaging data," Neuroimage, vol. 54, no. 3, pp. 2185–2197, 2011.

[108] N. Grira, M. Crucianu, and N. Boujemaa, "Active semi-supervised fuzzy clustering," Pattern Recognit., vol. 41, no. 5, pp. 1834–1844, 2008.

[109] I. A. Maraziotis, "A semi-supervised fuzzy clustering algorithm applied to gene expression data," Pattern Recognit., vol. 45, no. 1, pp. 637–648, 2012.

[110] P. D. McNicholas and S. Subedi, "Clustering gene expression time course data using mixtures of multivariate t-distributions," J. Stat. Plan. Inference, vol. 142, no. 5, pp. 1114–1127, 2012.

[111] E. Ben Ahmed, A. Nabli, and F. Gargouri, “Group extraction from professional social network using a new semi-supervised hierarchical clustering,” Knowl. Inf. Syst., vol. 40, no. 1, pp. 29–47, 2014.

[112] I. Davidson and S. S. Ravi, “Using instance-level constraints in agglomerative hierarchical clustering: Theoretical and empirical results,” Data Min. Knowl. Discov., vol. 18, no. 2, pp. 257–282, 2009.

[113] X. Liu, “Joint Constrained Clustering and Feature Learning based on Deep Neural Networks,” Master dissertation, Applied Sciences: School of Computing Science, 2017.

[114] Y. Pei, X. Z. Fern, T. V. Tjahja, and R. Rosales, “Comparing clustering with pairwise and relative constraints: A unified framework,” ACM Trans. Knowl. Discov. Data, vol. 11, no. 2, 2016.

[115] J. C. Nunnally and I. H. Bernstein, Psychometric Theory, Third Edition. McGraw-Hill, New York. 1994.

[116] I. Diaz-Valenzuela, V. Loia, M. J. Martin-Bautista, S. Senatore, and M. A. Vila, “Automatic constraints generation for semisupervised clustering: experiences with documents classification,” Soft Comput., vol. 20, no. 6, pp. 2329–2339, 2016.

[117] Y. Huang and T. M. Mitchell, “Exploring hierarchical user feedback in email clustering,” in AAAI Workshop - Technical Report, vol. WS-08-04, pp. 36–41, 2008.

[118] K. C. Duong , “Constrained clustering by constraint programming,” PhD Thesis, 2014.

[119] B. M. Nogueira, A. M. Jorge, and S. O. Rezende, “HCAC: Semi-supervised hierarchical clustering using confidence-based active learning,” in International Conference on Discovery Science, vol. 7569 LNAI, pp. 139–153, 2012.

[120] K. Bade and A. Nürnberger, “Personalized hierarchical clustering,” in Proceedings - IEEE/WIC/ACM International Conference on Web Intelligence (WI Main Conference Proceedings), pp. 181–187, 2007.

[121] Y. Hamasuna, Y. Endo, and S. Miyamoto, “Semi-supervised hierarchical clustering using clusterwise tolerance based pairwise constraints,” in International Conference on Modeling Decisions for

Artificial Intelligence, 2010, pp. 152-162, 2010. Springer, Berline, Heidelberg.

[122] P. Hansen and B. Jaumard, “Cluster Analysis and Mathematical Programming,” Math. Program., vol. 79, no. 1–3, pp. 191–215, 1997.

[123] J. G. Dy and C. E. Brodley, “Feature Selection for Unsupervised Learning,” J. Mach. Learn. Res., vol. 5, pp. 845–889, 2004.

[124] G. De Soete, “Optimal variable weighting for ultrametric and additive tree clustering,” Qual. Quant., vol. 20, no. 2–3, pp. 169–180, 1986.

[125] E. Y. Chan, W. K. Ching, M. K. Ng, and J. Z. Huang, “An optimization algorithm for clustering using weighted dissimilarity measures,” Pattern Recognit., vol. 37, no. 5, pp. 943–952, 2004.

[126] R. C. De Amorim, “Learning feature weights for K-Means clustering using the Minkowski metric,” Department of Computer Science and Information Systems Birkbeck, University of London, Doctoral Dissertation, 2011.

[127] A. E. Raftery and N. Dean, “Variable selection for model-based clustering,” J. Am. Stat. Assoc., vol. 101, no. 473, pp. 168–178, 2006.

[128] W. Ahmad and A. Narayanan, “Feature weighing for efficient clustering,” in 6th Intl. Conference on Advanced Information Management and Service, IMS, pp. 236–242, 2010.

[129] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, “Cluster validity methods: Part I,” SIGMOD Record, vol. 31, no. 2. pp. 40–45, 2002.

[130] K. Bade, M. Hermkes, and A. Nürnberger, “User oriented hierarchical information organization and retrieval,” in European Conference on Machine Learning , vol. 4701 LNAI, pp. 518-526, 2007, Springer, Berlin, Heidelberg.

[131] K. Daniels and C. Giraud-Carrier, “Learning the threshold in hierarchical agglomerative clustering,” in Proceedings - 5th International Conference on Machine Learning and Applications, ICMLA, pp. 270-278, 2006, IEEE.

[132] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J. M. Pérez, and I. Perona, “An extensive comparative study of cluster validity indices,” Pattern

Recognit., vol. 46, no. 1, pp. 243–256, 2013.

[133] T. Van Craenendonck and H. Blockeel, “Using internal validity measures to compare clustering algorithms,” AutoML Work. ICML 2015, pp. 1–8, 2015.

[134] T. Caliñski and J. Harabasz, “A Dendrite Method For Cluster Analysis,” Commun. Stat., vol. 3, no. 1, pp. 1–27, 1974.

[135] M. Friedman, “The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance,” J. Am. Stat. Assoc., vol. 32, no. 200, pp. 675–701, 1937.

[136] J. Demšar, “Statistical Comparisons of Classifiers over Multiple Data Sets,” J. Mach. Learn. Res., vol. 7, pp. 1–30, 2006.

[137] P. Nemenyi, “Distribution-free multiple comparisons.,” Dissertation Abstracts International, PhD Thesis, Princeton University, 1963.

[138] S. Holm, “A simple sequentially rejective multiple test approach,” Scand J Stat., vol. 6, no. 2, pp. 65–70, 1979.

[139] J. P. Shaffer, “Modified sequentially rejective multiple test procedures,” J. Am. Stat. Assoc., vol. 81, no. 395, pp. 826–831, 1986.

[140] J. Shaffer, “Multiple Hypothesis Testing,” Annu. Rev. Psychol., vol. 46, no. 1, pp. 561–584, Jan. 1995.

[141] H. B. Mann and D. R. Whitney, “On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other,” Ann. Math. Stat., vol. 18, no. 1, pp. 50–60, 1947.

[142] I. Davidson and S. S. Ravi, “Agglomerative hierarchical clustering with constraints: Theoretical and empirical results,” in European Conference on Principles of Data Mining and Knowledge Discovery, vol. 3721 LNAI, pp. 59–70, 2005.

[143] C. Perruchet, “Constrained agglomerative hierarchical classification,” Pattern Recognit., vol. 16, no. 2, pp. 213–217, 1983.

[144] K. L Wagstaff and C. Cardie, “Intelligent clustering with instance-level constraints,” 2002.

[145] D. T. Mihailović et al., “The choice of an appropriate information dissimilarity measure for hierarchical clustering of river streamflow time series, based on calculated Lyapunov exponent and Kolmogorov measures,” Entropy, vol. 21, no. 2, 2019.

[146] L. Ferreira and D. B. Hitchcock, “A comparison of hierarchical methods for clustering functional data,” Commun. Stat. Simul. Comput., vol. 38, no. 9, pp. 1925–1949, 2009.

[147] S. Hands and B. Everitt, “A Monte Carlo Study of the Recovery of Cluster Structure in Binary Data by Hierarchical Clustering Techniques,” Multivariate Behav. Res., vol. 22, no. 2, pp. 235–243, 1987.

[148] G. W. Milligan and M. C. Cooper, “A study of standardization of variables in cluster analysis,” J. Classif., vol. 5, no. 2, pp. 181–204, 1988.

[149] J. D. Bora and K. A. Gupta, “Effect of Different Distance Measures on the Performance of K-Means Algorithm: An Experimental Study in Matlab,” Int. J. Comput. Sci. Inf. Technol., vol. 5, no. 2, pp. 2501–2506, 2014.

[150] N. N. Mohammed and A. M. Abdulazeez, “Evaluation of partitioning around medoids algorithm with various distances on microarray data,” in Proceedings - IEEE International Conference on Internet of Things, IEEE Green Computing and Communications, IEEE Cyber, Physical and Social Computing, IEEE Smart Data, iThings-GreenCom-CPSCom-SmartData, pp. 1011–1016, 2018.

[151] I. Morlini and S. Zani, “Dissimilarity and similarity measures for comparing dendrograms and their applications,” Adv. Data Anal. Classif., vol. 6, no. 2, pp. 85–105, 2012.

[152] A. S. Shirkhorshidi, S. Aghabozorgi, and T. Ying Wah, “A Comparison study on similarity and dissimilarity measures in clustering continuous data,” PLoS One, vol. 10, no. 12, p. e0144059, 2015.

[153] A. J. Vakharia and U. Wemmerlöv, “A comparative investigation of hierarchical clustering techniques and dissimilarity measures applied to the cell formation problem,” J. Oper. Manag., vol. 13, no. 2, pp. 117–138, 1995.

[154] D. T. C. Lai and J. M. Garibaldi, “A comparison of distance-based semi-supervised fuzzy c-means clustering algorithms,” in IEEE International Conference on Fuzzy Systems, 2011, pp. 1580–1586.

[155] M. Lichman, UCI Machine Learning Repository [http://archive.ics.uci.edu/ml], 2013.

[156] D. Soria et al., “A methodology to identify consensus classes from clustering algorithms applied to immunohistochemical data from breast cancer patients,” Comput. Biol. Med., vol. 40, no. 3, pp. 318–330, 2010.

[157] R. Huang, Z. Zhang, and W. Lam, “Text clustering with limited user feedback under local metric learning,” in Asia Information Retrieval Symposium, vol. 4182 LNCS, pp. 132–144, 2006.

[158] Y. Huang and T. M. Mitchell, “Text clustering with extended user feedback,” in Proceedings of the Twenty-Ninth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 413–420, 2006.

[159] A. Dubey, I. Bhattacharya, and S. Godbole, “A cluster-level semi-supervision model for interactive clustering,” in Joint European Conference on Machine Learning and Knowledge Discovery in Databases,vol. 6321 LNAI, no. PART 1, pp. 409–424, 2010.

[160] T. F. Covões, E. R. Hruschka, and J. Ghosh, “A study of K-Means-based algorithms for constrained clustering,” Intell. Data Anal., vol. 17, no. 3, pp. 485–505, 2013.

[161] Y. Lei, D. Yu, Z. Bin, and Y. Yang, “Interactive K -Means Clustering Method Based on User Behavior for Different Analysis Target in Medicine,” Comput. Math. Methods Med., 2017.

[162] B. M. Nogueira, “Hierarchical semi-supervised confidence-based active clustering and its application to the extraction of topic hierarchies from document collections,” Universidade de São Paulo, Doctoral Dissertation, São Carlos, 2013.

[163] K. L. Wagstaff, “Value, cost, and sharing: Open issues in constrained clustering,” in International workshop on knowledge discovery in inductive databases,vol. 4747 LNCS, pp. 1–10, 2006.

[164] I. Davidson, K. L. Wagstaff, and S. Basu, “Measuring constraint-set utility for partitional clustering algorithms,” in European Conference on Principles of Data Mining and Knowledge Discovery, vol. 4213 LNAI, pp. 115–126, 2006.

[165] D. Greene and P. Cunningham, “Constraint selection by committee: An ensemble approach to identifying informative constraints for semi-supervised clustering,” in European Conference on Machine Learning, vol. 4701 LNAI, pp. 140-151, 2007.

[166] W. Atwa and K. Li, “Active Query Selection for Constraint-Based Clustering Algorithms,” in International Conference on Database and Expert Systems Applications, vol. 8644 LNCS, no. PART 1, pp. 438–445, 2014.

[167] L. Cai, T. Yu, T. He, L. Chen, and M. Lin, “Active Learning Method for Constraint-Based Clustering Algorithms,” in International Conference on Web-Age Information Management, 2016, vol. 9659, pp. 319-329, 2016.

[168] M. Ganji, “Semi-supervised Community Detection and Clustering,” PhD Thesis, 2017.

[169] M. Okabe and S. Yamada, “Clustering using boosted constrained k-Means Algorithm,” Front. Robot. AI, vol. 5, 2018.

[170] K. Bade and A. Nürnberger, “Creating a cluster hierarchy under constraints of a partially known hierarchy,” in Society for Industrial and Applied Mathematics - 8th SIAM International Conference on Data Mining, Proceedings in Applied Mathematics 130,vol. 1, pp. 13–24, 2008.

[171] G. Hang, D. Zhang, J. Ren, and C. Hu, “A hierarchical clustering algorithm based on K-means with constraints,” in 4th International Conference on Innovative Computing, Information and Control, ICICIC, pp. 1479–1482, 2009.

[172] A. D. Peterson, A. P. Ghosh, and R. Maitra, “Merging K-means with hierarchical clustering for identifying general-shaped groups,” Stat, vol. 7, no. 1, 2018.

[173] C. R. Lin and M. S. Chen, “A robust and efficient clustering algorithm based on cohesion self-merging,” in Proceedings of the ACM SIGKDD

International Conference on Knowledge Discovery and Data Mining, pp. 582–587, 2002.

[174] T. S. Chen , T. H. Tsai, Y. T. Chen, C. C. Lin, R. C. Chen, S. Y. Li, and H. Y. Chen, "A combined K-means and hierarchical clustering method for improving the clustering efficiency of microarray," in Proceedings of International Symposium on Intelligent Signal Processing and Communication Systems, ISPACS, pp. 405–408, 2005.

[175] M. S. Hasan and Z. H. Duan, "Hierarchical k-Means: A Hybrid Clustering Algorithm and Its Application to Study Gene Expression in Lung Adenocarcinoma," in Emerging Trends in Computational Biology, Bioinformatics, and Systems Biology: Algorithms and Software Tools, pp. 51–67, 2015.

[176] W. H. E. Day and H. Edelsbrunner, "Efficient algorithms for agglomerative hierarchical clustering methods," J. Classif., vol. 1, no. 1, pp. 7–24, 1984.

[177] D. Petrov, J. Che, B. Zhou, A. Santrosyan, Y. Zhou, A. Hadj Khodabakhshi, O. Tanaseichuk, and T. Jiang, "An Efficient Hierarchical Clustering Algorithm for Large Datasets," Austin J. Proteomics, Bioinforma., vol. 2, no. 1, pp. 1–6, 2015.

[178] W. Dong, J. D. Ren, and D. Zhang, "Hierarchical K-means clustering algorithm based on silhouette and entropy," in International Conference on Artificial Intelligence and Computational Intelligence, vol. 7002 LNAI, no. PART 1, pp. 339–347, 2011.

[179] R. Syal, G. V. S. R. Prasad, and V. V. Kumar, "A Novel Hybrid Clustering Algorithm : Integrated Partitional and Hierarchical Clustering Algorithm for Categorical Data," Int. J. Comput. Sci. Emerg. Technol., vol. 3, no. 5, pp. 138–146, 2012.

[180] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms, vol. 07-09-Janu, pp. 1027–1035, 2007.

[181] M. Z. Rodriguez, C. H. Comin, D. Casanova, O. M. Bruno, D. R. Amancio, L. D. F. Costa, and F. A. Rodrigues, " Clustering algorithms: A comparative approach," PLoS One, vol. 14, no. 1, Jan. 2019.

[182] Y. Liu, Z. Li, H. Xiong, X. Gao, and J. Wu, "Understanding of internal

clustering validation measures," in Proceedings - IEEE International Conference on Data Mining, ICDM, 2010, pp. 911–916.

[183] R. Riyaz and I. Rashid, "Efficient Framework for Finding Optimal Data Partitions for Hierarchical Centroid Algorithm," Int. J. Res. Appl. Sci. Eng. Technol., vol. 5, no. 11, pp. 2863–2871, 2017.

[184] G. W. Milligan and M. C. Cooper, "An examination of procedures for determining the number of clusters in a data set," Psychometrika, vol. 50, no. 2, pp. 159–179, 1985.

[185] H. Liu and H. Motoda, Computational Methods of Feature Selection. CRC Press, 2007.

[186] R. Bellman, Dynamic programming. Rand Corporation, Princeton University Press, 1957.

[187] W. S. DeSarbo, J. D. Carroll, L. A. Clark, and P. E. Green, "Synthesized clustering: A method for amalgamating alternative clustering bases with differential weighting of variables," Psychometrika, vol. 49, no. 1, pp. 57–78, 1984.

[188] J. Z. Huang, M. K. Ng, H. Rong, and Z. Li, "Automated variable weighting in k-means type clustering," IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, no. 5, pp. 657–668, 2005.

[189] G. De Soete, "OVWTRE: A program for optimal variable weighting for ultrametric and additive tree fitting," J. Classif., vol. 5, no. 1, pp. 101–104, 1988.

[190] V. Makarenkov and P. Legendre, "Optimal variable weighting for ultrametric and additive trees and K-means partitioning: Methods and software," J. Classif., vol. 18, no. 2, pp. 245–271, 2001.

[191] H. Frigui and R. Mahdi, "Semi-supervised clustering and feature discrimination with instance-level constraints," in IEEE International Conference on Fuzzy Systems, pp. 1–6, 2007.

[192] R. C. De Amorim, "Constrained clustering with Minkowski Weighted K-Means," in CINTI - 13th IEEE International Symposium on Computational Intelligence and Informatics, Proceedings, pp. 13–17, 2012.

Appendix-A

Datasets

There are twelve datasets in this study. Eleven of them were obtained from the UCI Machine Learning Repository [155]. The last one is the Nottingham Tenovus Breast Cancer (NTBC) dataset retrieved by Soria et al.[156]. The datasets are the works of various researchers, chiefly form the machine learning community, and gathered by the machine-learning group at the University of California, Irvin. They are described below.

Wine dataset. It is the most popular dataset in studies. Wine data set is data set composed of chemical analysis of wine grown in Italy and processed with different methods. It consists of 178 instances with 13 features, denoting three dissimilar classes of wines. Features are “Alcohol, Malic acid, Ash, Alcalinity of ash, Magnesium, Total phenols, Flavanoids, Nonflavanoid phenols, Proanthocyanins, Color intensity, Hue, OD280/OD315 of diluted wines and Proline”.

Iris dataset. This is the most widely utilised dataset in the literature. The dataset consists 3 classes of 50 instances each, where each class represents a type of iris plant which are “Setosa, Versicolor and Verginica”. Each instance is characterised by four uninterrupted attributes, namely, “Sepal Width, Sepal Length, Petal Width, and Petal Length”.

Zoo dataset. It contains 101 instances with 17 features, denoting seven dissimilar classes of animals. Features are “animal name, hair, feathers, eggs, milk, airborne, aquatic, predator, toothed, backbone, breathes, venomous, fins, legs, tail, domestic and catsize”. Each class is represented by number from 1 to 7 and animals in each class are listed below:

1- “(20) chicken, crow, dove, duck, flamingo, gull, hawk, kiwi, lark, ostrich, parakeet, penguin, pheasant, rhea, skimmer, skua, sparrow, swan, vulture, wren.
2- (41) aardvark, antelope, bear, boar, buffalo, calf, cavy, cheetah, deer, dolphin, elephant, fruitbat, giraffe, girl, goat, gorilla, hamster, hare, leopard, lion, lynx, mink, mole, mongoose, opossum, oryx, platypus, polecat, pony, porpoise, puma, pussycat, raccoon, reindeer, seal, sealion, squirrel, vampire, vole, wallaby,wolf.

3- (5) pitviper, seasnake, slowworm, tortoise, tuatara.
4- (10) clam, crab, crayfish, lobster, octopus, scorpion, seawasp, slug, starfish, worm.
5- (13) bass, carp, catfish, chub, dogfish, haddock, herring, pike, piranha, seahorse, sole, stingray, tuna.
6- (8) flea, gnat, honeybee, housefly, ladybird, moth, termite, wasp.
7- (4) frog, frog, newt, toad".

Pima Indians diabetes (diabetes). It comprises 768 instances with eight attributes. Pima Indian diabetes (Pima) 2-hour serum insulin (muU/ml) in feature number 5 was ejected owing to several missing values. Two classes represent the possible diagnostics (the patients present or do not present signs of diabetes). Attributes are " Pregnancies: **Number of times pregnant**, **Glucose**: Plasma glucose concentration a 2 hours in an oral glucose tolerance test, **Blood Pressure**: Diastolic blood pressure (mm Hg), **SkinThickness**: Triceps skin fold thickness (mm), **Insulin**: 2-Hour serum insulin (mu U/ml), **BMI**: Body mass index (weight in kg/(height in m)^2), **Diabetes Pedigree Function**: Diabetes pedigree function and **Age**"**.**

Cardiotocography (CTG) dataset. It comprises 2126 instances with 21 features, denoting three dissimilar classes representing "normal, suspicious and pathologic fetal state". Features are " **LB - FHR** baseline (beats per minute), **AC** - of accelerations per second, **FM** - of fetal movements per second, **UC** - of uterine contractions per second, **DL** - of light decelerations per second, **DS** - of severe decelerations per second, **DP** - of prolongued decelerations per second, **ASTV** - percentage of time with abnormal short term variability, **MSTV** - mean value of short term variability, **ALTV** - percentage of time with abnormal long term variability, **MLTV** - mean value of long term variability, **Width** - width of FHR histogram, **Min** - minimum of FHR histogram, **Max** - Maximum of FHR histogram, **Nmax** - of histogram peaks, **Nzeros** - of histogram zeros, **Mode** - histogram modem, **Mean** - histogram mean, **Median** - histogram median, **Variance** - histogram variance, **Tendency** - histogram tendency".

Breast Cancer Wisconsin Original (BCWO) dataset. It consists of 699 instances. However, the 16 instances have ejected owing to missing values. Each instance is characterised by 10 properties that denote aspects such as "code number: id number, Clump Thickness, Uniformity of Cell Size, Uniformity of Cell Shape, Marginal Adhesion, Single Epithelial Cell Size, Bare Nuclei, Bland Chromatin, Normal Nucleoli and Mitoses”. There are two classes, which indicate whether the tumour is “malignant or benign”.

Breast Cancer Wisconsin Diagnostic (BCWD) dataset. It consists of 569 instances. Each instance is characterised by 32 properties which are “ **ID** number, **radius** (mean of distances from center to points on the perimeter) , **texture** (standard deviation of gray-scale values) , **perimeter** (mean size of the core tumor) , **area**, **smoothnes**s (mean of local variation in radius lengths), **compactness** (mean of perimeter^2/area-1.0), **concavity** (mean of severity of concave portions of the contour), **concave points** (mean for number of concave portions of the contour), **symmetry**, **fractal_dimension** (mean for "coastline approximation" – 1) , **radius_se** (standard error for the mean of distances from center to points on the perimeter), **texture_se** (standard error for standard deviation of gray-scale values), **perimeter_se**, **area_se**, **smoothness_se** (standard error for local variation in radius lengths), **compactness_se** (standard error for perimeter^2 / area - 1.0), **concavity_se** (standard error for severity of concave portions of the contour**), concave points_se** (standard error for number of concave portions of the contour), **symmetry_se**, **fractal_dimension_se** (standard error for "coastline approximation") , **radius_worst** ("worst" or " largest " mean value for mean of distances from center to points on the perimeter) , **texture_worst** ("worst" or " largest " mean value for standard deviation of gray-scale values), **perimeter_worst, area_worst**, **smoothness_worst**("worst" or "largest" mean value for local variation in radius lengths), **compactness_worst** ("worst" or " largest " mean value for perimeter^2 / area - 1.0), **concavity_worst** ("worst" or " largest" mean value for severity of concave portions of the contour) , **concave points_worst** ("worst" or " largest "mean value for number of concave portions of the contour), **symmetry_worst** and **fractal_dimension_worst** ("worst" or " largest " mean value for "coastline approximation"). There are two classes, which indicating whether the tumour is malignant or benign”.

Dermatology dataset. It comprises 366 instances with 33 attributes, denoting six classes “psoriasis, seboreic dermatitis, lichen planus, pityriasis rosea, cronic dermatitis, and pityriasis rubra pilaris”. One attribute (age) has discarded owing to missing values. Attributes are “erythema, scaling, definite borders, itching, koebner phenomenon, polygonal papules, follicular papules, oral mucosal involvement, knee and elbow involvement, scalp involvement, family history, Age (linear), melanin incontinence, eosinophil in the infiltrate, PNL infiltrate, fibrosis of the papillary dermis, exocytosis, acanthuses, hyperkeratosis, Para keratosis, clubbing of the rete ridges, elongation of the rete ridges, thinning of the suprapapillary epidermis, spongiform pustule, munro microabcess, focal hypergranulosis, disappearance of the granular layer, vacuolisation and damage of basal layer, spongiosis, saw-tooth appearance of retes, follicular horn plug, perifollicular parakeratosis, inflammatory monoluclear infiltrate and band-like infiltrate".

Mammographic dataset. It consists of 961 instances, but the 131 instances have discarded owing to missing values. Each instance is characterised by 6 properties and two classes (which are benign and malignant). Properties are "**BI-RADS assessment**: 1 to 5 (ordinal, non-predictive!), **Age**: patient's age in years (integer), **Shape**: mass shape: round=1 oval=2 lobular=3 irregular=4 (nominal), **Margin**: mass margin: circumscribed=1 microlobulated=2 obscured=3 ill-defined=4 spiculated=5 (nominal) and **Density**: mass density high=1 iso=2 low=3 fat-containing=4 (ordinal)".

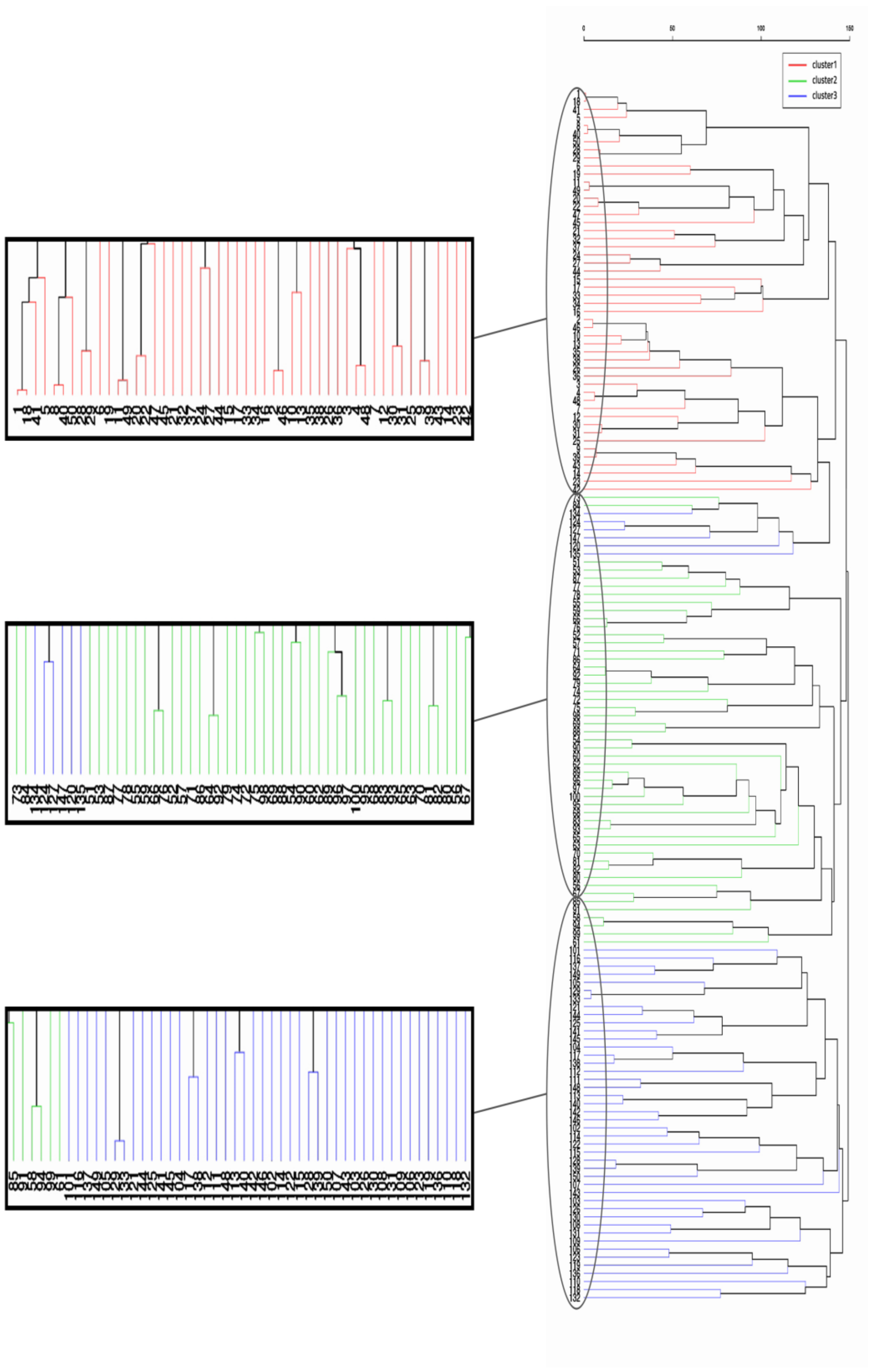
Banknote Authentication dataset. It contains 1372 instances with 5 properties, indicating two classes. Properties are "variance of Wavelet Transformed image, skewness of Wavelet Transformed image, curtosis of Wavelet Transformed image, entropy of image and class (integer) with two values: 0 (false) or 1 (true)".

Ionosphere dataset. It comprises 351 instances with 34 properties, indicating two classes and 34 attributes. Attributes are a0, a1 up to a32. Classes are types of radars having values "good" and "bad".

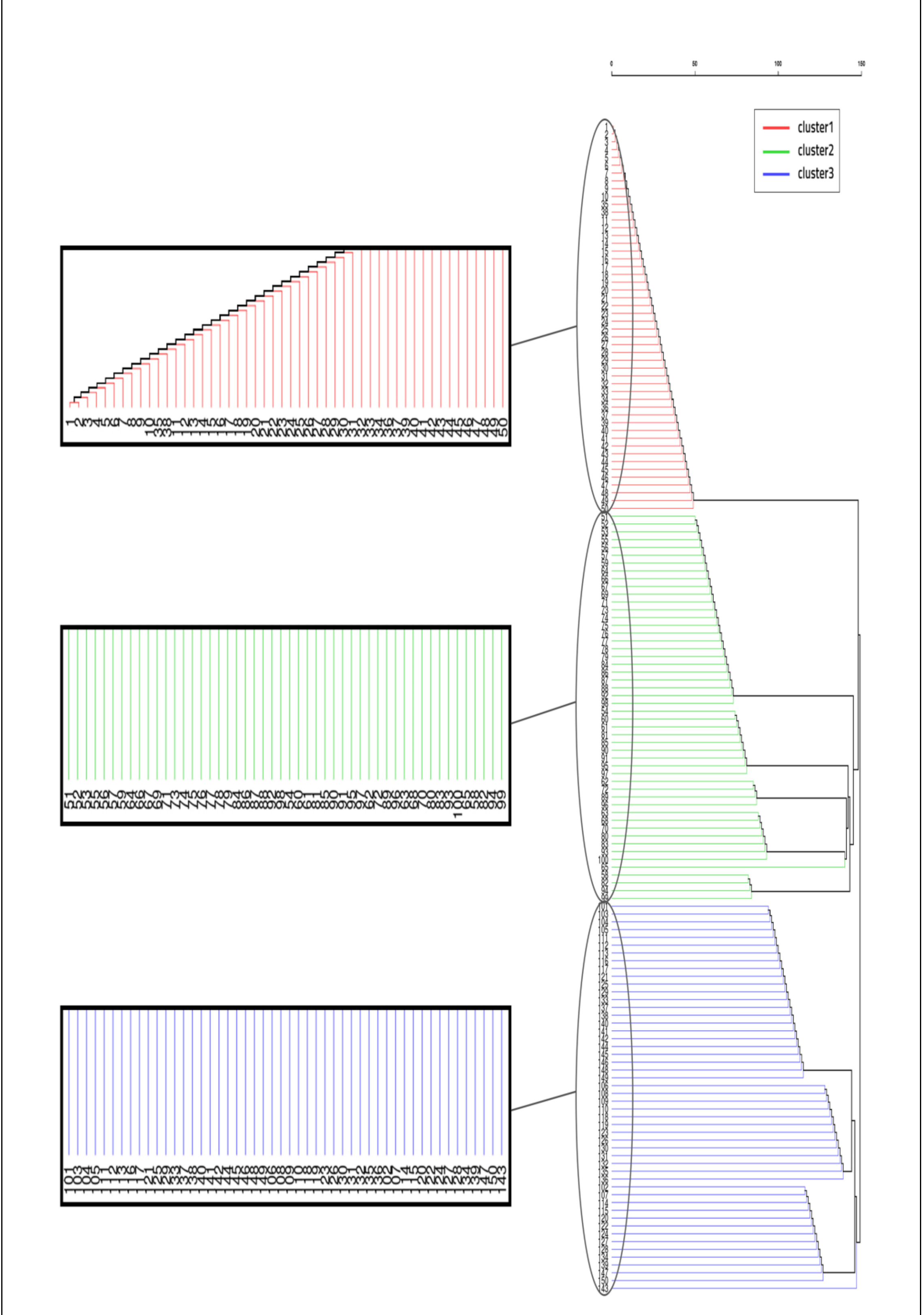
Nottingham Tenovus Breast Cancer (NTBC) dataset. It comprises 663 instances with 25 properties, indicating six classes. Features are" CK7/8, CK 18, CK 19, CK 5/6 , CK 14, Actin, p63, ER , PgR , AR ,EGFR , HER2 , HER3 , HER4, p53,nBRCA1 , FHIT , E-cad, P-cad ,MUC1 , MUC1co , MUC2 ,GCDFP , Chromo and Synapto . And classes are Luminal A, Luminal N, Luminal B, Basal - p53 altered, Basal - p53 normal and HER2".

Appendix-B

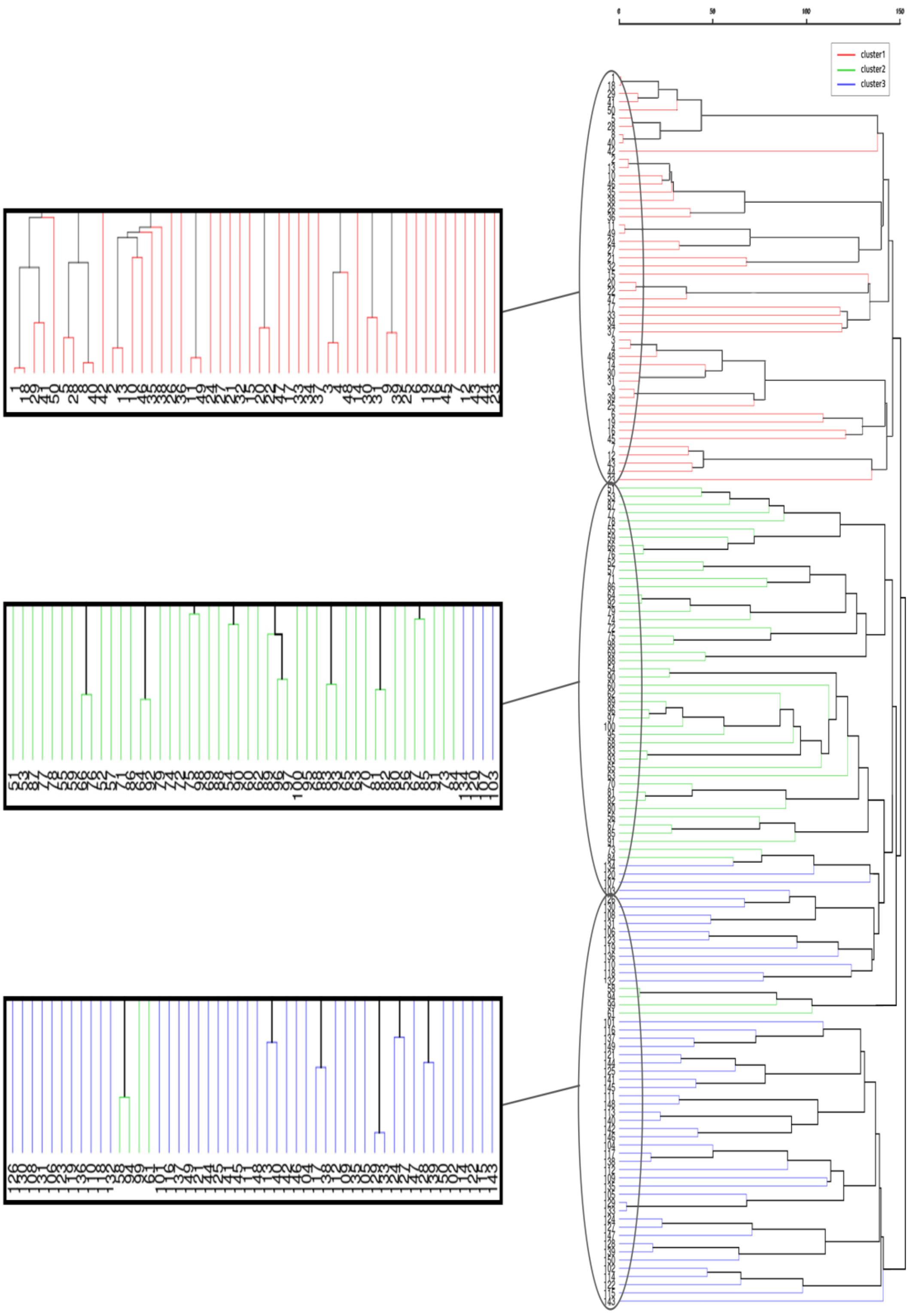
Iris CWHAC-IP



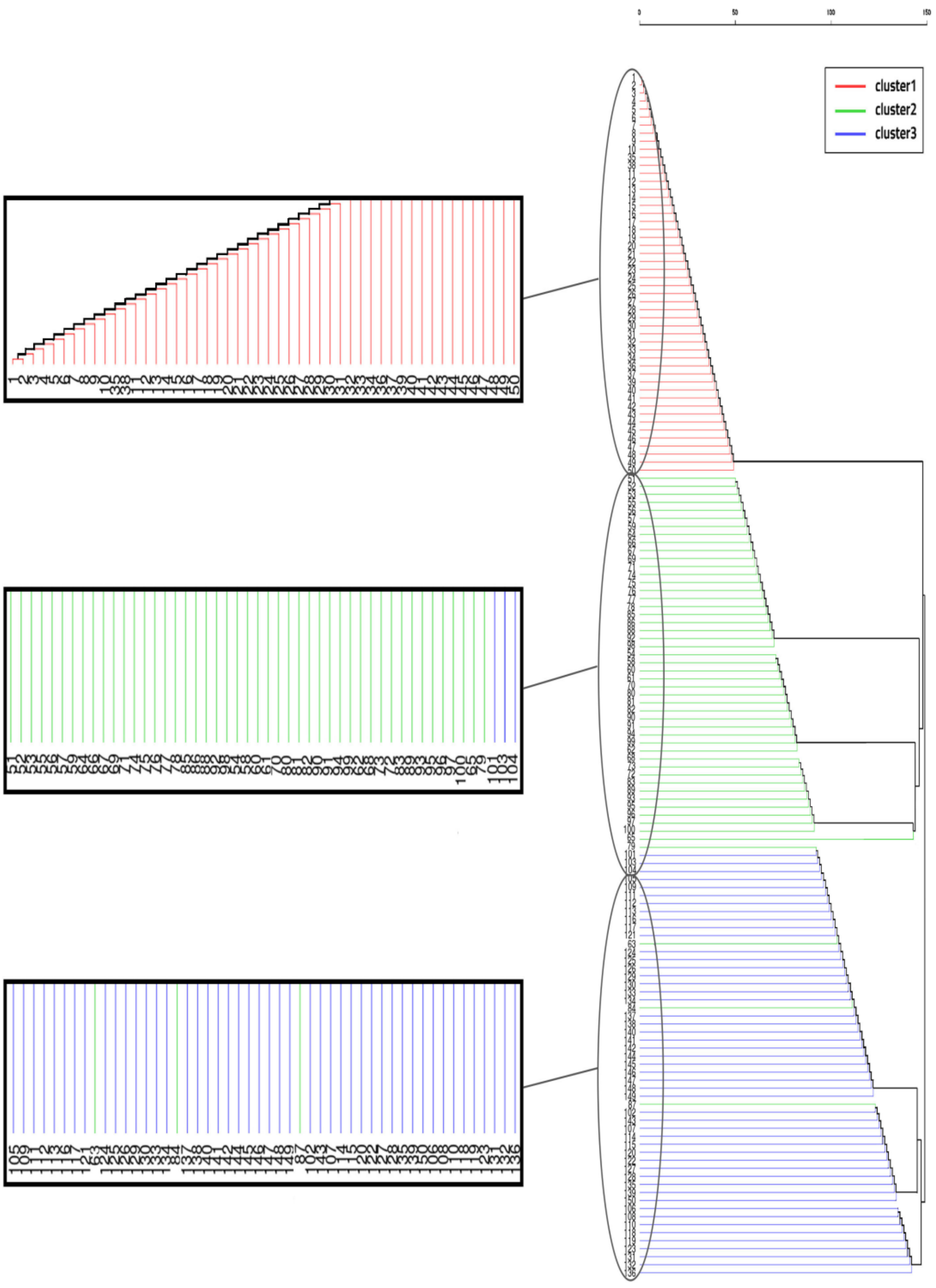
Iris CWHC-IKM-IP



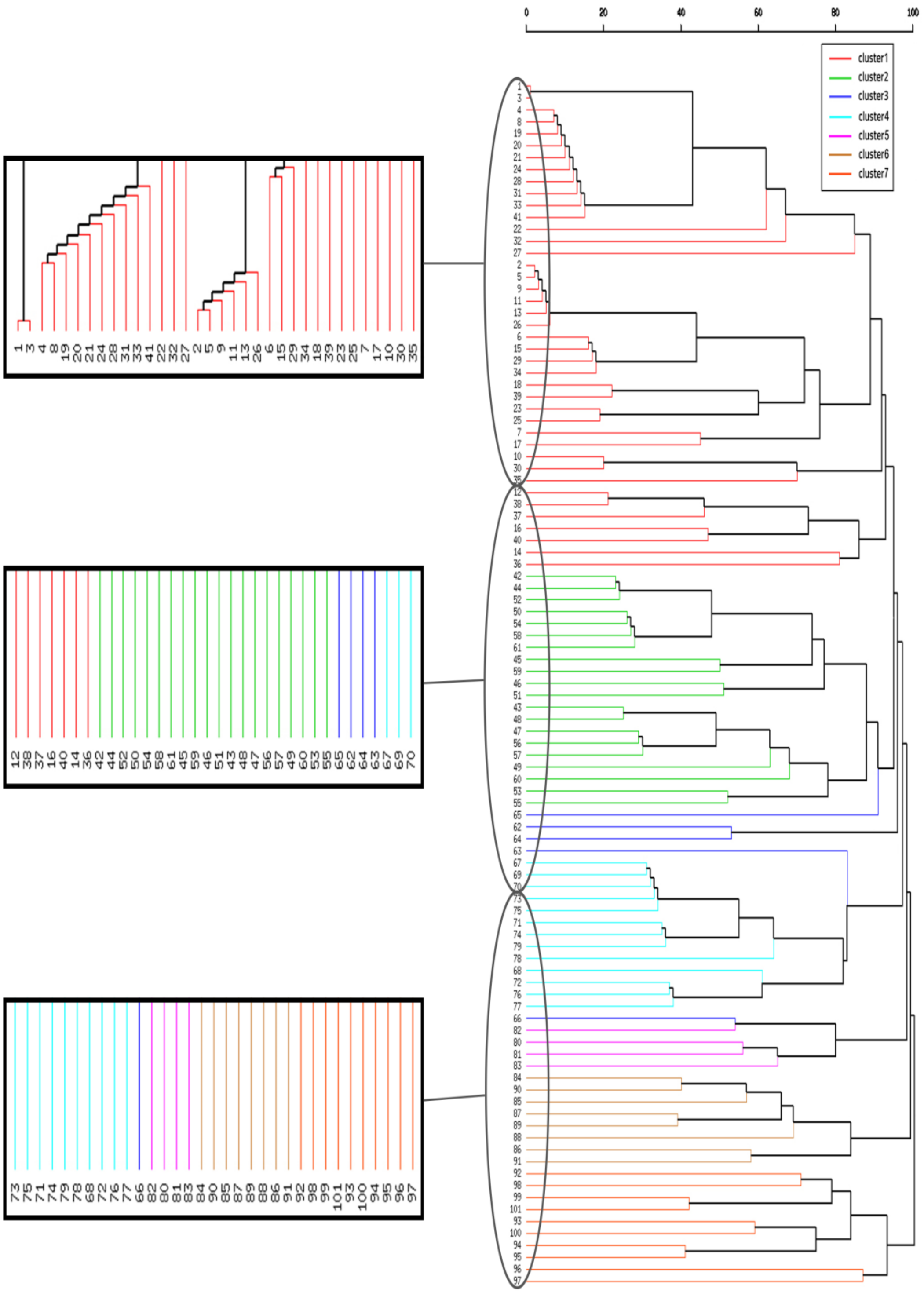
Iris CWHAC-UT



Iris CWHC-IKM-UT

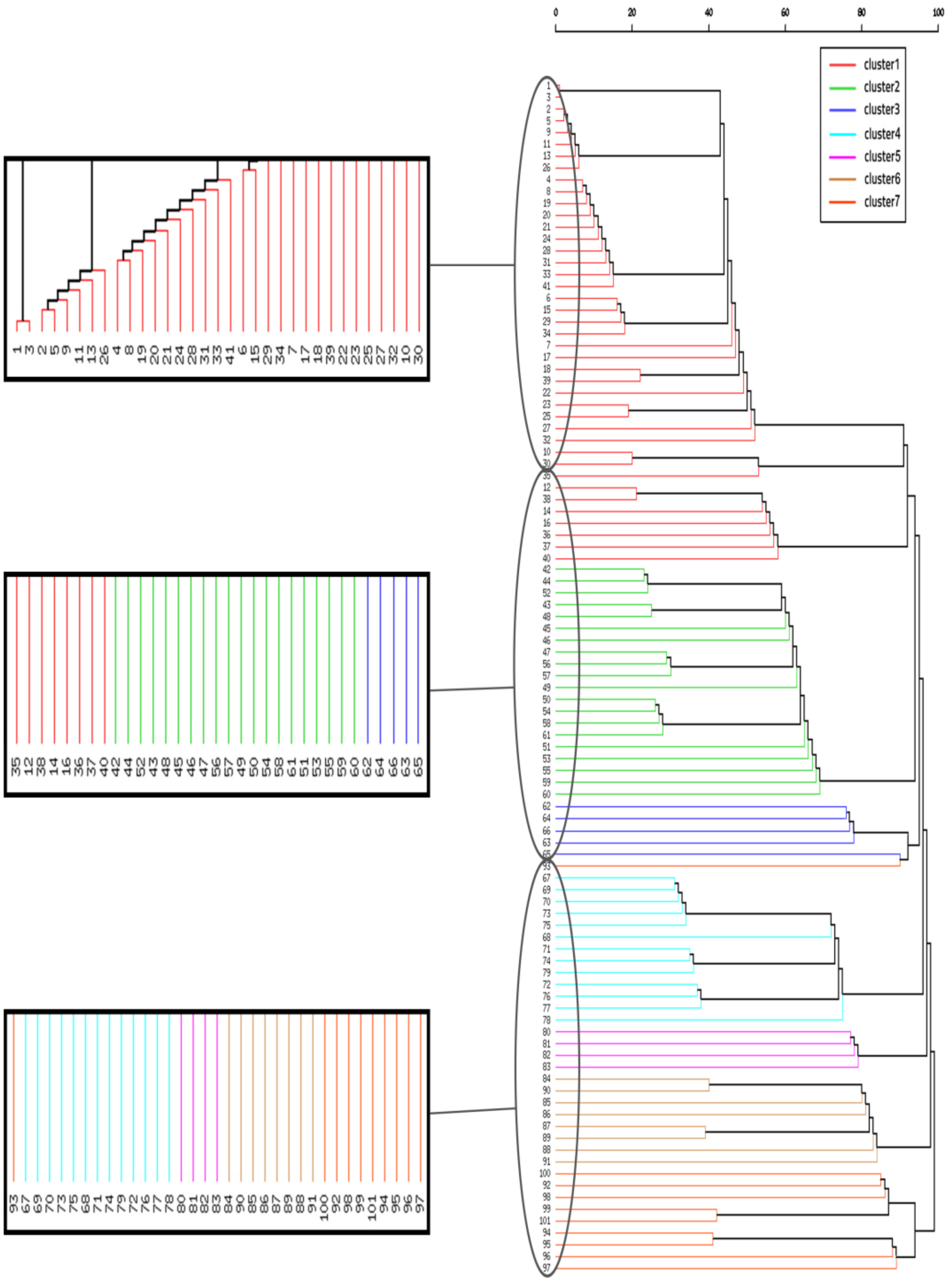


Zoo CWHAC-IP

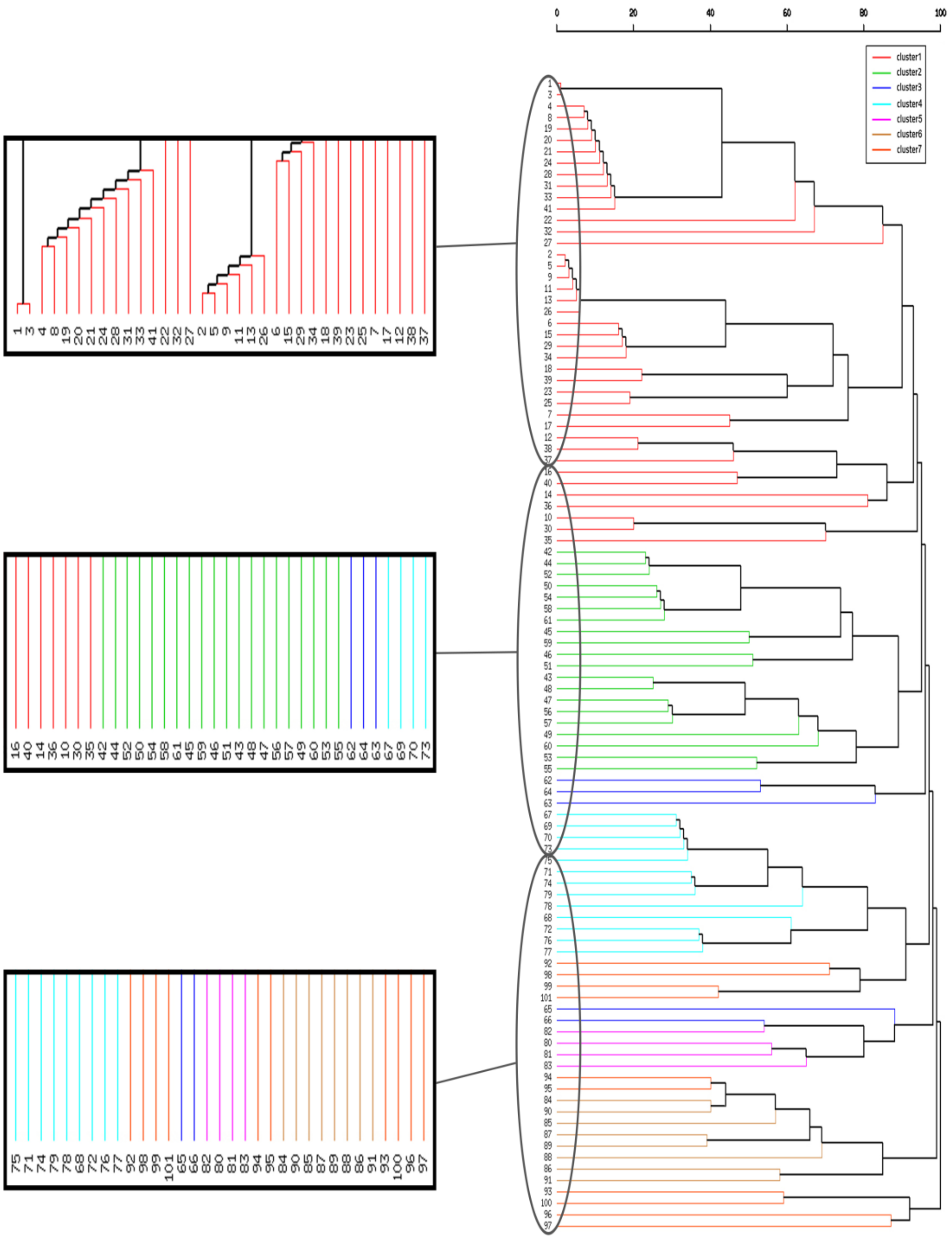


0
20
40
60
80
100
cluster1
cluster2
cluster3
cluster4
cluster5
cluster6
cluster7

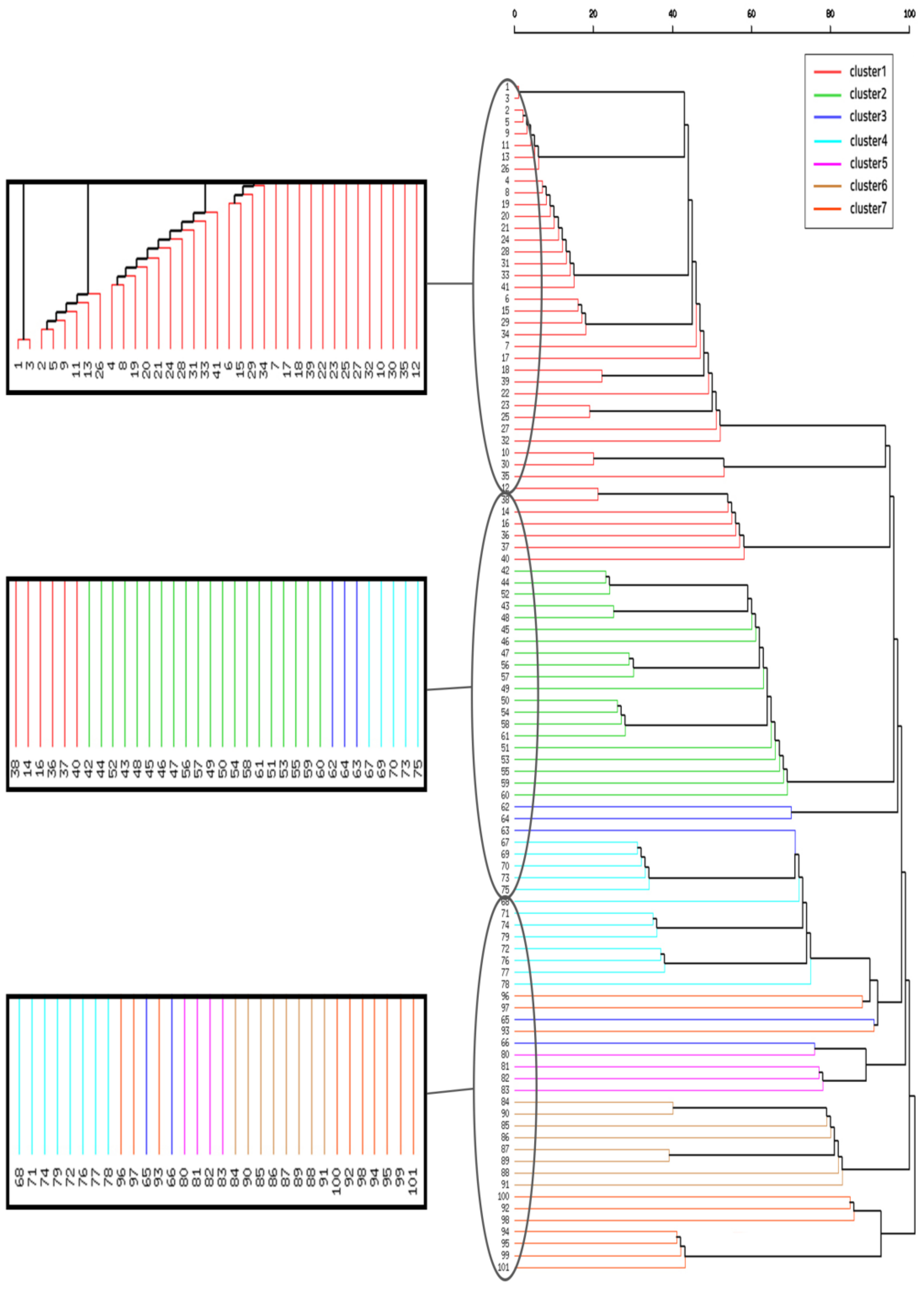
Zoo-CWHC-IKM-IP



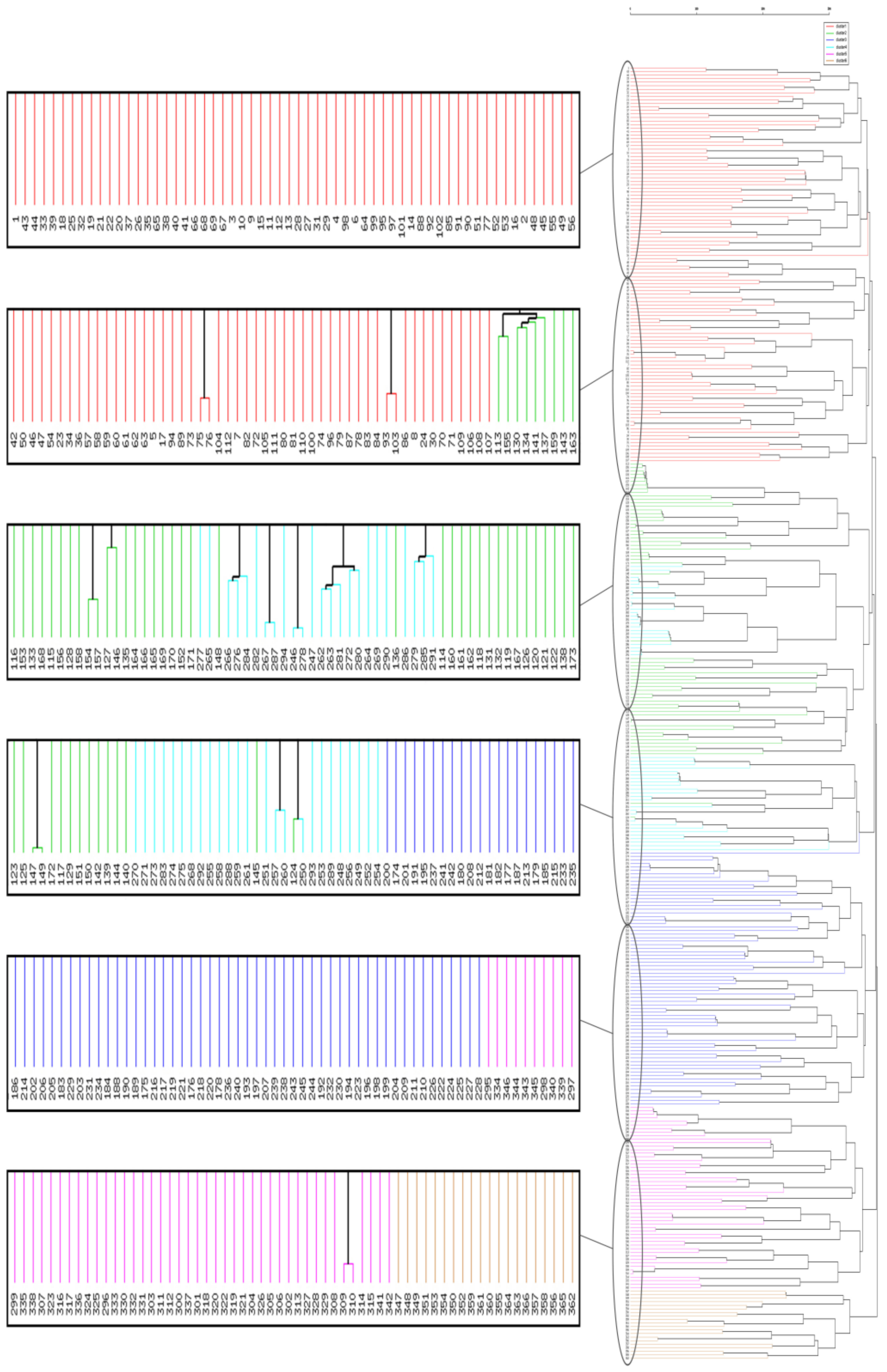
Zoo CWHAC-UT


0
20
40
60
80
100
cluster1
cluster2
cluster3
cluster4
cluster5
cluster6
cluster7

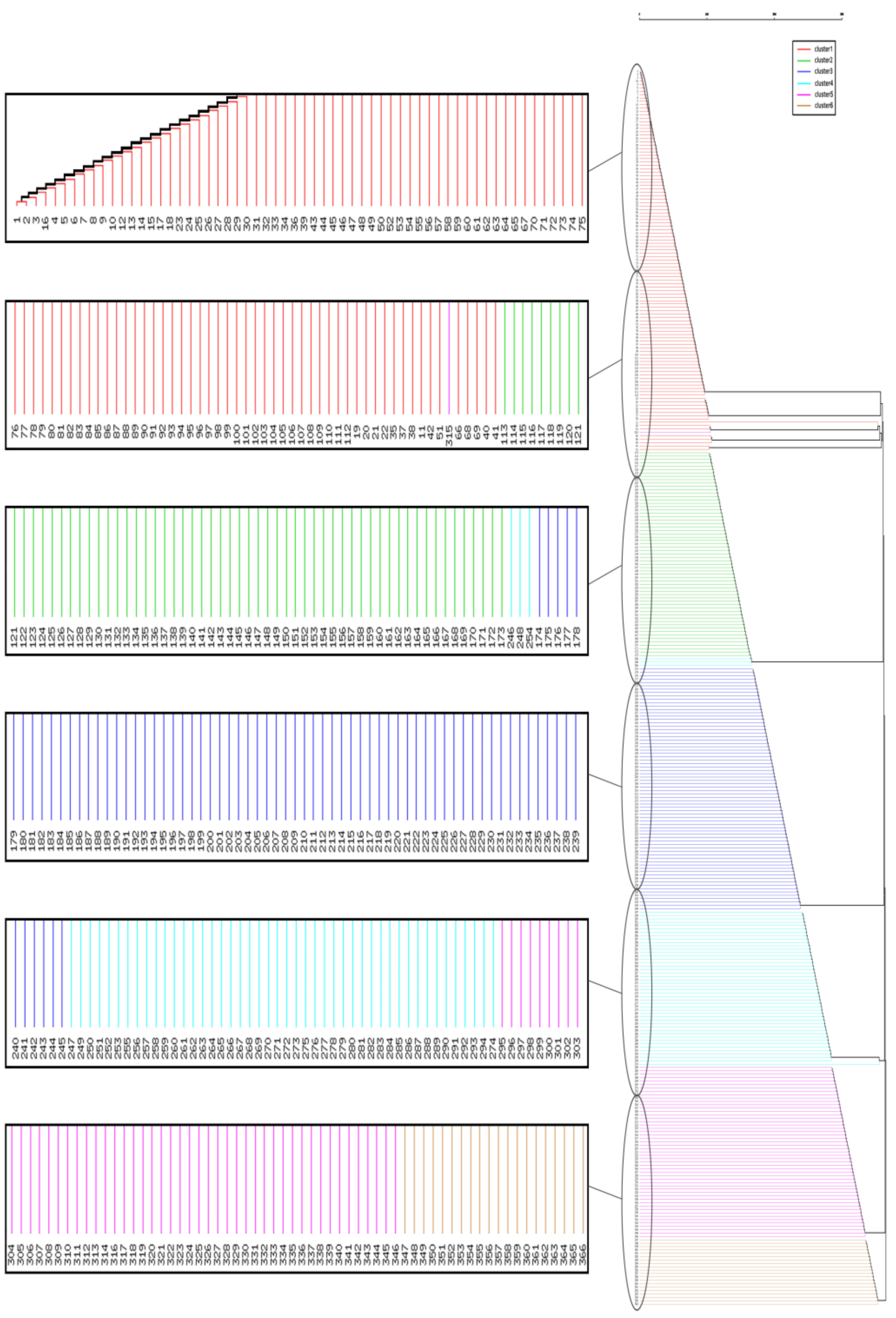
Zoo-CWHC-IKM-UT



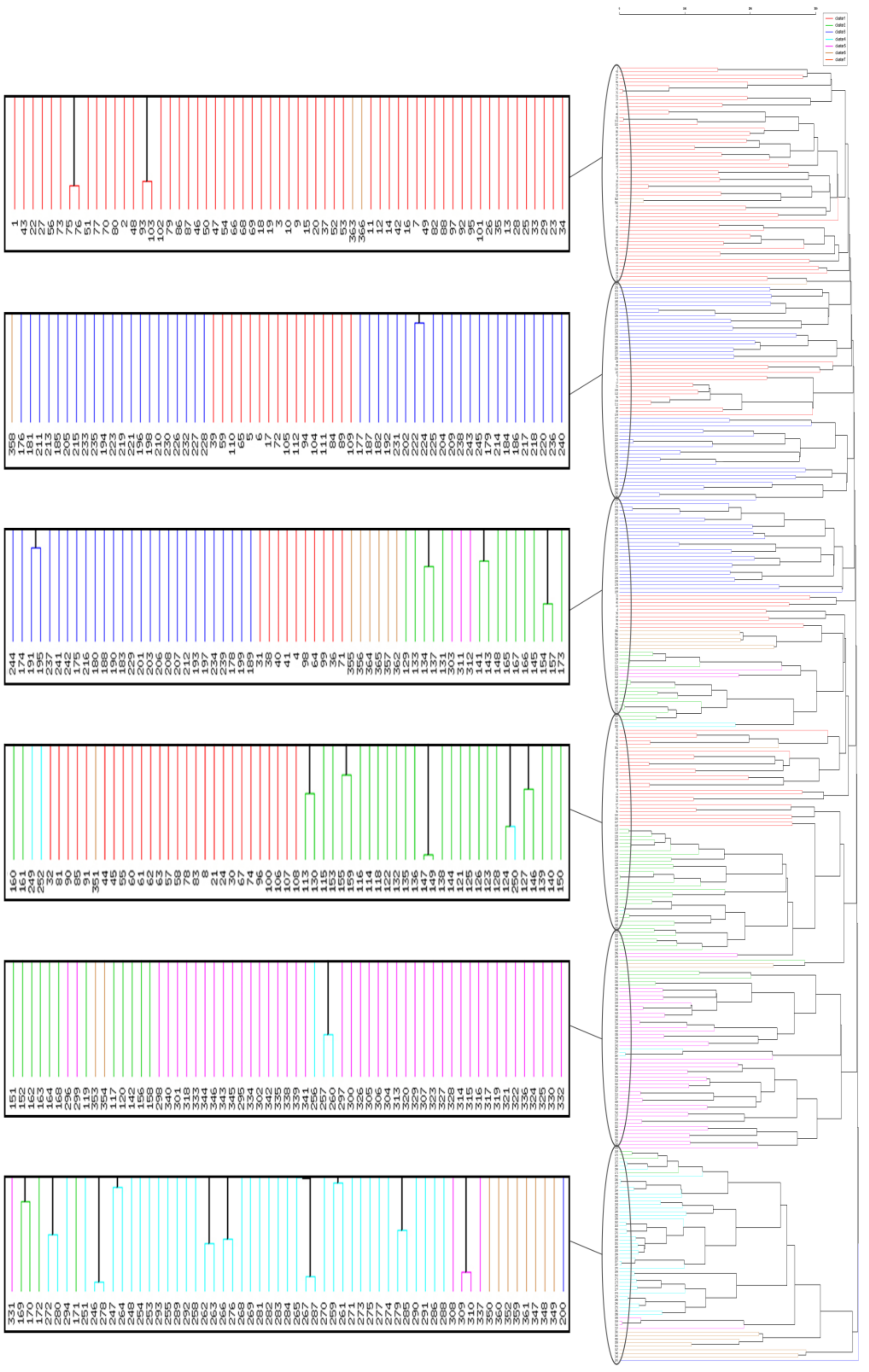
Dermatology CWHAC-IP



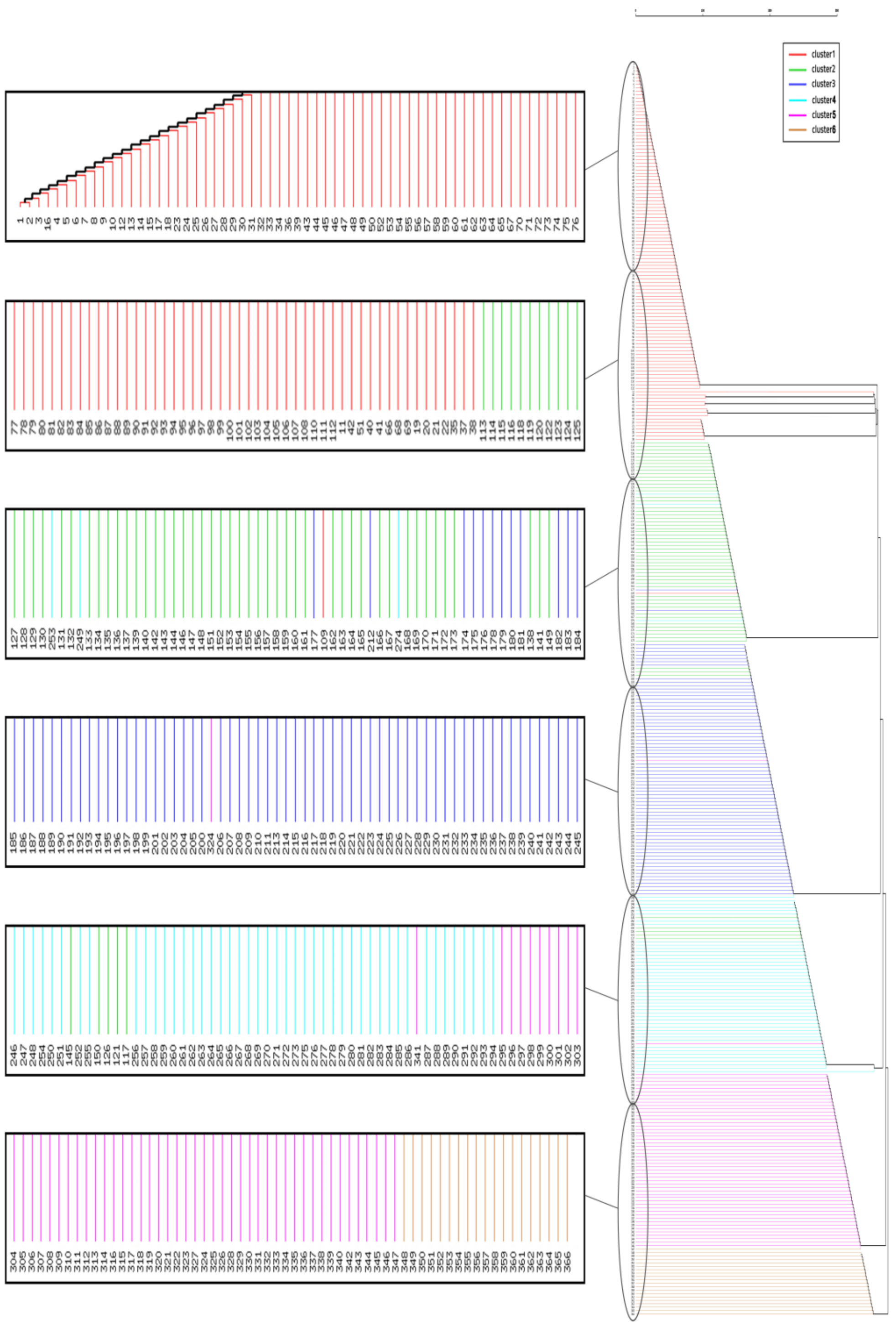
Dermatology CWHC-IK-IP



Dermatology CWHAC-UT



Dermatology CWHC-IK-UT



Appendix-C

This research has resulted three papers which has been submitted /published as conference and journal papers. The details could be found in blew sections:

Conference Published

1- Aljohani, A., Lai, D.T.C., Bell, P.C. and Edirisinghe, E.A., 2017, August. A Comparison of Distance Metrics in Semi-supervised Hierarchical Clustering Methods. In *International Conference on Intelligent Computing* (pp. 719-731). Springer, Cham.

2- Aljohani, A.A., Edirisinghe, E.A. and Lai, D.T.C., 2019, September. An Effective and Efficient Constrained Ward's Hierarchical Agglomerative Clustering Method. In *Proceedings of SAI Intelligent Systems Conference* (pp. 590-611). Springer, Cham.

Conference Paper will be Submitted

Aljohani, A.A. and Edirisinghe, E.A. 2019. Learning feature weights for Semi-supervised Hierarchical Clustering Methods.

Journal Paper will be Submitted

Aljohani, A.A. and Edirisinghe, E.A. 2019. Constrained Ward's Hierarchical Clustering Method using the two initial cluster setting strategies:Agglomerative and intelligent K-means.