Novel Feature Selection, Scalable Feature Extraction, and Clustering Algorithms for Plant Genome Sequences

Ph.D. Thesis

By

Rajesh Dwivedi

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

May 2024

Novel Feature Selection, Scalable Feature Extraction, and Clustering Algorithms for Plant Genome Sequences

A THESIS

submitted to the

INDIAN INSTITUTE OF TECHNOLOGY INDORE

in partial fulfillment of the requirements for

the award of the degree

of

DOCTOR OF PHILOSOPHY

By

Rajesh Dwivedi

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

May 2024

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **Novel Feature Selection, Scalable Feature Extraction, and Clustering Algorithms for Plant Genome Sequences** in the partial fulfillment of the requirements for the award of the degree of **Doctor of Philosophy** and submitted in the **Department of Computer Science and Engineering, Indian Institute of Technology Indore,** is an authentic record of my own work carried out during the time period from May 2021 to May 2024 under the supervision of Dr. Aruna Tiwari, Professor, Indian Institute of Technology Indore, India, Dr. Milind B. Ratnaparkhe, Principal Scientist (Biotechnology), ICAR-Indian Institute of Soybean Research (ICAR-IISR), Indore, India, and Dr. Neha Bharill, Associate Professor, Department of Computer Science and Engineering, Mahindra University, Ecole Centrale School of Engineering (MEC), Hyderabad, India.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other institute.

1/5/2024
Signature of the Student with Date
(Rajesh Dwivedi)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

1/5/2024
Signature of Thesis Supervisor with Date
(Dr. Aruna Tiwari)

Signature of Thesis Supervisor with Date 1/5/2024
(Dr. Milind Ratnaparkhe)

Signature of Thesis Supervisor with Date 1/5/2024
(Dr. Neha Bharill)

Rajesh Dwivedi has successfully given her Ph.D. Oral Examination held on

_____

Signature of Chairperson, OEB Date:	Signature of External Examiner Date:	Signature of Thesis Supervisor #1 Date:
Signature of Thesis Supervisor #2 Date:	Signature of Thesis Supervisor #3 Date:	Signature of PSPC Member #1 Date:
Signature of PSPC Member #2 Date:	Signature of Convener, DPGC Date:	Signature of Head of Discipline Date:

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my heartfelt gratitude to a number of persons who in one or the other way contributed by making this time as learnable, enjoyable, and bearable. At first, I would like to thank my supervisors **Prof. Aruna Tiwari**, **Dr. Milind Ratnaparkhe**, and **Dr. Neha Bharill** who was a constant source of inspiration during my work. Without their constant guidance and research directions, this research work could not be completed. Their continuous support and encouragement has motivated me to remain streamlined in my research work.

I am thankful to **Prof. Kapil Ahuja** and **Dr. Swaminathan Ramabadran**, my research committee member for taking out some valuable time to evaluate my progress all these years. Their good comments and suggestions helped me to improve my work at various stages. I am also grateful to **Dr. Ranveer Singh**, HOD of Computer Science and Engineering for his help and support.

My sincere acknowledgement and respect to **Prof. Suhas Joshi** , Director, Indian Institute of Technology Indore for providing me the opportunity to explore my research capabilities at Indian Institute of Technology Indore.

I extend my sincere thanks to the **Council of Scientific and Industrial Research (CSIR)** for funding the PhD research. This work was supported by the CSIR under Grant 22(0853)/20/EMR-II in collaboration with **ICAR-Indian Institute Of Soybean Research, Indore** and **Mahindra University, Hyderabad**.

I would like to appreciate the fine company of my dearest colleagues and friends especially, Dr. Preeti Jha, Abhishek Tripathi, and Achint kumar kansal. I am thankful to undergraduate students (Parul Mogre and Pranjal Gadge) who have also supported me in my research work. I am also grateful to the institute staffs for their unfailing support and assistance.

I would like to express my heartfelt respect to my parents for their love, care and support they have provided to me throughout my life. Special thanks to my wife (Pragya), my sisters (Shalini and Ragini), and friends as this thesis would not have been possible without the help of their support and encouragements. I also want to

thank my in-laws for their support and blessings.

Finally, I am thankful to all who directly or indirectly contributed, helped and supported me. To sign off, I write a quote by Albert Einstein:

"The only source of knowledge is experience." –Albert Einstein

Rajesh Dwivedi

To my family and friends

Abstract

Feature selection, extraction, and clustering techniques play crucial roles in various fields such as healthcare, bioinformatics, finance, and environment monitoring etc. Despite their potential to address significant issues like the time-consuming nature of alignment-based methodologies, their application in plant genomics remains limited. To overcome these challenges, researchers are delving into machine learning and parallel computing techniques. Clustering emerges as a promising approach, aiding in organizing and categorizing genomic sequences to uncover patterns and functional elements. However, the large-scale and high-dimensional nature of data in plant genomics complicates the clustering process and reduces accuracy. Additionally, the time-consuming nature of alignment-based approaches presents obstacles to real-time analysis, particularly in applications like clinical diagnostics. This necessitates the development of efficient dimensionality reduction methods such as feature selection, scalable feature extraction methods using Big Data frameworks, and incremental clustering techniques to enhance computational efficiency and model efficacy.

In this thesis work, we proposed to develop novel feature selection, scalable alignment-free feature extraction, and incremental clustering algorithms to address plant genomics challenges. First, we proposed a novel clustering-based hybrid feature selection approach using Ant Colony Optimization to address the dimensionality problem in plant genomics data. This method selects features randomly and evaluates their quality using K-means clustering based on the Silhouette Index and Laplacian score. By allowing random feature selection, it enables better exploration of the feature space, avoiding local optima and generating global optimal solutions. The proposed feature selection approach demonstrates excellent performance when evaluated on the benchmark datasets. However, to evaluate its effectiveness on plant genome data, we collected the massive real-life plant genome data from Indian Council of Agricultural Research-Indian Institute of Soybean Research (ICAR-IISR), Indore, India.

To test the proposed feature selection approach on the huge genome data there is a need to transform genomics sequences into feature vectors. So, we introduced two

scalable feature extraction approaches using Apache Spark. The first approach employs a 14-dimensional scalable feature extraction method, which captures sequence length, nucleotide base frequency, pattern organization, distribution, and entropy to generate fixed-length numeric vectors. This method efficiently extracts context-based features and mitigates the extraction of identical features for dissimilar sequences. The features extracted through this approach are clustered using K-means and Fuzzy c-means, after that the quality of the resulting clusters is measured to assess the effectiveness of this approach. However, this method does not extract features based on the chemical properties of nucleotides, which would provide important information about genome functionality. So, to further improve the performance, we proposed another 13-dimensional scalable feature extraction method that extracts the significantly important features based on the classification of nucleotides using their chemical properties in terms of entropy and the length of the sequence. The performance of this approach is evaluated by providing the extracted feature vector as input to the K-means clustering and results are evaluated in terms of cluster quality. However, the clustering methods (K-means and Fuzzy c-means) used in these methods are not able to handle the real-time dynamic data, which is required for rapid processing of plant genomics data.

To handle the real-time dynamic data, we proposed a multi-objective incremental clustering method for processing dynamic data that generates and updates clusters in real-time. To improve the dynamic clustering process, the proposed method employed Euclidean distance to calculate the similarity between data points and constructs a fitness function with three primary clustering objective functions: inter-cluster distance, intra-cluster distance, and cluster density. The proposed method employed the concept of objective weighting, which allocates weight to each objective to generate a single Pareto-optimal solution for the constructed fitness function. The proposed incremental clustering performs well when tested on the benchmark datasets. Finally, we investigated the real-life plant genomics and protein data obtained from ICAR-IISR, Indore, using our developed feature selection and incremental clustering approaches.

List of Publications

A. Published

A1. In Refereed Journals

1. **Rajesh Dwivedi**, Aruna Tiwari, Neha Bharill, and Milind Ratnaparkhe. *A Novel Clustering-Based Hybrid Feature Selection Approach Using Ant Colony Optimization*, Arabian Journal for Science and Engineering, (Springer), vol. 48, pp. 10727-10744, 2023, DOI= 10.1007/s13369-023-07719-7. (**IF:2.9**)

2. **Rajesh Dwivedi**, Aruna Tiwari, Neha Bharill, Milind Ratnaparkhe, Parul Mogre, Pranjal Gadge, and Kethavath Jagadeesh. *A novel apache spark-based 14-dimensional scalable feature extraction approach for the clustering of genomics data*, The Journal of Supercomputing, (Springer), vol. 80, pp. 3554-3588, 2023, DOI=10.1007/s11227-023-05602-8. (**IF:3.3**)

3. **Rajesh Dwivedi**, Aruna Tiwari, Neha Bharill, Milind Ratnaparkhe, Rishabh Soni, Rahul Mahbubani and Saket Kumar. *An incremental clustering method based on multiple objectives for dynamic data analysis*, Multimedia Tools and Applications, (Springer), vol. 83, pp. 38145-38165, 2023, DOI=10.1007/s11042-023-17134-7. (**IF:3.6**)

A2. In Refereed Conferences

1. **Rajesh Dwivedi**, Aruna Tiwari, Neha Bharill, and Milind Ratnaparkhe. *A Hybrid Feature Selection Approach for Data Clustering Based on Ant Colony Optimization*, In International Conference on Neural Information Processing (ICONIP) (Springer), Singapore, pp. 659-670, November, 22-26, 2022, DOI= https://doi.org/10.1007/978-3-031-30111-7_55.

2. **Rajesh Dwivedi**, Aruna Tiwari, Neha Bharill, Milind Ratnaparkhe, Abhishek Tripathi and Preeti Jha *A Novel Feature Extraction Approach for the Clustering and Classification of Genome Sequences*, In 2023 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 1018-1023, Mexico City, Mexico, North America, December, 5-8, 2023. DOI= 10.1109/SSCI52147.2023.10372047 (**Received travel grant and financial support from IEEE Computational Intelligence Society and Science and Engineering Research Board (SERB) under the Department of Science and Technology (DST), Govt. of India**)

3. **Rajesh Dwivedi**, Aruna Tiwari, Neha Bharill, Milind Ratnaparkhe, *A Novel Feature Extraction Technique for The Clustering of SNP Data*, In International Conference on Vegetable Oils 2023 (ICVO 2023), pp. 223-224, Hyderabad, India, January 17-21, 2023.

B. Communicated

In Refereed Journals

1. **Rajesh Dwivedi**, Aruna Tiwari, Neha Bharill, and Milind Ratnaparkhe *Unsupervised Feature Selection Methods: A Taxonomy of Approaches Including Pros, Cons, and Challenges*, The Journal of Supercomputing, (Springer), 2023 (**IF:3.3**)

2. **Rajesh Dwivedi**, Aruna Tiwari, Neha Bharill, Milind Ratnaparkhe, Saurabh Kumar Singh, Prashant Kumar, Ayush Kumar. *An Alignment-Free Scalable Feature Extraction Method for Genomic Data Clustering*, Multimedia Tools and Applications, (Springer), 2023 (**IF:3.6**)

Contents

Abstract i

List of Publications iii

List of Figures xi

List of Tables xv

List of Abbreviations and Acronyms xix

1 Introduction 1

1.1 Motivation . . . 5

1.2 Objectives . . . 7

1.3 Thesis Contributions . . . 8

1.4 Organization of the Thesis . . . 12

2 Literature Survey 17

2.1 Feature Selection . . . 17

2.1.1 Development of Feature Selection . . . 20

2.1.2 Types of Feature Selection . . . 23

2.2 Feature Extraction Approaches . . . 31

2.3 Incremental Clustering for Dynamic Data Analysis . . . 34

2.4 Big Data Processing Framework . . . 38

2.5 Performance Evaluation Measures . . . 46

2.5.1 External Evaluation Measures . . . 46

2.5.2 Internal Evaluation Measures . . . 47

2.6 Real-life Genome and Protein Data 49
2.6.1 Wheat, Rice, and Soybean Genome Dataset Description 51
2.6.2 Rice SNP Dataset Description 52
2.6.3 Soybean Protein Dataset Description 53

3 A Novel Clustering-Based Hybrid Feature Selection Approach using Ant Colony Optimization 55

3.1 Introduction 55
3.2 Proposed Clustering-Based Hybrid Feature Selection Approach using Ant Colony Optimization 57
3.2.1 Preliminaries 57
3.2.2 Proposed Method 58
3.3 Experimental Evaluation 61
3.3.1 Dataset Details 62
3.3.2 Evaluation Measures 62
3.3.3 Hyperparameter Settings for the Proposed Approach 63
3.3.4 Experimental Analysis 63
3.3.5 Comparative Performance Analysis 73
3.4 Summary 75

4 A Novel Apache Spark Based 14-Dimensional Scalable Feature Extraction Approach for the Clustering of Genomics Data 77

4.1 Introduction 78
4.2 Novel 14-Dimensional Scalable Feature Extraction Approach 79
4.3 Experimental Evaluation 84
4.3.1 Dataset Details 84
4.3.2 Evaluation Measures 85
4.3.3 Parameter Settings for Evaluation Models 85
4.3.4 Experimental Analysis 85
4.4 Summary 106

5 A Novel 13-Dimensional Alignment-Free Scalable Feature Extraction Method for Genomic Data Clustering **108**

5.1 Introduction . . . 109

5.2 Proposed 13-Dimensional Alignment-Free Scalable Feature Extraction Method . . . 109

5.3 Experimental Evaluation . . . 114

5.3.1 Dataset Details . . . 116

5.3.2 Evaluation Measures . . . 116

5.3.3 Hyperparameter Settings for Evaluation Model . . . 116

5.3.4 Experimental Analysis . . . 116

5.4 Summary . . . 122

6 A Novel Multi-Objective Based Incremental Clustering Method for Dynamic Data Analysis **124**

6.1 Introduction . . . 124

6.2 Proposed Multi-Objective Based Incremental Clustering . . . 126

6.2.1 Preliminaries . . . 126

6.2.2 Proposed Method . . . 127

6.3 Experimental Findings . . . 132

6.3.1 Dataset Details . . . 132

6.3.2 Performance Evaluation Measures . . . 132

6.3.3 Parameter Settings . . . 132

6.3.4 Experimental Analysis . . . 133

6.4 Time Complexity Analysis of the Proposed Approach . . . 138

6.5 Summary . . . 139

7 Investigation of Massive Real-Life Plant SNP and Protein Datasets on Developed Feature Selection and Multi-Objective Based Incremental Clustering **141**

7.1 Introduction . . . 142

7.2 Dataset Details 143
7.3 Preprocessing of Plant SNP and Protein Datasets 144
7.3.1 Preprocessing of SNP Datasets using 12d-FET 144
7.3.2 Preprocessing of Protein Datasets using 60d-FET 145
7.4 Experimental Analysis of Proposed Feature Selection on SNP and Protein Datasets 147
7.5 Experimental Analysis of Proposed Multi-Objective Based Incremental Clustering on SNP Datasets 154
7.6 Summary 157

8 Conclusions and Future Work **159**
8.1 Summary of Research Achievements 161
8.2 Future Research Directions 164

Bibliography **167**

List of Figures

2.1 Categories of feature subsets 18
2.2 Clustering by taking only f_1 19
2.3 Clustering by taking f_1 and f_3 20
2.4 Clustering by taking f_1 and f_2 20
2.5 Development of feature selection process 21
2.6 Filter method for feature selection 23
2.7 Wrapper approach for feature selection 25
2.8 Embedded method for feature selection 27
2.9 Hybrid method for feature selection 28
2.10 Working of incremental clustering 35
2.11 Classification of Big Data processing frameworks 40
2.12 Layered architecture of Apache Spark 41
2.13 Cluster application of Apache Spark 44

3.1 Flow diagram of NCHFS-ACO 59
3.2 Results on Iris dataset 64
3.3 Results on Sonar dataset 65
3.4 Results on Vehicle silhouettes dataset 66
3.5 Results on Ionosphere dataset 67
3.6 Results on Pima dataset 68
3.7 Results on Wine dataset 69
3.8 Results on Wdbc dataset 70
3.9 Results on Parkinsons dataset 71
3.10 Results on Pendigits dataset 72

3.11 Results on Waveform dataset 73
3.12 Comparison between Solorio et al. [1] and NCHFS-ACO in SI 74
3.13 Comparison between Solorio et al. [1] and NCHFS-ACO in JI 75
3.14 Comparison between Solorio et al. [1] and NCHFS-ACO by taking the same no. of features computed by Solorio et al. [1] 75

4.1 Proposed 14d-SFET architecture 80
4.2 Implementation of proposed 14d-SFET on Apache Spark cluster 82
4.3 SI on Wm82.a1 86
4.4 DBI on Wm82.a1 88
4.5 SI on Wm82.a2 90
4.6 DBI on Wm82.a2 90
4.7 SI on Wm82.a4 92
4.8 DBI on Wm82.a4 92
4.9 SI on Lee.a1 93
4.10 DBI on Lee.a1 93
4.11 SI on ZH13.a1 95
4.12 DBI on ZH13.a1 97
4.13 SI on PI483463.a1 99
4.14 DBI on PI483463.a1 99
4.15 SI on W05.a1 101
4.16 DBI on W05.a1 101
4.17 JI on Splice 103
4.18 RI on Splice 103
4.19 JI on Promoter 104
4.20 RI on Promoter 104

5.1 Flow diagram of proposed approach 110
5.2 Example of genome sequence 111
5.3 Execution of proposed 13dim-SFA on Apache Spark 114
5.4 Results on Wheat dataset 117

5.5 Results on Rice 1 dataset . . . 118
5.6 Results on Rice 2 dataset . . . 119
5.7 Results on Rice 3 dataset . . . 120
5.8 Results on Rice 4 dataset . . . 121

6.1 Flowchart of proposed approach . . . 128
6.2 Comparison on Iris data . . . 134
6.3 Comparison on Glass data . . . 135
6.4 Comparison on Wine dataset . . . 136
6.5 Comparison on Sonar dataset . . . 137
6.6 Comparison on Parkinsons dataset . . . 138

7.1 Results on MAGIC-rice . . . 148
7.2 Results on SNP-seek rice . . . 149
7.3 Results on 248Entries rice . . . 151
7.4 Results on W05 Protein dataset . . . 152
7.5 Comparison between Solorio et.al. [1] and NCHFS-ACO in SI on SNP datasets . . . 153
7.6 Comparison between Solorio et.al. [1] and NCHFS-ACO in SI on protein dataset . . . 153
7.7 Comparison on MAGIC-rice . . . 155
7.8 Comparison on SNP-seek rice . . . 156
7.9 Comparison on 248Entries rice . . . 157

List of Tables

2.1 Operations performed in Apache Spark execution 43

3.1 Benchmark datasets from UCI machine learning repository 62
3.2 Hyperparameter settings for NCHFS-ACO 63
3.3 Results on Iris dataset 64
3.4 Results on Sonar dataset 65
3.5 Results on Vehicle silhouettes dataset 66
3.6 Results on Ionosphere dataset 67
3.7 Results on Pima dataset 68
3.8 Results on Wine dataset 69
3.9 Results on Wdbc dataset 70
3.10 Results on Parkinsons dataset 71
3.11 Results on Pendigits Dataset 72
3.12 Results on Waveform dataset 73
3.13 Number of features reduced by the proposed approach in comparison with Solorio et al. [1] for the benchmark datasets 74

4.1 Example of genome sequence with a novel power method 79
4.2 Parameter Settings for evaluation models 85
4.3 Results on Wm82.a1 87
4.4 Results on Wm82.a2 89
4.5 Results on Wm82.a4 91
4.6 Results on Lee.a1 94
4.7 Results on ZH13.a1 96
4.8 Results on PI483463.a1 98

4.9 Results on W05.a1 . . . 100
4.10 Results on Splice dataset . . . 102
4.11 Results on Promoter dataset . . . 105
4.12 Distributed analysis of datasets . . . 106

5.1 Hyperparameter settings for evaluation models . . . 116
5.2 Results on Wheat Dataset . . . 117
5.3 Results on Rice 1 Dataset . . . 118
5.4 Results on Rice 2 Dataset . . . 119
5.5 Results on Rice 3 Dataset . . . 120
5.6 Results on Rice 4 Dataset . . . 121
5.7 Distributed analysis of datasets . . . 122

6.1 Benchmark datasets . . . 132
6.2 Parameter settings for the proposed MOB-IC . . . 133
6.3 Results on Iris dataset . . . 134
6.4 Results on Glass dataset . . . 135
6.5 Results on Wine dataset . . . 136
6.6 Results on Sonar dataset . . . 136
6.7 Results on Parkinsons dataset . . . 137
6.8 Comparison of the proposed MOB-IC time complexity with MOC-FS and online K-means . . . 139

7.1 Example of SNP sequence . . . 144
7.2 Real-life plant SNP and protein datasets . . . 147
7.3 Few best performing k values on MAGIC-rice . . . 148
7.4 Results on MAGIC-rice . . . 148
7.5 Few best performing k values on SNP-seek rice . . . 149
7.6 Results on SNP-seek rice . . . 149
7.7 Few best performing k values on 248Entries rice . . . 150
7.8 Results on 248Entries rice . . . 150

7.9 Few best performing k on W05 Protein dataset 151
7.10 Results on W05 Protein dataset 151
7.11 Results on MAGIC-rice dataset 154
7.12 Results on SNP-seek rice dataset 155
7.13 Results on 248Entries rice dataset 156

List of Abbreviations and Acronyms

ACO Ant Colony Optimization

SNP Single Nucleotide Polymorphism

DNA Deoxyribonucleic Acid

BLAST Basic Local Alignment Search Tool

SI Silhouette Index

ICAR Indian Council of Agricultural Research

IISR Indian Institute of Soybean Research

JI Jaccard Index

DBI Davies Bouldin Index

RI Rand Index

12d-FET 12-dimensional Feature Extraction Approach

NMI Normalized Mutual Information

CH Index Calinski-Harabasz Index

60d-FET 60-dimensional Feature Extraction Technique

SFS Sequential Forward Selection

SBS Sequential Backward Selection

SFFS Sequential Forward Floating Selection

SBFS Sequential Backward Floating Selection

GA Genetic Algorithm

K-NN K-Nearest Neighbors

PSO Particle Swarm Optimization

BPSO Binary PSO

MCFS Multi-Cluster Feature Selection

LLE Locally Linear Embedding

FFEI Fuzzy Feature Evaluation Index

NDFS Non-Negative Discriminative Feature Selection

SVM Support Vector Machine

BAG Bagging Classifier

CT Computed Tomography

ANN Artificial Neural Network

1-gram 1-gram Feature Extraction Approach

MLP Multilayer Perceptron

CNN Convolutional Neural Network

RNA Ribonucleic Acid

17d-FET 17-dimensional Feature Extraction Technique

DNC Di-nucleotide Count

AwA Animal with Attributes

SLMOPs Super Large Scale Multi-Objective Optimization Problems

GPU Graphics Processing Unit

MOC-FS Multi-Objective based Incremental Clustering by Fast Search

IoT Internet of Things

IDC Internet Data Center

NGS Next-Generation Sequencing

GWAS Genome-Wide Association Studies

HDFS Hadoop Distributed File System

RDD Resilient Distributed Datasets

URI Uniform Resource Identifier

DAG Directed Acyclic Graph

NCHFS-ACO Novel Clustering-Based Hybrid Feature Selection Approach using Ant Colony Optimization

14d-SFET 14-dimensional Scalable Feature Extraction Technique

13dim-SFA 13-dimensional Alignment-Free Scalable Feature Extraction Approach

PDS Position Distribution Sequence

LFD Local Frequency Distribution

PSS Partial Sum Sequence

MOB-IC Multi-Objective Based Incremental Clustering

Chapter 1

Introduction

Feature selection, extraction, and clustering are pivotal techniques in machine learning, data mining, and pattern recognition with applications across diverse domains such as healthcare and medicine [2], bioinformatics [3], finance [4], image and signal processing [5], environmental monitoring [6], social media analysis [7], and many more [8, 9]. They play crucial roles in identifying pertinent features and uncovering meaningful patterns. However, despite their widespread utility, the utilization of these approaches in bioinformatics, particularly in the field of plant genomics [10], remains limited. Various issues persist in plant genomics that could be addressed by employing these techniques. For instance, plant genomics often relies on alignment-based sequence matching methodologies [11] to facilitate the comparison of nucleotide or amino acid sequences, allowing for the understanding of evolutionary connections, similarities, and differences among genomes and genes. However, the major disadvantage of alignment-based approaches in plant genomics is their time-consuming nature. As the size and complexity of plant genomes increase with advances in sequencing technologies, the computational demands of alignment-based methods pose significant time and resource challenges. Furthermore, the time-consuming nature of alignment-based approaches can hamper real-time analysis and decision-making, especially in applications that require rapid processing of genomic data, such as clinical diagnostics or field-based research. Furthermore, the scalability of alignment-based methods may be limited when analyzing diverse plant species with different genome sizes and

structures.

To address these challenges, researchers are investigating alternative machine learning approaches, such as clustering [12], alignment-free feature extraction methods [13], feature selection approaches [14], and parallel computing strategies [15], to speed up sequence analysis and improve the efficiency of plant genomic studies, while maintaining genomic accuracy and reliability.

In plant genomics, clustering [16] is a popular machine learning technique to handle these challenges. Clustering is used for organizing and categorizing similar genome sequences. Researchers can use clustering algorithms on genomic data to group sequences that share common characteristics such as sequence homology, structural features, or evolutionary relationships. This enables the identification of genomic patterns, genetic variants, and functional elements in plant genomes. Since, the majority of the data generated by plant genomics and other sources is on a large scale and has high dimensionality, which makes the clustering process more complex, resulting in higher computing time and reduced clustering accuracy. Dealing with high-dimensional data presents new problems for data processing efficiency and effectiveness. To address such challenges, feature selection is one of the most commonly used dimensionality reduction methods, which is useful in reducing the high dimensionality of large-scale data by selecting a small subset of non-redundant and significant features and thus eliminating redundant features in order to construct effective prediction models.

There are four main categories of feature selection methods for unlabeled data: filter method, wrapper method, embedded method, and hybrid method. Among these approaches, in most of the cases, hybrid method performs well because it can capitalize on the strengths of each approach (filter and wrapper), leading to enhanced feature selection accuracy and model performance. In addition, it gives more generalized results in comparison to the wrapper method and adopts the characteristics of filter and wrapper simultaneously. Different types of hybrid feature selection approaches have been proposed by several researchers [17, 18, 19, 20, 1].

As the research increases in this area, various researchers [21, 22, 23, 24] used

Ant Colony Optimization (ACO) to select the important features, because it provides increased accuracy due to its adaptability and discrete representation. An ACO is a bio-inspired algorithm proposed by Dorigo et al. [25] in 1990. This approach simulates the social behaviour of ants seeking food. The ACO may simply use the filter measure along with the wrapper measure to accelerate the search for an optimal feature subset since it has additional parameters that control the search direction. Consequently, ACO may be more suited for high dimensional feature subsets.

Nowadays, a lots of feature selection approaches [21, 22, 23, 24] use ACO solely for labeled data. Conversely, the usage of ACO for unlabeled data has been relatively unexplored. Since most genomic data is unlabeled, designing an efficient hybrid feature selection method for unlabeled data is still an exciting open problem calling for further investigation.

Another issue in domain of plant genomics is handling of large-scale genome data. The extensive volume of data generated within the domain of genomics establishes it as a substantial source of Big Data. In bioinformatics, the application of high-throughput sequencing technologies generates considerable datasets that consist of Single Nucleotide Polymorphisms (SNP) [26], protein [27], and Deoxyribonucleic Acid (DNA) [28] sequences. In the past, scientists grouped large-scale genome sequences by observing similarity using alignment-based methods such as Basic Local Alignment Search Tool (BLAST) [29]. This provided crucial information regarding the evolutionary relationships between genes. However, this method is very time consuming, more computationally extensive and demands high memory. To address these challenges, various clustering techniques such as K-means [30] and Fuzzy c-means [31] are used, which require significantly less time than alignment-based approaches without compromising the accuracy. Since, the genomics data have high dimensionality [32, 33, 34] and huge size, clustering of these high-dimensional genome sequences using these approaches leads the prerequisite of creating an efficient feature selection approach which selects the relevant and non-redundant features from the complete set and reduces the dimensionlity by eliminating unnecessary features. However, to apply the feature selection and clustering on the genome data, there is a need to develop novel alignment-free

feature extraction methods which transforms the genome data into feature vectors (numeric values). Alignment-free feature extraction methods are an efficient alternative to traditional alignment-based techniques because they extract relevant features from genomic data without requiring sequence alignment. These techniques include k-mer counting [35], frequency-based methods [36], and compression-based algorithms [37]. By avoiding alignment, these methods reduce computational complexity and allow for faster analysis of large genomic datasets. Nowadays a lot of alignment-free feature extraction approaches [38, 39] has been developed but most of it are not scalable, i.e., unable to handle the large-scale data. This leads to the poor efficiency [40]. Researchers also addressed this issue by developing scalable feature extraction methods for genome data [41, 42] using Big Data frameworks. Due to the massive generation of genome data day by day, there is a need to innovate advancements in the present method/technology to handle such exponentially growing data. Another challenge is that such huge amounts of data keep on generating regularly from geographically various sources such as Google, Facebook, Twitter, bioinformatics and many more [43, 44]. This data, characterized as online or dynamic, presents the need for real-time analytical methodologies. The analysis of the dynamic data stream can be helpful to derive various conclusions by using the data mining approaches.

One method for dealing with real-time dynamic data is incremental clustering [45]. In the above discussed scenario, the generated dataset is dynamic in nature, so it is not possible to get all data objects at the starting of clustering process. In real-time environment, when new data is coming non-incremental clustering will have to re-cluster all the data, which is highly inefficient and waste lots of computing resources. On the other hand, in incremental clustering whenever new data is coming only clustering needs to be performed on the new data and compare the newly generated clusters with the results of earlier clustering. The method optimizes the clustering process and is particularly suited for applications where time is a crucial component in usability.

One of the biggest problems with incremental clustering is that most of it only uses a similarity-based measure and a single objective to cluster the dynamic data points together [46]. Because of this, most of the clustering algorithms are not strong

against changes in the shape, size, dimensionality, and other properties of the clusters. Multi-objective clustering is used to deal with this problem. So, there is a need of an incremental clustering approach which optimizes the multiple important clustering objectives and efficiently handle the dynamic data.

Based on the above discussion, we can say that there is a need to innovate advancements in the present method or technology to handle various challenges of plant genomics. Motivated by the success of feature selection techniques, Big Data frameworks, and incremental clustering approaches, this thesis investigated the hybrid feature selection, scalable alignment-free feature extraction techniques for genome data and clustering of real-time genome sequences by proposing the multi-objective based incremental clustering. However, the proposed feature selection and incremental clustering models are general purpose which can be applied to any problem.

1.1 Motivation

This dissertation is a study of the design and analysis of a novel feature selection, scalable alignment-free feature extraction techniques, and incremental clustering for handing various issues of plant genomics.

Clustering in plant genomics aids in organizing genomics data [47], revealing genetic patterns and relationships across plant species. Due to the high-dimensional characteristics of genomics data, the clustering procedure is inefficient. As a result, there is a need to design an effective feature selection method for dealing with high-dimensional unlabeled data. Furthermore, in the case of large scale genomics data, it is necessary to employ a scalable alignment-free feature extraction methodology in order to transform biological data into a novel format that is compatible with various data mining techniques. Nowadays, the majority of feature extraction strategies lack scalability, which results in an exorbitant processing time for the vast quantities of genomics data. As a result, it is essential to devise scalable and effective methods for extracting features from genome sequences. In order to analyze massive genome data and implement feature extraction techniques, Big Data analytic frameworks are

applied. In addition, incremental clustering is a viable option for clustering real-time dynamic data generated from various sources in plant genomics. Since, majority of incremental clustering methods rely on a solitary objective to group the dynamic data points together [46]. Consequently, the majority of clustering algorithms exhibit limited robustness when confronted with alterations in the shape, size, dimensionality, and other characteristics of the clusters. Therefore, it is necessary to suggest the approach of multi-objective based incremental clustering.

This thesis investigates the feature selection, scalable feature extraction and incremental clustering approaches to solve various problems of plant genomics. The first problem solved is the high dimensionality problem of genome data. To overcome this problem, we proposed a novel clustering-based hybrid feature selection approach based on ACO which uses K-means clustering to assign the fitness of features in terms of Silhouette Index (SI) along with the Laplacian score in the feature selection process, and to evaluate the performance of the proposed feature selection algorithm. The proposed feature selection approach can be applicable for both labeled and unlabeled data and provides improved clustering performance on a variety of benchmark datasets. Moreover, in order to assess the efficacy of the proposed feature selection algorithm on plant genome data, we obtained massive genome, SNP and protein datasets comprising complex plant genomes from the Indian Council of Agricultural Research-Indian Institute of Soybean Research (ICAR-IISR), Indore, India. These datasets were utilized to extract, select features, and cluster genome sequences. The process of clustering enables ICAR-IISR scientists to characterize genome, SNP, and protein sequences of various plant species in an efficient manner. To preprocess large-scale genome sequences, we proposed novel scalable alignment-free feature extraction methods that yield fixed-length numeric feature vectors for SNP and genome sequences. The proposed scalable techniques are intended to manage Big Data using the Apache Spark cluster computing architecture. In addition, to handle real-time dynamic data, we proposed a multi-objective based incremental clustering method for processing dynamic data. The proposed method enhances the dynamic clustering process by utilizing Euclidean distance to measure the similarity between data points. It also establishes a fitness

function that incorporates three key clustering objective functions: inter-cluster distance, intra-cluster distance, and cluster density. The proposed method utilizes the concept of objective weighting, where a weight is assigned to each objective to produce a single Pareto-optimal solution for the fitness function that is constructed. We investigated the performance of proposed incremental clustering approach on the real-life plant SNP datasets and evaluated results in terms of cluster quality.

1.2 Objectives

In this thesis, we aim to achieve the following objectives:

(i) To develop an efficient hybrid feature selection approach for labeled and unlabeled data, which selects the highly relevant and non-redundant features from the complete feature set.

(ii) To develop a novel scalable feature extraction approach i.e., Apache Spark based 14 dimensional feature extraction approach to extract the important features based on the arrangement from the genome sequences. Further, the extracted features are used as input to the K-means and Fuzzy c-means to cluster massive genome sequences.

(iii) To develop another alignment-free scalable feature extraction method which extracts 13-dimensional feature vector based on the classification of nucleotides using their chemical properties. Further the extracted features are used as input to the K-means clustering to cluster the genome sequences.

(iv) To develop an incremental clustering approach based on multiple objectives, which efficiently handles the real-time dynamic data by optimizing the three major clustering objectives, i.e., intra-cluster distance, inter-cluster distance, and cluster density.

(v) To investigate the performance of proposed feature selection and incremental clustering on the real-life SNP and protein datasets.

1.3 Thesis Contributions

The work done in this field makes significant contributions by designing and developing novel feature selection, scalable alignment-free feature extraction, and incremental clustering techniques for large genome data sets. These contributions fall into three broad categories. First, an unsupervised feature selection approach based on ACO is designed to select the most important and non-redundant features from the entire feature set. Furthermore, the Apark Spark design focuses on developing scalable alignment-free feature extraction approaches specifically tailored for managing large-scale genomics data. The outputs of these feature extraction approaches are fed into the clustering algorithm, which clusters genomic data. Furthermore, a novel incremental clustering method is proposed, which utilizes multiple objectives to effectively cluster real-time dynamic data. In addition, we investigated real-life plant genomics data using the proposed feature selection and incremental clustering approaches.

A succinct synopsis of our research contributions is as follows. The more comprehensive details are provided in subsequent chapters.

Contribution I:

To deal with high dimensionality of genome data, we proposed a novel clustering-based hybrid feature selection approach using ACO that selects features randomly and measures the qualities of features by K-means clustering in terms of SI and Laplacian score. The proposed feature selection approach allows random selection of features, which allows a better exploration of feature space and thus avoids the problem of being trapped in a local optimal solution, and generates a global optimal solution. As a result, when tested on ten benchmark datasets taken from UCI machine learning repository [48], our proposed method is found to be superior compared to the existing method in terms of SI and Jaccard Index (JI). The major contribution of the proposed work is presented subsequently.

- The proposed feature selection approach allows random selection of features, which allows a better exploration of feature space and thus avoids the problem of being trapped in a local optimal solution, and generates a global optimal

solution.

- We combined the Laplacian score as well as the SI to measure the relevancy of a feature rather than using the Laplacian score in the filter stages and then the SI in the wrapper stage separately. The combination of the Laplacian score and the SI facilitates the selection of more relevant features by adopting the characteristics of both measures at the same time.

- In the proposed method, we used an ACO with a tandem run strategy to select the most promising features from the previous iteration, which preserves the good feature throughout the computation without losing them. Hence, through this, we improved power of proposed approach and gave better results.

- The proposed method also provides a facility to choose the different number of optimal features rather than providing a feature subset with certain optimal features.

Contribution II:

To extract features from the huge genome sequences, we proposed an efficient Apache Spark based scalable feature extraction approach that extracts significantly important features from millions of genome sequences in less computational time. The proposed approach extracts features in five stages i.e., based on the length of the sequence, the frequency of nucleotide bases, the pattern organization of nucleotide bases, distribution of nucleotide bases and the entropy of the sequence to generate a fixed-length numeric vector consist of only 14 dimensions to describe each genome sequence uniquely. The feature extracted with the proposed scalable feature extraction approach is applied on K-means and Fuzzy c-means clustering techniques. We used seven real-life unlabeled plant genome datasets of soybean crop [49] and two labeled benchmark datasets, i.e., promoter [48] and splice [48]. To evaluate the performance of the proposed method on unlabeled datasets, we employed two internal evaluation measures, SI and Davies–Bouldin Index (DBI). On the other hand, to evaluate the performance of the proposed approach on the labeled datasets, we used two external

evaluation measures, JI and Rand Index (RI). The experimental results show that the proposed method is highly successful and efficient in terms of computing time in comparison to other state-of-the-art approaches. The major contribution of this work is as follows:-

- The proposed approach uses the alignment-free method for genome sequence analysis.
- The proposed approach is scalable and uses distributed computing to process the genome sequences to save computation time.
- The proposed approach removes the drawback of the total distance and distribution features identified in 12-dimensional feature extraction approach (12d-FET) [50], using a novel power method.
- The proposed approach extracts most essential features i.e., the sequence length, frequency of nucleotides, modified total distance, distribution and entropy of sequences to represent the each genome sequence uniquely.

Contribution III:

In previous work, we proposed a 14-dimensional feature extraction approach for obtaining features from genome sequences. This method extracts the most important features based on the arrangement of nucleotides. However, this method does not extract features based on nucleotide classification using chemical properties. So, to further improve the performance, we proposed another scalable feature extraction method which efficiently handles large scale genome data by distributing the tasks on various nodes and extract the significantly important features based on classification of nucleotides using their chemical properties in terms of entropy and the length of the sequence. We performed the experiments using five real-life plant genome datasets of rice and wheat crops obtained from rice genome library [51] and from Han et al. [52], respectively. The clustering of genome sequences is performed by taking extracted features using K-means clustering. We measured the clustering results in terms of SI and DBI. The major contribution of this work is as follows:-

- The proposed approach uses the alignment-free method for genome sequence analysis.
- The proposed approach is scalable and uses distributed computing to process the genome sequences to save computation time.
- The proposed approach extracts features by classifying the nucleotides into three classes, i.e, Pyrimidine and Purine group, Keto and Amino group, and strong hydrogen bond and weak hydrogen bond group.
- The experimental finding shows that the proposed approach performs well when compared with the other state-of-the-art approaches.

Contribution IV:

To handle real-time dynamic data, we proposed a multi-objective incremental clustering approach. This method has three primary clustering objectives: inter-cluster distance (distance between clusters), intra-cluster distance (mean distance between each data point and the center of its cluster), and cluster density (to improve clustering precision). The optimal clustering requires a high inter-cluster distance, a low intra-cluster distance, and a high cluster density. To optimize the three objectives mentioned above at the same time, the proposed approach combines all three objective functions with weight coefficients to form a single fitness function. Hence, facilitates a optimal clustering, which is strong against changes in the shape, size, dimensionality, and other properties of the clusters. We used two evaluation measures named RI and Normalized Mutual Information (NMI) to measure the effectiveness of the proposed approach. The proposed method outperforms other state-of-the-art methods on five benchmark datasets taken from UCI machine learning repository [48]. The major contribution of the proposed work is as follows:

- The proposed approach simultaneously optimizes three major clustering objective functions i.e., intra-cluster distance, inter-cluster distance, and cluster density rather than optimizing each objective separately. Hence facilitates the better clustering by adopting the characteristics of three objectives at the same time.

- The proposed approach creates and updates the cluster in real-time during dynamic data processing, hence does not demand the various clustering parameters, i.e., number of clusters, assignment of cluster center in advance.

- The proposed approach uses objective weighting concept, hence assigns almost equal weight to each clustering objective and got rid of getting biased solution towards a particular objective and facilitates better Pareto-optimal solution.

- The proposed approach efficiently clustered the small, medium and huge size datasets even if the number of dimensions are more.

Contribution V:

The research contributions described in I and IV pertain to benchmark datasets. However, to investigate the performance of the proposed feature selection and multi-objective based incremental clustering on real-life plant SNP and protein datasets, we extracted features from these sequences. Specifically, we applied a 12d-FET [50] to extract features from the SNPs and a 60-dimensional feature extraction technique (60d-FET) [53] to extract features from the protein sequences. We evaluated the performance of the proposed feature selection on both SNP and protein data in terms of SI. Furthermore, we evaluated the performance of proposed incremental clustering solely on the SNP data, and measured results in terms of both SI and the Calinski-Harabasz Index (CH Index).

1.4 Organization of the Thesis

This thesis is divided into a total of eight chapters. Below, a synopsis of each chapter is presented:

Chapter 1 (Introduction)

This chapter discusses the background knowledge of plant genomics, feature selection, scalable feature extraction, and incremental clustering, the rationale for our exertion, and the outcome of this thesis.

Chapter 2 (Literature Survey)

This chapter discusses different methodologies for feature selection and its functioning. Subsequently, the discussion shifts towards the techniques employed for extracting features from genomic data and the subject of genome clustering. Furthermore, it offers a succinct summary of incremental clustering for the purpose of analyzing dynamic data, as well as an evaluation of the most advanced incremental clustering techniques. Subsequently, the study of Big Data processing frameworks is elucidated, along with the parallel processing approaches employed for handling Big Data. After that, it offers a comprehensive evaluation of performance metrics. Furthermore, this chapter includes comprehensive information regarding the specific genome datasets employed in real-life scenarios.

Chapter 3 (A Novel Clustering-Based Hybrid Feature Selection Approach Using Ant Colony Optimization)

In this chapter, we presented a hybrid feature selection algorithm based on ACO that removes redundant and irrelevant features that have a negative impact on model building and selects the more appropriate features from data with a large number of features. This approach measures the relevancy of a feature using a combination of Laplacian score and SI, rather than using Laplacian score in the filter stages and SI in the wrapper stage. It also employs a tandem run strategy to choose the most promising features from the leader subset. Furthermore, this chapter contains an experimental evaluation that compares our proposed method to another feature selection method proposed by Solorio et al. [1]. The results are compared on ten benchmark datasets taken from the UCI machine learning repository in terms of SI and JI.

Chapter 4 (A Novel Apache Spark Based 14-Dimensional Scalable Feature Extraction Approach for the Clustering of Genomics Data)

The methods proposed in Chapters 3 and 6 do not take raw genome sequences into account for feature selection and incremental clustering. To address this issue, in this chapter, we proposed a novel 14-dimensional scalable feature extraction approach based on Apache Spark that converts raw genome sequences into numerical features.

This method extracts significant features, from thousands of genome sequences, by distributing tasks across multiple cores. This method also overcomes the drawback of another state-of-the-art method proposed by Preeti et al. [50]. The performance of this approach is measured by clustering these feature vectors using K-means and Fuzzy c-means. The proposed approach is tested on unlabeled genome sequences of seven soybean crop varieties, as well as two benchmark labeled datasets from the UCI machine learning repository. The experimental results are compared to three cutting-edge approaches in terms of SI and DBI for the unlabeled dataset, and RI and JI for the labeled dataset.

Chapter 5 (A Novel 13-Dimensional Alignment-Free Scalable Feature Extraction Method for Genomic Data Clustering)

The method proposed in Chapter 4 extracts features based solely on sequence order and length. However, this method does not extract features based on the chemical properties of nucleotides, which would provide additional information about genome function. In this chapter, we proposed a scalable 13-dimensional feature extraction method that classifies nucleotides based on chemical properties to extract key features. This method utilizes the Apache Spark cluster to preprocess large raw genomes by distributing tasks across multiple cores. We tested the performance of this approach on five unlabeled real-life plant genome datasets of rice and wheat crops obtained from rice genome library [51] and Han et al. [52], respectively. After extracting features, we clustered genome sequences with K-means and assessed performance in terms of SI and DBI. The experimental results shows that proposed approach performs well in comparison to the state-of-the-art approaches.

Chapter 6 (A Multi-Objective Based Incremental Clustering for Dynamic Data Analysis)

In this chapter, we presented a multi-objective based incremental clustering algorithm for real-time dynamic datasets. This method considers three primary clustering objectives: inter-cluster distance, intra-cluster distance, and cluster density, and then constructs a fitness function that combines these three to cluster the data points. In this method, we used the concept of objective weighting and assigned

nearly equal weight to each clustering objective in order to achieve optimal clustering results that are not biased toward a specific objective. Furthermore, the proposed approach creates and updates clusters in real-time, hence, eliminating the need for multiple clustering parameters in advance. The proposed approach is tested on five labeled benchmark datasets from the UCI machine learning repository, and the results are evaluated using the RI and NMI Indexes. The proposed method outperforms two cutting-edge methods proposed by Sivadi et al. [54] and Abernathy et al. [55].

Chapter 7 (Investigation of Massive Real-Life Plant SNP and Protein Datasets on Developed Feature Selection and Multi-objective Based Incremental Clustering)

This chapter presents the methodology of utilizing a 12d-FET [50] approach to analyze real-world SNP datasets. Furthermore, we elucidated the utilization of a 60d-FET [53] methodology for the analysis of extensive protein dataset. In addition, we examined the effectiveness of the proposed feature selection (Chapter 3) and multi-objective based incremental approach (Chapter 6) using the extracted features as input. Additionally assessed the performance based on the quality of the clusters in terms of internal evaluation measures.

Chapter 8 (Conclusions and Future Work)

This chapter presents a brief summary of the impact made by this thesis and outlines possible future directions of our research.

Chapter 2

Literature Survey

This thesis investigates the design of feature selection, scalable feature extraction, and incremental clustering for handling various issues of plant genomics. Therefore, this chapter offers a comprehensive literature survey of these approaches. The survey related to feature selection approaches and its type is presented in Section 2.1. Next, the discussion turns to the existing feature extraction strategies used for genome data in Section 2.2. In addition, Section 2.3 provides a concise overview of incremental clustering for dynamic data analysis and a review of state-of-the-art incremental clustering methods. The study pertaining to Big Data processing frameworks is described, together with the parallel processing techniques for managing Big Data, in Section 2.4. Furthermore, Section 2.5 provides an analysis of performance measures. The last section (Section 2.6) covers the details of the real-life genome and protein datasets used in this study.

2.1 Feature Selection

Feature selection refers to the procedure of picking a portion of original features based on their significance and redundancy. According to Yu and Liu [56], the feature subsets can be categorized into four groups: (i) highly relevant feature subsets, (ii) weakly relevant and non-redundant feature subsets, (iii) weakly relevant and redundant feature subsets, and (iv) completely irrelevant and noisy feature subsets as

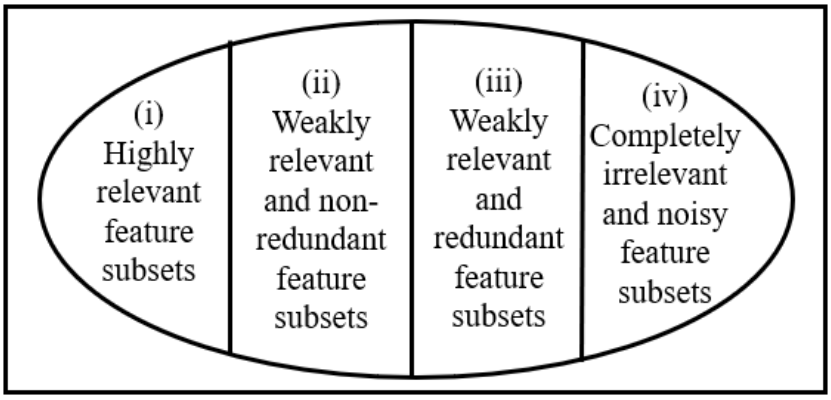


Figure 2.1: Categories of feature subsets

shown in Figure 2.1. A feature is irrelevant if it does not contribute to the accuracy of the prediction. To construct a decent prediction model, choosing all highly relevant and some weakly relevant features is desirable while excluding irrelevant, redundant, or noisy features. Sometimes, weakly relevant features that are non-redundant and compatible with assessment methods can also help to improve the prediction accuracy. During the feature selection process, redundant ones are usually thrown out because they may have critical statistical relationships with other features, not because they have information that isn't useful. Sometimes, a feature may be unimportant as a standalone entity, but it might be beneficial when paired with other features.

In machine learning, the experimental data may be unlabeled, labeled, or partially labeled. This makes it possible to use unsupervised, supervised, and semisupervised feature selection techniques to select the essential and relevant features. Usually, labeled data is a collection of samples that are annotated by meaningful labels. Supervised feature selection refers to the procedure of picking a group of features based on a set of criteria for figuring out the value and importance of the features. On the other hand, unlabeled data is made up of samples and things that can be seen without labels. Unsupervised feature selection, in which we don't know anything about the underlying functional classes ahead of time uses data structures like data variance, separability, and distribution to figure out the importance of each feature. In the semisupervised feature selection, some portion of labeled data is added to unlabeled data as extra information to make an unsupervised feature selection work better. Nowadays, a lot

of literature is available within the scope of supervised and semisupervised feature selection. However, the unsupervised feature selection is relatively less explored. So, here, our primary focus is to explore unsupervised feature selection approaches [57].

In unsupervised learning, clustering or grouping is the primary operation performed on the unlabeled data to identify the essential clusters. Clustering can be negatively impacted by extraneous and redundant data features, which can deteriorate the cluster quality, leads to extensive computation cost, and increase memory needs. Consequently, to improve the performance the unsupervised feature selection is performed to get rid of of such redundant and unimportant features. To illustrate this concept, we provided the Figure 2.2, Figure 2.3, and Figure 2.4 which demonstrate the clustering of a dataset by taking different feature subsets. Figure 2.2 shows that f_1 is adequate for identifying distinct clusters. However, Figure 2.3 shows that f_3 is redundant and negatively affects the homogeneity of clusters. In Figure 2.4, it is shown that f_2 is unimportant, has no effect on the clustering process at all because f_1 is alone capable to identify the distinct cluster. In addition, various subsets of characteristics including pertinent information may provide varying degrees of clustering. Therefore, to investigate the various methodologies of unsupervised feature selection, we provided an overview of unsupervised feature selection approaches, their development, and its types in the following subsection.

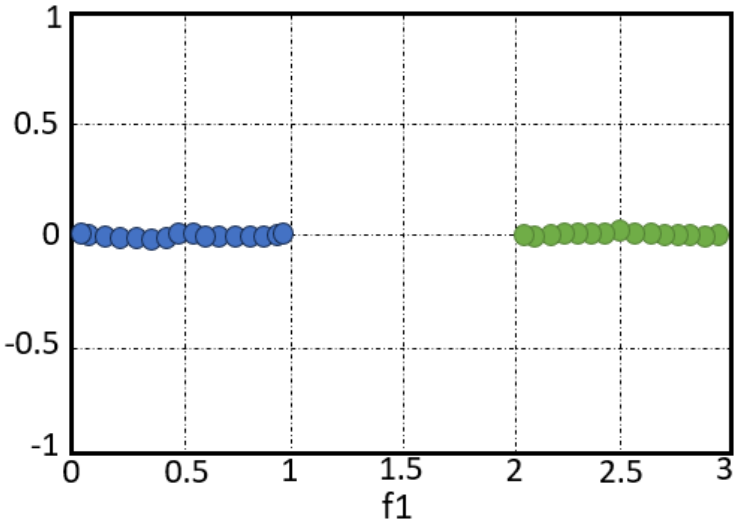


Figure 2.2: Clustering by taking only f_1

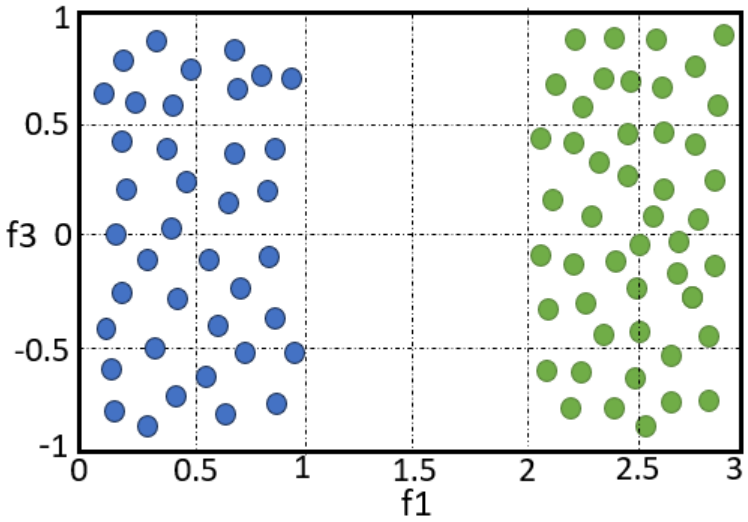


Figure 2.3: Clustering by taking f_1 and f_3

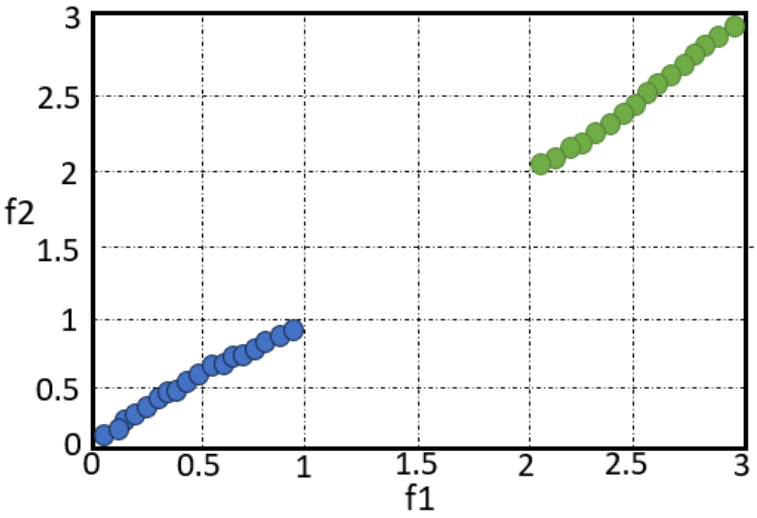


Figure 2.4: Clustering by taking f_1 and f_2

2.1.1 Development of Feature Selection

The development of the feature selection process consists of five steps named search direction, search strategy, evaluation criteria, stopping criterion, and result validation, as shown in Figure 2.5. These steps are discussed in detail subsequently.

First step:

The first step of the feature selection procedure is to determine the starting point

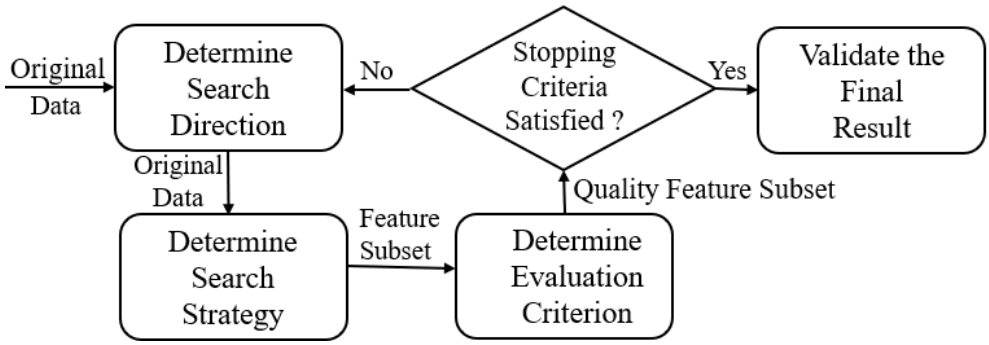


Figure 2.5: Development of feature selection process

and the search direction. Two ways are available for this process: forward search and backward search. In the forward search, the construction of the feature subset begins with a null subset, and then adding the features occurs in successive iterations. On the other hand, in backward search, the process starts with a complete set of features, and then the elimination of features happens in successive iterations.

Second step:

The second step of the feature selection process determines the search strategy. In this step, a subset of features is selected using a predetermined search technique. There are three categories for search techniques named as sequential, randomized, and exponential. The sequential search strategy is also called "greedy hill-climbing search", in which the addition of one feature happens at a time. The typically used sequential search approaches are sequential forward selection (SFS) and sequential backward selection (SBS). These search strategies are easy to implement, and the complexity of these strategies is proportional to the number of features. It can handle problems with multiple features that are similar. However, these methods do not work well with indices that aren't monotonic, and they may produce the nesting effect because once a feature is inserted (or deleted), it can't be deleted (or inserted) again. Also, they are sensitive to how features interact, so it's easy for them to get stuck in local minima. To resolve this issue, sequential forward floating selection (SFFS) and sequential backward floating selection (SBFS) were introduced by giving users ways to re-select deleted features and remove already included features. A few examples of the sequential search strategy are beam search, best-first search, improved version of

best-first search, and the plus-l take-away r algorithm.

Another search technique is the randomized search method that select features at random and then employs two distinct search methods. Firstly, it uses search methods like simulated annealing and random hill-climbing that work in a sequential or two-way manner. Secondly it uses search strategies that don't follow a linear approach, like the Genetic Algorithm (GA), the las vegas algorithm, and the tabu search.

The exponential search begins with the original features and finds the best solution. But this strategy is hard to use and takes a lot of computing power, particularly for high-dimensional datasets. An illustration of this technique is exhaustive search, which looks at all possible subsets to find the best one.

Third step:

The third step of the feature selection process is the determination of evaluation methods. In this step, the selected subset of features is examined based on specific evaluation method. There are four types of evaluation methods for choosing features: filter, wrapper, embedded, and hybrid, which are elaborated in detail in Section 2.1.2.

Fourth step:

The fourth step of the feature selection task is determining the stopping criteria. It determines when the feature selection process should end. A selection of good stopping criterion can avoid overfitting, making finding the best feature subset easier and more effective. Decisions taken in earlier phases affect the choice of a termination criterion. Common cutoffs include reaching a certain number of features or iterations, getting better by a certain percentage between iterations, or getting an ideal feature subset formed on some evaluation function.

Fifth step:

The fifth step of the feature selection procedure is validation. Various validation techniques have been proposed to test how well potential feature sets work for the learning algorithm. In the supervised context, the most common ways to estimate error are cross-validation and performance measurements based on a confusion matrix. On the other hand, in the unsupervised context, the Rand Index and the Jaccard Index are used to measure similarity. In previous studies [58, 59], some additional validation

and analysis have also been done. For example, the Kuncheva Index is used to measure stability, and analysis of variance is used to measure complexity. The various existing feature selection approaches are discussed subsequently.

2.1.2 Types of Feature Selection

As previously mentioned, feature selection is categorized into four types based on evaluation criteria, which are detailed next.

Filter method:

The first evaluation method is the filter method, where feature relevance is measured using four distinct categories of evaluation measures: information, distance, consistency, and dependency. Since the filter method doesn't depend on any learning algorithm, it can be used to find general solutions for different classifiers or clustering techniques. The filter method is the oldest and is also called an open-loop method. The working of the filter method is shown in Figure 2.6.

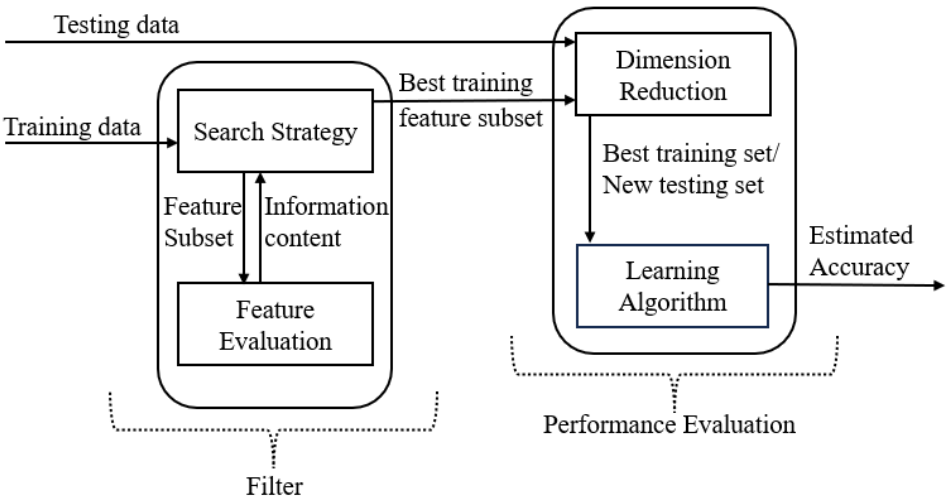


Figure 2.6: Filter method for feature selection

Various researchers [60, 21, 61] have proposed different approaches to filter feature selection, which are discussed as follows. In 2014, Banerjee et al. [60] proposed a filter method based on singular value decomposition entropy. In this method, they selected features by evaluating the entropy of the initial data matrix by observing its singular values. The entropy ranges from 0 to 1. When the entropy value is close

to 0 (low), this means the spectrum of the data matrix is not consistently scattered, and the well-formed cluster is generated. On the other hand, when the entropy value is close to 1 (high), the spectrum of the data matrix is consistently scattered, and clustering is not well-defined. Later, Tabakhi et al. [21] proposed an unsupervised Ant Colony Optimization (ACO) based feature selection that uses cosine similarity measures to measure the similarity between features. The number of artificial ants used in this study was equal to the number of attributes in the dataset, so each ant was responsible for constructing a feature subset of the dataset. If the selected attributes appear in most of the subsets then the selected attributes are updated with the higher pheromone value. Feature subsets with low similarity and high pheromone values were iteratively considered and potentially added until the maximum number of iterations were reached. They tested their approach on a variety of UCI machine learning datasets [48], including the wine and breast cancer datasets, and received an average classification error of 19.8 percent. In 2017, Solorio et al. [61] proposed a feature selection method based on the Laplacian score for mixed data. In this method, they evaluated features by assessing the changes in the spectrum distribution (spectral gaps) of the first non-trivial eigenvalues of the normalized Laplacian matrix when each feature is omitted from the entire collection of features individually. After that, they arranged features in downward order according to their individual spectral gaps.

The main drawback of above filter based feature selection approaches is that the chosen features may not perform well across all learning models. Features selected based on specific criteria might not be universally effective across different algorithms. This limits their applicability and effectiveness, highlighting the need to consider the characteristics of individual models when selecting features. To overcome this drawback, researchers [62, 63, 64, 65] have explored wrapper based feature selection methods, which are discussed subsequently.

Wrapper method:

The second evaluation method is the wrapper method which binds the feature selection process around the learning algorithm and makes use of performance accuracy or the learning error rate as a criterion for evaluating feature quality. Unlike filter method,

it chooses the most useful feature subset by reducing the error of a specific learning approach. The wrapper method usually gives better results in comparison to the filter method because it uses bias directly in the learning algorithm and considers the feature dependency. The wrapper method gives less generalized features in contrast to the filter method because it uses a learning algorithm to assign fitness to a feature. So, no assurance can be given that the picked features are the best for other different learning algorithms. The working of the wrapper method is shown in Figure 2.7.

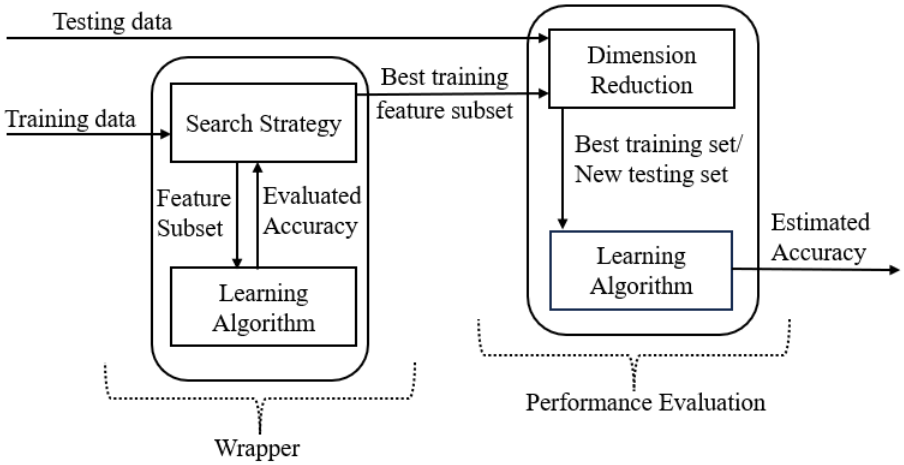


Figure 2.7: Wrapper approach for feature selection

Next, we discussed wrapper feature selection methods proposed by different researchers [62, 63, 64, 65, 66]. Kanan et al. [62] developed an ACO based wrapper technique for feature selection to identify the human face images. The K-Nearest Neighbors (K-NN) classifier used in this method to select the relevant feature subsets on the basis of two factors, i.e, it combines the feature subset size and pheromone level to identify the relevant feature subsets. This approach was tested on the Cambridge University facial image dataset consisting of 400 images with an obtained accuracy of 99.7% and 98.5% by considering two different sets of features. Later, Javani et al. [63] introduced a new Particle Swarm Optimization (PSO) based simultaneous clustering and feature selection approach that uses a probabilistic K-means clustering and a new kernelized validity index to overcome the negative impact of the evolutionary process initial condition. However, the main drawback of this method is that it hasn't been tested on high-dimensional datasets. Later, Swetha and Devi [64] proposed a

two-stage PSO feature selection approach for clustering. In the beginning, they used two-stage PSO to select features and then used clustering on those features. Prakash et al. [65] proposed another feature selection method based on Binary PSO (BPSO), in which each feature subset is encoded by 0 or 1. BPSO was used for feature selection, and K-means clustering was used to measure the quality of possible feature subsets in terms of Silhouette Index [67]. In 2019, Prakash and Singh [66] proposed another simultaneous feature selection and clustering approach based on genetically inspired multi-objective binary gravitational search. In this method, they used feature subset size and Silhouette Index as objectives to search for possible solution spaces. They also used an external archive to make the non-dominated set. Later on, K-means clustering is applied to the segregated dataset according to the selected feature subset, and then they measured F-score for each subset. The subset giving the best F-score was selected as the final subset. Experimental results show that this approach performs better than the elitist non-dominated sorting GA proposed by Deb et al. [68].

The main disadvantage of above discussed wrapper feature selection approaches is that they produce results that are closely related to the specific learning model used. While designed to improve performance within a specific model, this approach may limit generalizability across different algorithms or datasets. Additionally, these methods are often time-consuming. To overcome the time-consuming issue, embedded method is introduced, which is given next.

Embedded method:

The third evaluation method is the embedded method. Unlike wrapper method, it is a way to choose features built into the learning algorithm and utilizes the properties of the algorithm, which help in deciding how to evaluate features. Even though the performance is the same, the embedded method is more efficacious and easier to compute than the wrapper method. This is because the embedded method eliminates the cost of running the learning algorithm and looking at each feature subset over and over again. Also, this method is less likely to overfit than the wrapper method. Figure 2.8 illustrates the working of the embedded method.

Some approaches of embedded feature selection is discussed subsequently. In 2010,

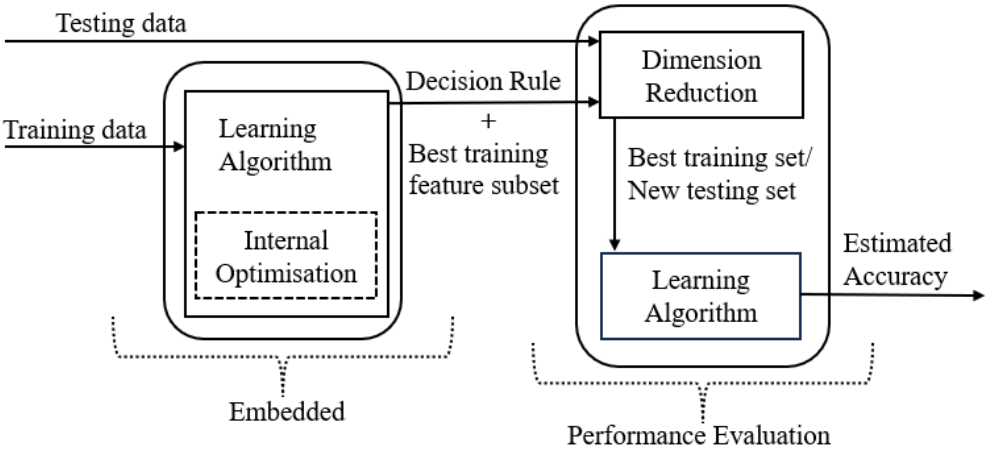


Figure 2.8: Embedded method for feature selection

Cai D et al. [69] proposed an embedded sparse learning feature selection method named multi-cluster feature selection (MCFS) for the clustering task. In this approach, they used three steps to perform the feature selection task. First, they used spectral analysis to compute the correlation between features. In the second step, an L1-regularized least square regression model is used to measure how well the features fit. In the third step, the features with the greatest coefficient values found earlier in the process are chosen. Recent work has been done on sparse learning models that use non-convex sparse regularizer functions and locally linear embedding (LLE). Zechao Li et al. [70] introduced a non-negative discriminative feature selection (NDFS) method for selecting discriminative features from data. In this strategy, they used spectral clustering and feature selection to choose the most discriminative features. Furthermore, they used a non-negative constraint to learn a more accurate cluster label. They tested their method on several benchmark datasets, including UMIST, AT&T [71], and JAFFE [72]. Luo et al. [73] came up with a new feature selection method that models the data's manifold structure with LLE. The objective is to describe the intrinsic local geometry with an LLE graph in place of the usual pairwise similarity matrix and a structure regularization term. A feature-level reconstruction rating is set up for each feature using the LLE graph. This score is used to choose the final subset of features.

Although embedded approaches reduce computation time, however, their reliance

on the specific learning algorithm for feature selection may limit their generalizability across various models or datasets. To overcome the limitations of individual methods, a hybrid approach has been proposed. This method combines the strengths of filter and wrapper techniques to achieve more robust and effective feature selection. By integrating both strategies, it aims to enhance performance and adaptability across various learning scenarios. The details of hybrid method is followed next.

Hybrid method:

The fourth evaluation method is the hybrid method. A hybrid approach can be created by fusing two distinct strategies (such as a filter and a wrapper) or two techniques that share a common criterion or two feature selection methods. The objective of the hybrid method is to take advantage of the best features of both methods. It employs many evaluation criteria at various stages of the search to enhance the accuracy and speed of predictions by making better use of faster computers. There are two different hybridization methods now in use. One approach uses the filter method, which first narrows down the feature set before passing it through the wrapper method to find the optimal feature subset as shown in Figure 2.9. On the other hand, the second strategy couples the filter and wrapper measures to allocate the relevance score to a specific feature.

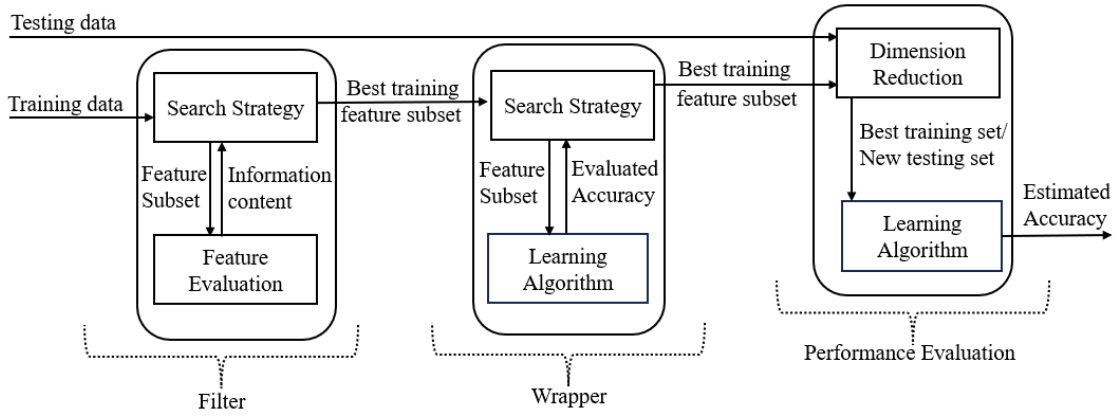


Figure 2.9: Hybrid method for feature selection

Numerous researchers [17, 74, 19, 75, 76, 77, 22, 23] have explored hybrid feature selection approaches, which are discussed next. Dash and Liu [17] developed a hybrid feature selection approach that derives entropy from data similarity and evaluates fea-

tures in the filter stage using an entropy-based measure. The wrapper stage employs scatter separability criteria and K-means clustering to select the relevant feature subset. The disadvantage of this approach is that it is computationally intensive. Ahmed Al-Ani [74] suggested an ACO based hybrid feature selection algorithm. In this approach, artificial ants were used to search feature space and to generate the feature subsets. These subsets were evaluated by the neural network. After that, they used mutual information to identify a predetermined number of best feature subsets, and these feature subsets were considered for future iteration until the termination criteria have achieved. They tested this approach on two datasets, namely, speech segment and image texture, and earned an accuracy of 84.2% and 89.3%, respectively. Li et al. [19] later proposed a hybrid feature selection algorithm based on Dash and Liu's concept. To improve performance, they used a fuzzy feature evaluation index (FFEI) in conjunction with an exponential entropy index to evaluate the feature in the filter stage. For the wrapper stage, they used a scatter separability criterion and a Fuzzy c-means algorithm; however, this approach also has a high computational cost. Later on, Sivagaminathan et al. [75], proposed a hybrid feature selection approach, in which they used the ACO and neural network together. In ACO, artificial ants were used to explore feature space and to generate feature subsets. Furthermore, a neural network trained using the Levenberg-Marquard backpropagation algorithm was used to evaluate the efficacy of these feature sets by computing the classification error. This method was tested on six datasets taken from the UCI machine learning repository [48] and achieved an accuracy of 77.5 to 98%. Later on, Chen et al. [76] suggested a hybrid feature selection approach based on ACO, which combines SVM classifier and F-Score measure. They used SVM classification accuracy and feature subset size to update the pheromone value of features. They tested this approach on 80 images having 19 features and 4 class labels, and thus obtained the precision value of 97.08%.

Liu et al. [77] proposed a hybrid feature selection approach. They used a fisher criterion with a genetic optimization algorithm to measure the fitness of features at the filter stage. Later in the wrapper stage, feature subsets with fitness above a threshold value will be passed to the K-NN classifier for evaluation. This method

was tested on four different classifiers, namely K-NN, Naive Bayes, Support Vector Machine (SVM), and Bagging classifier (BAG), and achieved an accuracy of 75.70%, 73.58%, 80.26%, and 77.88%, respectively. Solorio et al. [1] proposed a hybrid feature selection technique that ranks features using a Laplacian score and measures a feature subset using a modified Calinski-Harabasz Index. They compared their approach to the one proposed by Dash and Liu [17] and Li et al. [19] on several benchmark datasets taken from the UCI machine learning repository [48] as well as on synthetic datasets and found it to be more effective. Dhalia et al. [22] proposed an ACO based approach that selects the relevant feature subset via a tandem run strategy. They used the cosine similarity measure to compare features, and SVM to assign fitness to a feature. They validated their method using lung computed tomography (CT) scan images to diagnose bronchitis and achieved an accuracy of 81.66 percent. Joseph et al. [23] proposed a feature selection approach using ACO and Artificial Neural Network (ANN) for text classification. They used reuter's dataset to test the accuracy. Wenping et al. [24] proposed an ACO based hybrid feature selection method which uses an interval strategy to identify the size of optimal feature subset. They tested their approach on 11 high-dimensional datasets and found that their method takes less execution time and performed better than various state-of-the-art methods.

Based on the above discussion, it can be concluded that hybrid methods outperform filter and wrapper methods. Additionally, ACO [62, 21, 74] is used in a variety of tasks for feature selection and provides increased accuracy due to its adaptability and discrete representation. Moreover, ACO [75, 22, 76] may simply use the filter measure along with the wrapper measure to accelerate the search for an optimal feature subset since it has additional parameters that control the search direction. Consequently, ACO may be more suited for high-dimensional feature subsets. However, the majority of the approaches in the preceding literature use ACO in feature selection for labeled data; however, the use of ACO for unlabeled data in an unsupervised context has received little attention. Plant genomics is one of the area where lot of genomics data of various plants are unlabeled [49]. So, there is a need to handle such data by applying clustering after feature selection to improve the clustering performance.

Hence, we proposed a novel clustering based hybrid feature selection approach using ACO, which will be presented in Chapter 3.

However, to utilize feature selection and clustering on genome data, novel feature extraction approaches are required to transform the genome data into feature vectors (numeric values). So, in the next section, we discussed the various feature extraction approaches.

2.2 Feature Extraction Approaches

Feature extraction from genome sequences [50] transforms biological sequences into numerical data, making them easier to analyze using data mining methods. This preprocessing stage is critical for a reliable knowledge process because it directly influences the final result. Several features can determine each genome sequence, resulting in a vector of numerical values obtained by a description function that binds sequences and features. Bandyopadhyay [78] proposed a 1-gram feature extraction approach (1-gram) for the classification of protein sequences into known superfamilies. In this approach, they counted the occurrence of each amino acid in a protein sequence. The primary disadvantage of this method is that it does not extract context-based features and requires a lot of time to process the huge amount of protein sequences. Subsequently, Terje et al. [79] proposed a sliding window-based feature extraction technique for DNA sequences. They used window sizes of 2, 3, and 4 to extract the frequencies of di-nucleotide, tri-nucleotide, and tetra-nucleotide patterns, respectively. They tested their approach to classifying the eukaryotic and prokaryotic Deoxyribonucleic Acid (DNA) sequences using a SVM and Multilayer Perceptron (MLP) classifier and achieved an accuracy of 85% and 84.6%, respectively. The drawback of this approach is that they used only frequency-based features to describe the DNA sequence. In 2014, Bao et al. [80] introduced an alignment-free feature extraction method for the clustering of DNA sequences using K-means. They employed Shi et al.'s [81] method to classify nucleotides into three categories: pyrimidine or purine, keto or amino, and strong or weak hydrogen bonds. After converting the DNA sequences into three classes,

they extracted features based on the word probability distribution rather than simply matching the sequences. The primary shortcoming of this strategy is that they did not extract features such as sequence length and did not test it on massive datasets. Later on, in 2016, Nguyen et al. [82] proposed a Convolutional Neural Network (CNN) based approach to classify the DNA sequences. They passed one hot vector representation of a DNA sequence as an input to CNN. They tested their method on 10 histone, splice, and promoter datasets. They achieved significantly higher accuracy than the previous methods proposed by Towell et al. [83], Li et al. [84], and Higashihara et al. [85]. The drawback of this approach is that, one-hot representation does not use any contextual or semantic information of sequence, which is very important in DNA analysis. In 2017, Raid AL-Zubi et al. [86] suggested a hybrid feature selection method for the complex disease Single Nucleotide Polymorphisms (SNP). In this work, they used lossless transformation to convert the SNP sequence into numerical values. They tested their approach on 5 SNP datasets, namely thyroid cancer, colorectal cancer, breast cancer, autism, and mental retardation, and achieved accuracy up to 89.50%. The disadvantage of this approach is that it does not use any semantic information of the sequence.

Helaly et al. [87] suggested a deep learning approach for the taxonomy classification of biological bacterial sequence. They used various representations like one-hot encoding, inter-encoding, and k-mers-based representation to describe the biological sequences. They tested their approach on the 16S rRNA (Ribonucleic Acid) dataset using a deeper CNN and achieved an accuracy of 91.7% with more representative representation and 90.6% with less figurative representation. After that, Jasbir Dhaliwal and John Wanger [88] developed a new feature extraction method for highly expressed SNPs. They used k-mers as features to describe the SNP sequence. They said that optimal k-mers and feature size might differ for different research problems. They evaluated their algorithm on 49 human tissues using a multinomial Naive Bayes and got optimal k-mers of size 3. One major disadvantage of using k-mers is that large-sized k-mers take a massive amount of memory to store an SNP sequence and also take a lot of time for processing. To overcome this disadvantage, numerous Big Data pro-

cessing frameworks are adopted to make the approaches scalable so that these scalable approaches can process massive SNP data. In 2021, Jha et al. [50] proposed a novel scalable 12-dimensional feature extraction approach (12d-FET) for the SNP sequence analysis of unlabeled real-life plant genome datasets. To describe an SNP sequence, they used three types of features, i.e., frequency, total distance, and arrangement of nucleotides. They tested their approach using kernelized scalable random sampling with iterative Fuzzy c-means and evaluated results in terms of Silhouette Index [67] and Davies-Bouldin Index [89]. The drawback of this approach is that the total distance and distribution feature for each nucleotide may be the same for the non-similar sequences. Due to this inability, this approach may be unable to distinguish between the sequences. Also, this approach does not take the essential features in consideration like the sequence length, which is different for the different organisms. Later, Bonidia et al. [90] proposed a feature extraction package for the analysis of genome sequence called mathfeature, which consists of several feature extraction approaches. In this package, the authors presented a 17-dimensional feature extraction technique (17d-FET) comprised of the di-nucleotide count (DNC) and tsallis entropy. The primary drawback of this method is that it did not extract context-based characteristics.

Based on the preceding study, it can be stated that majority of feature extraction strategies are not scalable due to this unable to process the huge amount of genomics data efficiently. In addition, several algorithms do not extract essential features like the sequence length, entropy, and context-based characteristics. Moreover, some approaches are also suffering from the curse of dimensionality. In order to address the deficiencies noted in this research, we proposed two feature extraction approaches, i.e, 14-dimensional feature extraction approach in Chapter 4 and 13-dimensional feature extraction approach in Chapter 5. After feature extraction, the clustering of these feature vectors is performed in order to cluster the similar genome sequences into one group. Nowadays, different clustering algorithms, such as K-means [30] and Fuzzy c-means [31], are available; nonetheless, these clustering approaches function best with static data. These methods cannot conduct clustering on dynamic real-time data. Today, plant genomics generates a large volume of dynamic real-time data [91], driven by

diverse plant developmental stages and inherent genetic diversity across species. The analysis of this data provides a deeper understanding of genetic variations, mutations, and evolutionary patterns. So, in order to perform the analysis of this dynamic data, incremental clustering is a widely used technique [54, 92]. Hence, in the subsequent section, we provided an exhaustive examination of the most recent advancements in incremental clustering techniques.

2.3 Incremental Clustering for Dynamic Data Analysis

Incremental clustering [45] is an effective method for dynamically evaluating changing data streams. Unlike classic clustering approaches, incremental clustering responds to incoming data in real-time, allowing for the smooth integration of new information without having to reprocess the entire dataset. This method is very useful in dynamic contexts where the data is continually changing. Incremental clustering enables immediate insights by effectively upgrading cluster structures as new data enters, making it a suitable choice for applications for applications where time is an important factor in usability, such as online streaming analytics, fraud detection, field-based research, and continuous monitoring. Its capacity to manage changing datasets distinguishes it as a flexible tool for dynamic data analysis. The working of incremental clustering is shown in Figure 2.10. Some techniques of incremental clustering are presented subsequently.

In 2019, Peng Zhou et al. [93] proposed a scalable incremental multi-view clustering. In this approach, they integrated the views incrementally rather than ensemble all views at once. They initially created a model with a small subset of views, and when a new view comes incrementally, they just updated the created model. They tested their method on four benchmark datasets: sun397, UCI digit, AwA (Animal with Attributes), and corel. Also they tested their method on two time series datasets, i.e., gas sensor and condition monitoring of hydraulic systems. They found that their approach performed better than the other state-of-the-art approaches. In 2021, Ling

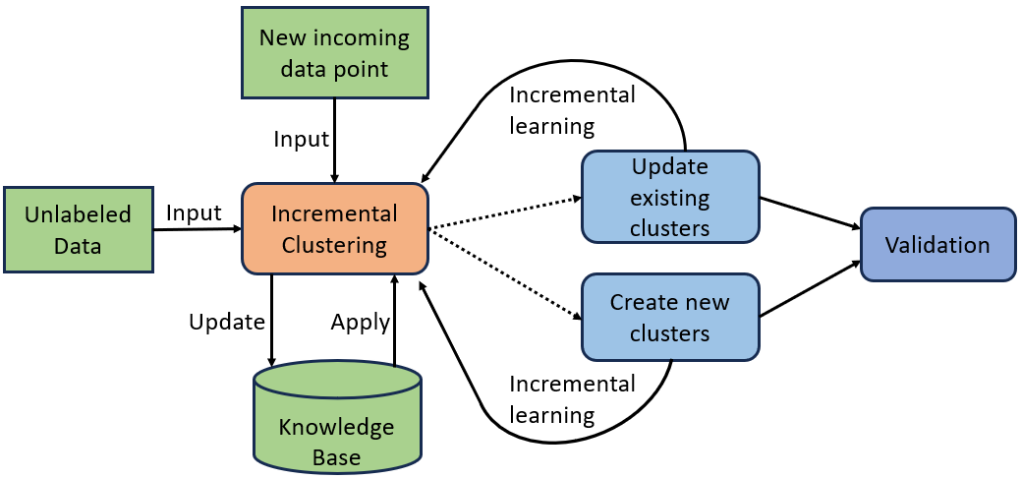


Figure 2.10: Working of incremental clustering

Wang et al. [94] proposed an incremental clustering technique for the time series data. In this approach, they divided the clustering process into two steps. In the first step, they used a fuzzy cluster validity index to automatically figure out how many initial clusters will be best for the clustering of the subset of data. Later, in the second step, they added the new data points to the existing clusters in small steps. In 2021, Sirisup et al. [95] came up with the idea of incremental density-based clustering that uses fuzzy-based incremental clustering as the one-pass scheme. In this approach, they used the modified version of the valley-seeking algorithm to improve the clustering performance and simplify the parameter-choosing process. They tested their method on static and dynamic data, and found that it outperformed the other state-of-the-art methods. In 2022, Amber et al. [55] later came up with the incremental online K-means clustering method for color quantization. This method fixed the problems that other incremental K-means clustering methods had with speed and initialization without making the algorithm more complicated. For color quantization, they used a binary splitting formulation of MacQueen's online K-means clustering. They tested their approach on various public image datasets and found that it was significantly faster than the previously proposed K-means clustering approaches. However, this approach yielded similar results to the previously proposed K-means clustering approaches.

Most of the above-mentioned techniques use a similarity-based metric and a single

objective to cluster dynamic data points together [46]. As a result, most clustering methods are sensitive to changes in the shape, size, dimensionality, and other features of clusters. Multi-objective clustering is utilized to address this issue. It partitions a dataset into groups of comparable objects while optimizing many objectives at the same time. Multi-objective clustering is a subset of multi-objective optimization [96] that seeks optimal trade-offs between goals while adhering to restrictions. In 2019, Tien et al. [97] came up with a fuzzy method that uses evolutionary multi-objective optimization based fuzzy method to find overlapping clusters in real-time data. In this method, they optimized the community centers by using a specially tailored multi-objective evolutionary algorithm. They tested their approach on various synthetic and real-time data, but not on dynamic data. In 2018, Zareizadeh et al. [98] came up with a way to cluster genes using clonal selection and multiple objectives. In this method, they combined the two cluster validity indices as the two conflicting objective functions and used a new way to update the population and to quickly arrive at a solution. They tested their approach on a variety of microarray datasets and found that their approach performed better than the other gene clustering approaches. In 2022, Weimin Li et al. [99] proposed a multi-objective optimization strategy by combining the three main characteristics of a dynamic social network, namely continuity, temporal variability, and stability, to detect the community structure. In the initial phase of the algorithm, probability fusion was used to create the appropriate network partition and to assure rapid convergence. Later, they proposed two neighbouring fusion methods, named neighbour swarm and neighbour diversity. On numerous real-world and synthetic datasets, they discovered that their method outperformed numerous state-of-the-art methods. Later, Ye Tian et al. [100] proposed a novel evolutionary algorithm for super large scale multi-objective optimization problems (SLMOPs) based on fast clustering. In this technique, similar decision variables were grouped using the fast-clustering method, and a single variable from each group was used to reduce the search space. In a later phase, they proposed an evolutionary approach in which Graphics Processing Unit (GPU) computations were utilised to accelerate the computations. On a variety of benchmark and real-world problems with more than one million decision variables,

they found that their approach outperformed the numerous existing state-of-the-art approaches. However, this method is incapable of handling incremental data.

Bejarano et al. [101] proposed a clustering-based technique for analyzing the pareto-optimal front in multi-objective optimization problems. In this procedure, K-means and Fuzzy c-means clustering were used to separate the optimal pareto optimal front to improve the quality of the solutions discovered. They employed eight standard test functions, four of which have a continuous pareto optimal front and four of which have a non-continuous pareto optimal front. The main disadvantage of this strategy is that it necessitates the use of additional parameters and computations. Later on, Sivadi Balakrishna [54] came up with a multi-objective based incremental clustering by fast search (MOC-FS) technique to create and update clusters in real-time. In this approach, they considered three objective functions, i.e., intra-cluster distance, inter-cluster distance, and cluster density to be optimized for the purpose of dynamic clustering. They tested their approach on the four IoT benchmark datasets: CRAWAD, BWS-AS, minute-weather data, and linked sensor. They found that their approach performed better than the MCFS [102], HCFS [103], and HOCFS [104] approaches. The problem with this method is that in the optimization process, all objective functions, whether they are meant to be maximized or minimized, are added together with positive coefficients.

Based on the above research, we can say that multi-objective optimization-based approaches produce better clustering results because they optimize multiple goals at the same time. Additionally, incremental clustering is the prominent approach for dealing with dynamic data. Hence, we proposed a novel incremental clustering based on multiple objectives in Chapter 6 in order to handle the real-time dynamic data generated from various agricultural research institutions.

Genomics is inherently a Big Data domain, given the massive volumes of genetic information it encompasses [105]. Therefore, there is a crucial need to adapt and improve the developed algorithms using Big Data framework to deal with this massive amount of data. Consequently, an extensive exploration of Big Data frameworks is offered in the subsequent section.

2.4 Big Data Processing Framework

Big data [106] refers to huge amounts of data created from various sources at rapid speeds and in diverse forms. It contains both structured and unstructured data, such as text, photos, videos, sensor data, and social media interactions. Today, vast amounts of data are generated on a daily basis from a variety of sources, such as health, government, social networks, marketing, and finance. This is because of a number of recent developments in technology, such as the explosion of cloud computing [107], the Internet of Things (IoT) [108] and the widespread use of smart devices [109]. In 2012, The Internet Data Center (IDC) [110] predicted that digital data would rise 300 times between 2005 and 2020, from 130 exabytes to 20,000 exabytes [111]. In a recent white paper, IDC forecasted that by 2025, digital data would expand by 175 zettabytes [112]. As the amount of Big Data increases from various sources, there is a requirement for authentic research with thorough inquiry in Big Data analytics to extract insights from the important information included inside Big Data.

Plant genomics is an important source of Big Data. The science of plant genomics creates huge amounts of data, mostly through high-throughput sequencing methods like Next-Generation Sequencing (NGS) [113]. These technologies quickly sequence DNA from plants, producing huge datasets of genomic sequences such as entire genomes, transcriptomes, and epigenomes. Furthermore, progress in Genome-Wide Association Studies (GWAS) and population genomics adds to large-scale genomic datasets by looking at genetic differences in different plant groups. The amount of genomic data is also increased by using methods like chromatin immunoprecipitation [114] sequencing (ChIP-seq) and DNase-seq [115], which give useful information about how chromatin is accessed and how DNA and proteins interact. The use of multi-omics methods, such as genomics, Transcriptomics, and Epigenomics, increases the complexity and amount of data in plant genomics research. Nowadays, biologists aren't using traditional labs to find new biomarkers for diseases. Instead, they now rely on extremely large amounts of genomic data that are provided by various sources. Plant genomics is entering into a new era of Big Data as technologies like automated

genome sequencers and genome clustering become more affordable and efficient. Clustering is a widely used data mining approach that is applied to analyse the genetic data in the field of plant genomics. Clustering aids in the classification of genes, regulatory elements, and functional regions by categorizing similar sequences, as genomes contain vast amounts of data. Determining the family to which a newly sequenced genome belongs is of constant interest to biologists [116]. This will allow researchers to learn how this genome evolved and what biological roles it plays. To analyze raw genomic data using clustering, it is necessary to create multiple feature extraction algorithms that convert raw genome sequences into feature vectors. The growing volume of genetic data has exerted substantial strain on feature extraction methodologies. In order to surpass constraints in both space and time, there is a requirement to expand their ability to work beyond a solitary device. In order to accommodate the large-scale genome data, it is necessary to employ Big Data management systems to optimize the performance of feature extraction algorithms.

In recent times, numerous sophisticated processing frameworks such as, Map Reduce [117], Apache Flink [118], Apache Hadoop [119], Apache Storm [120], Apache Samza [121], and Apache Spark [122, 123, 124] have been specifically developed for the purpose of handling Big Data [125, 126, 127, 128]. These frameworks are typically classified based on their data processing approaches, such as batch processing and stream processing. Batch processing involves processing large volumes of data in discrete chunks or batches, on the other hand, stream processing analyzes data continuously in real-time as it arrives, enabling near-instantaneous insights and actions. Classification of these framework according to data processing approach is shown in Figure 2.11.

Apache Spark outperforms other big data frameworks due to its in-memory processing, diverse APIs for batch, stream, and machine learning operations, and fault-tolerant architecture, which make it easy to use and integrate into existing systems while maintaining high performance. Spark is constructed based on Hadoop's data volume paradigm, specifically the Hadoop Distributed File System (HDFS). Apache Spark is an ideal choice for delivering an application that utilizes the MapReduce ap-

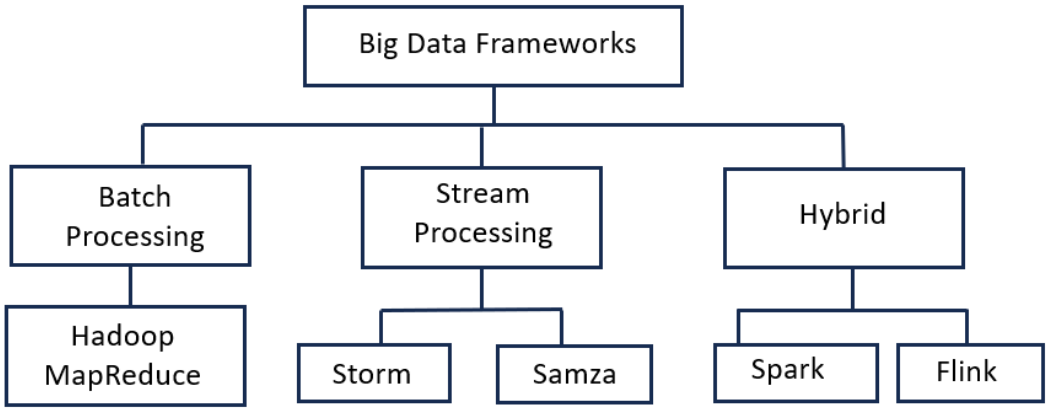


Figure 2.11: Classification of Big Data processing frameworks

proach. The development of this technology originated at the University of California in 2009. Spark exhibits a performance that is up to 100 times swifter than Hadoop MapReduce and notably quicker than other frameworks. It is made for developers to perform complex tasks on large datasets quickly. With an API that is easy to use, Spark can help the user in extensive data analysis and parallel computing by reducing the grunt work. Many times during the handling of data, we tend to partition the data into various parts to make the computation process easier. Spark follows the same principle to distribute tasks across different worker nodes during data shuffling. If the partitions are less in number, larger blocks of data are given to each node. Hence the work will be divided efficiently in parallel manner.

As previously mentioned in Section 1.2, it is proposed to design of scalable feature extraction techniques to handle the huge amount of genomics data using Apache Spark. Hence, the working of Apache Spark framework is detailed next.

Working of Apache Spark

Apache Spark employs a tiered design in which all Spark components and layers are loosely coupled. It is organized into three main layers: upper-level libraries, Spark core, and cluster management. It also contains a storage layer and other features, such as Resilient Distributed Dataset. Apache Spark uses Hadoop to retrieve data from Spark engines [129]. Figure 2.12 illustrates the layered architecture of Apache Spark used in our experiment. The following explanation gives a comprehensive overview of

its layers and features.

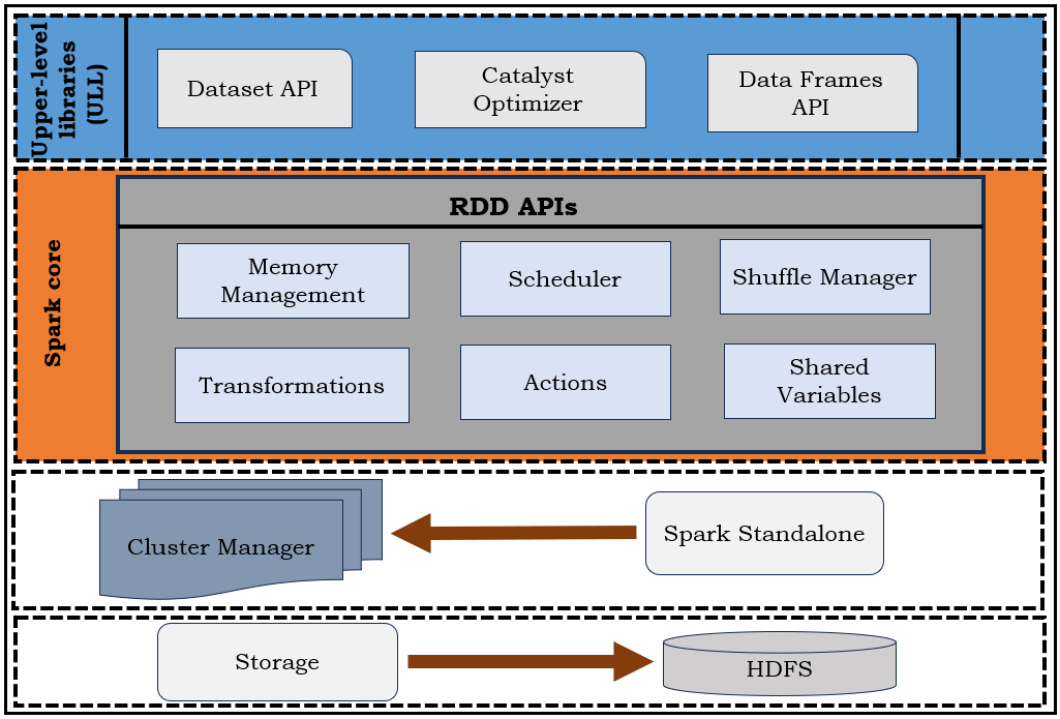


Figure 2.12: Layered architecture of Apache Spark

- Upper-level libraries: Spark offers a number of upper-level libraries [53], including MLlib, GraphX, Spark Streaming, SparkR, and Spark SQL, which add specialized functionality to the Spark core. These libraries are intended to simplify and improve specific data processing tasks.

- Spark core: The Spark core is compatible with many cluster managers and allows for seamless integration with Hadoop data sources. It provides an intuitive programming interface known as Resilient Distributed Dataset for efficiently processing large datasets. The Spark core is integrated with the Scala programming language, but it also supports APIs in R, Python, and Java. For our implementation, we used the Python API. Furthermore, the Spark core provides essential functionalities for in-memory cluster computing, including job scheduling, data shuffling, error recovery, and memory management [130].

- Cluster managers: The use of a cluster manager simplifies the acquisition of

cluster resources required for job execution. The Spark engine is intended to work with its native cluster manager, known as a standalone. This built-in cluster manager is in charge of efficiently managing resource allocation and sharing across multiple Spark applications. It ensures that resources are efficiently used and distributed across clusters, resulting in peak performance and resource utilization for Spark applications.

- Storage: Spark does not have its own storage mechanism. It works with any Hadoop-compatible data source, including HBase, HDFS, Casandra, and Hive [34].

- Resilient Distributed Dataset (RDD): The concept of RDD serves as the foundation for Spark core. RDDs are read-only collections of distributed records. They provide fault tolerance and function as a parallel data structure, allowing for explicit data storage on disk or memory. RDDs give users control over data partitioning and enable manipulation via a comprehensive set of operators. This enables us to efficiently distribute data across computations to meet various workload requirements. There are two ways to create RDDs: parallelizing an existing collection in the driver program or accessing a dataset in an external storage system, such as a shared filesystem, HDFS, HBase, or any data source that supports a Hadoop input format [122]. Parallelized collections are created by invoking SparkContext's parallelize method on an existing collection in the driver application. The pieces of the collection are duplicated to form a distributed dataset that can be processed concurrently. PySpark can generate distributed datasets from several Hadoop-compatible storage sources, including the local file system, HDFS, Cassandra, HBase, Amazon S3, and others. Spark supports a variety of file formats, including text files, sequence files, and any other input format used in Hadoop. Text file RDDs can be created using SparkContext's textFile method. This function takes a Uniform Resource Identifier (URI) for the file, which can be a local file path on the machine or a URI like hdfs://, s3a://, and so on. It reads the file and treats the contents as a series of

Table 2.1: Operations performed in Apache Spark execution

Operation	**Description**	**Spark code**
Creation of RDD	Parallelizing an existing collection in the driver program	data = sc.parallelize([“a”, “b”, “c”])
	Accessing a dataset in an external storage system	data = sc.textFile(“data.txt”)
Transformations	Transforming one RDD into another RDD using map( )	t_data = data.map(lambda x: (x, 1))
Aggregations	Aggregating the data using reduceByKey( )	a_data = t_data.reduceByKey(lambda a, b: a + b)
Action	Collecting the final result to the driver program	result = a_data.collect()

lines.

There are two categories of operations that can be executed on a RDD:

1 Transformations: Transformations are operations performed on a RDD that yield another RDD.

   – Map: A map is a function that performs a transformation operation in Apache Spark. This operation is applied to every element of the RDD and the result is returned as a new RDD. In Spark, the map function operates on individual elements by applying custom code provided by the developer, resulting in the generation of one element at a time. The map function applies a transformation to each element of an RDD, resulting in a new RDD of the same length. The input and output RDDs will generally contain an equal amount of records.

   – Aggregations: Aggregations include calculating statistics for a dataset that contains key-value pairs, where the statistics are calculated for

all elements that have the same key. Spark utilizes a reduceByKey function to combine data with the same key as part of a set of operations. This process consolidates the values for each key by employing a related reduction technique. This functionality is only applicable to RDDs that consist of key-value pairs. In this process, an associative function is provided as a parameter, which can be linked to the source RDD and will generate a new RDD with the resulting key-value pairs. Table 2.1 presents the operations involved in the execution of Apache Spark.

2 Actions: Transfers all data from the RDD to the driver application, allowing for computation and providing the result directly to the driver. The "collect" is an action that retrieves all elements of an RDD from the worker nodes to the master node and returns them as an array or list in the driver program. It's used for bringing the distributed data back to the driver program for further processing or for displaying results.

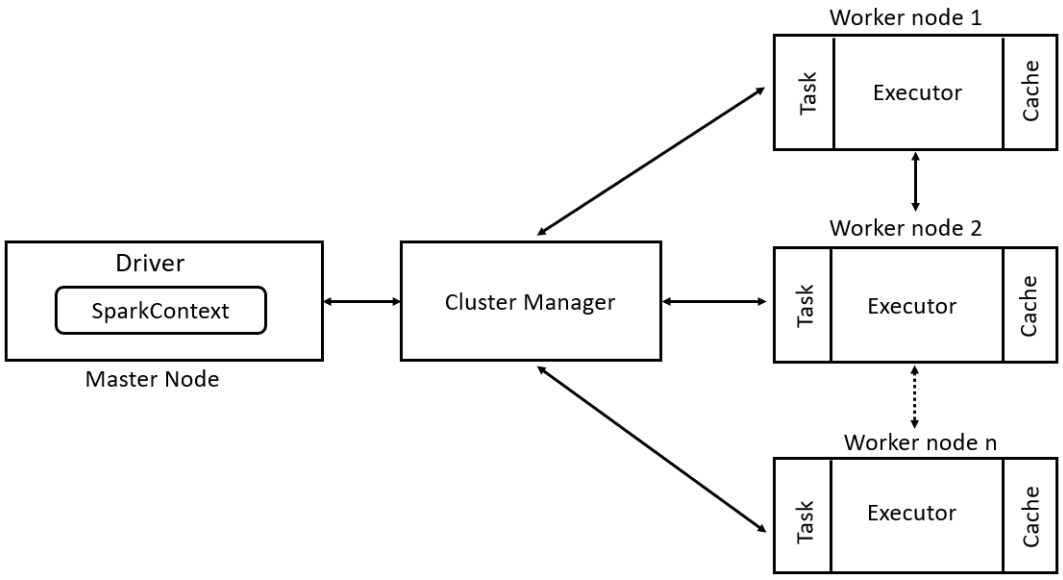


Figure 2.13: Cluster application of Apache Spark

- Spark-application: In the context of running a Spark application, there are five key components: a cluster administrator, a controller program, tasks, executors,

and workers as shown in Figure 2.13. The driver program, which functions as an application and uses Spark as a library, determines the highest-level flow of control for the intended computation. On the other hand, a worker provides memory, storage resources, and CPU to a Spark application. Spark creates a Java Virtual Machine (JVM) process, known as an executor, on each worker to complete tasks for the associated application. Meanwhile, program results are obtained by running a set of computations, known as a job, on a cluster managed by the Spark controller. Multiple jobs can be launched within a Spark application. To efficiently manage the workload, Spark divides it into stages of a Directed Acyclic Graph (DAG), with each stage consisting of a set of tasks. Spark assigns the smallest work units, known as tasks, to executors for execution. The Spark context is the primary interface for Spark functions, allowing communication between the driver program and Spark. It connects to the computing cluster, allowing the driver program to interact with Spark's distributed computing resources [53].

Based on the preceding discussion, it can be inferred that Big Data offers an efficient solution for swiftly managing massive datasets. Furthermore, the Apache Spark framework stands out for its capability of in-memory computation, rendering it particularly notable. Consequently, to attain high performance while maintaining accuracy, we expanded our feature extraction methodologies using the Apache Spark framework to accommodate the substantial volume of genomic data. As a result, we devised two scalable feature extraction methods based on Apache Spark, detailed in Chapters 4 and 5.

This thesis explores the development of novel feature selection, scalable feature extraction, and incremental clustering approaches. The performance evaluation measures used to assess the efficacy of these approaches are detailed below.

2.5 Performance Evaluation Measures

Various research studies [1, 50, 53, 54] have employed both external and internal evaluation measures to assess the performance of clustering-based feature selection, feature extraction, and incremental clustering. So, in order to have the comparison, we also evaluated the performance of our proposed approaches using the external and internal evaluation indexes. External metrics evaluate the performance of clustering based on how well it fits the ground truth or expert categorization. Therefore, to evaluate the external evaluation metric, the labels of the dataset are required. Some of external evaluation metrics are Rand index, Normalized Mutual Information, and Jaccard Index. On the other hand, internal evaluation metrics, evaluate the clustering performance based on the degree of fit between the formed clusters and the data itself. These metrics do not utilise the labels of the dataset. Some of the internal evaluation metrics are the Silhouette Index, the Calinski-Harabasz Index, and Davies-Bouldin Index.

2.5.1 External Evaluation Measures

We used three external evaluation measures, i.e., Rand Index, Normalized Mutual Information, and Jaccard Index to evaluate the performance of proposed approaches on the labeled datasets. The details of these metrics are provided subsequently.

Rand Index (RI)

The RI [131] evaluates the proportion of accurate decisions. In clustering, an accurate decision means, if two data points are similar, they should be clustered together; if they are unlike, they should be clustered apart. The RI ranges from 0 to 1 and a high value of RI represents better clustering. The RI is computed using Eq. (2.1).

$$Rand\,index = \frac{Number\,of\,accurate\,decisions}{Total\,number\,of\,decisions}. \tag{2.1}$$

Normalized Mutual Information (NMI)

The idea of NMI [132] accounts for two distinct types of partitioning: (1) partitioning according to clusters, and (2) partitioning according to classes. Next, it explains how

much the two partitions are compatible with one another (how you know about one of them if you know the other one). The NMI is calculated using Eq. (2.2).

$$NMI = \frac{2 * I(L_{class}; L_{clus})}{H(L_{class}) + H(L_{clus})}. \tag{2.2}$$

Where L_{class} and L_{clus} is the class labels, and cluster labels respectively. $H()$ is a function to calculate the entropy and $I(L_{class}; L_{clus})$ represents the mutual information between L_{class} and L_{clus}.

Jaccard Index (JI)

JI [133] is a metric that can be used to assess the effectiveness of any clustering strategy. The effectiveness of clustering strategy is measured on the basis that how closely it matches to the ground truth or expert classification. No match between clustering and ground truth is represented by a JI value of 0 and a perfect match is shown by an index value of 1. JI is computed by Eq. (2.3).

$$JI = \frac{n_{11}}{n_{11} + n_{01} + n_{10}}. \tag{2.3}$$

Where n_{11} is the number of pairs of instances classified together by the clustering algorithm and expert classification, n_{01} is the number of pairs of instances classified together by the clustering algorithm but in a different class by expert classification, and n_{10} is the number of pairs of instances classified together by expert classification but in different classes by clustering.

2.5.2 Internal Evaluation Measures

We used three internal evaluation measures, i.e., Silhouette Index, Calinski-Harabasz Index, and Davies-Bouldin Index to evaluate the performance of proposed approaches on the unlabeled datasets. The details of these metrics are provided subsequently.

Silhouette Index

Silhouette index (SI) [67] is extensively used as a metric for measurement and analysis in unsupervised learning. It is determined by data points cohesiveness with the other

data points in that cluster and its separation from the cluster that is closest to it. It ranges from -1 to 1. A high value of SI represents that well defined clusters are formed. The SI is computed by calculating the average Silhouette coefficient of all data points.

For a m^{th} data point d_m, the Silhouette coefficient SC_{d_m} is computed using Eq. (2.4).

$$SC_{d_m} = \frac{o_{d_m} - s_{d_m}}{maximum(o_{d_m}, s_{d_m})}. \tag{2.4}$$

Where o_{d_m} denotes the mean distance of data point m to all other data points in the nearest cluster and s_{d_m} denotes the mean distance of data point m to all other data points in the same cluster.

Calinski-Harabasz Index (CH Index)

The CH Index [134] measures that how similar a data point to its own cluster known as cohesion compared to other clusters known as separation. The higher value of CH Index means the clusters are dense and well separated, although there is no acceptable cut-off value. The CH Index is calculated using Eq. (2.5).

$$CH_{index} = \frac{trace(S_c)}{trace(S_d)} * \frac{N - K}{K - 1}. \tag{2.5}$$

Where $trace()$ represents the trace of the scattering matrix. S_c is the average covariance of each cluster and S_d represents the covariance of the data set whose members are the cluster means. N and K are the number of data points and the number of clusters, respectively.

Davies-Bouldin Index

The Davies-Bouldin Index (DBI) [89] is a measurement used to assess the accuracy of clustering results. The DBI is calculated by calculating the dissimilarity between data points within each cluster and the dissimilarity between the centroids of various clusters. Its value ranges from 0 to infinity. A low DBI implies that the clusters are compact and well-separated. The DBI value should be as low as possible for the best clustering outcome. The DBI is computed by using Eq. (2.6).

$$DBI = \frac{1}{m}\sum_{x=1}^{m} maximum_{b \neq a}\left[\frac{Z_a + Z_b}{d(clus_a, clus_b)}\right]. \quad (2.6)$$

Where m is the number of clusters, and $d(clus_a, clus_b)$ represents the dissimilarity between the centroids of cluster a and b. Z_a and Z_b denotes the average dissimilarity between the data points within cluster a and b, respectively. The SI, CH Index, and DBI are used to assess the effectiveness of proposed techniques on unlabeled datasets. The next section describes the genome datasets used in our experiments.

2.6 Real-life Genome and Protein Data

In the field of plant genomics, there are primarily two types of data generated: genome data [135, 136, 137] and protein data [138, 27]. Genome data encompasses the entirety of genetic information that is encoded within an organism's DNA. The genetic material encompasses the essential directives required for the growth, maturation, operation, and procreation of an organism. Genome data consists of the complete arrangement of nucleotides, the fundamental units of DNA, in a specific order along the chromosomes. The study of genome data has changed many areas of biology and medicine by giving us new ideas about how traits, diseases, and the evolutionary connections between species are based on genes [139]. The study of genome data entails sequencing, assembling, and annotating DNA sequences to discover genes, regulatory elements, and genetic variants. Genome data is used as a starting point for many scientific projects, like studying the evolution of species, learning genetic diseases, and making personalized medicine.

On the other hand, protein data [138], which is made up of amino acid patterns, is the foundation of life and plays an important part in biological processes. Proteins are essential macromolecules that perform a variety of functions within cells, including speeding up chemical processes, supporting structure, allowing cells to communicate with one another, and transporting chemicals across cell membranes. The sequence of amino acids in a protein determines its shape and function. Each amino

acid contributes a distinct set of properties to the protein. The sequence of amino acids in proteins determines how they fold into distinct three-dimensional forms [140]. This structure determines how proteins function biologically and interact with other molecules. Protein data, particularly amino acid sequences, are essential for understanding how biological processes and circumstances operate at the molecular level. Analyzing protein sequences allows scientists to learn more about how proteins work, how species change, and how illnesses work. Protein data is also useful for discovering and developing new medications, as many treatments target specific proteins implicated in disease processes.

Advances in sequencing technology have resulted in exponential growth of plant genomic and associated data. Annotating sequenced genomes yields many potential protein-coding sequences. Sequenced land plants have a range of 25,000 to over 90,000 putative genes. Additionally, large-scale sequencing projects generate massive amounts of protein and DNA sequence data. Currently, over 100 plant genomes projects have been completed or are close to completion, with over 200 ongoing projects, which leads to exponential growth of genetic data. As a result, it is projected that during the next ten years, genomics will generate an annual data volume ranging from 2 to 40 exabytes [141]. The utilization of computer databases is increasingly prevalent in the organization of the extensive quantities of biological data presently accessible, with the aim of facilitating researchers in locating pertinent information.

The advent of high-throughput sequencing technologies has led genome researchers into the era of machine learning, where the research approach has transitioned from hypothesis-driven to data-driven. The incorporation of machine learning and Big Data techniques like feature selection, incremental clustering, and scalable feature extraction in genomics offers new avenues for research, fundamentally transforming how genome data is analyzed and interpreted. The next section provides a comprehensive overview of the genome, SNP, and protein dataset utilized in our experimental investigation.

2.6.1 Wheat, Rice, and Soybean Genome Dataset Description

The set of instructions that make up an organism is called the genome, and it is made of DNA. DNA is made up of four nucleotide bases: Adenine (A), Cytosine (C), Guanine (G), and Thymine (T). We conducted the experiments using four plant genome datasets of rice crop, seven datasets of soybean crop, and one dataset of wheat crop obtained from rice genome library[1] [51], soybase repository[2] [49], and Han et al. [52], respectively. The rice dataset includes the genome sequence of the Nipponbare rice subspecies as well as annotations for the 12 rice chromosomes. To create four datasets, we merged three sets of chromosomes (ch). In addition, we collected two labeled benchmark gene sequence datasets from UCI machine learning repository [48] named splice and promoter. The detailed description of these datasets are presented subsequently.

Rice 1: To create the Rice 1 dataset, we combined the gene sequences from three chromosomes (ch), namely ch1, ch2, and ch3. This dataset consists of 21,701 gene sequences.

Rice 2: In this dataset, we integrated gene sequences from three chromosomes: ch4, ch5, and ch6. This collection contains 17,116 gene sequences.

Rice 3: In this dataset, we combined gene sequences from three chromosomes: ch7, ch8, and ch9. This collection includes 13,797 gene sequences.

Rice 4: In this dataset, we integrated gene sequences from three chromosomes: ch10, ch11, and ch12. This collection contains 13,340 gene sequences.

Williams82: Williams 82 (Wm82), a type of soybean used to make the reference genome sequence, was made by switching the Phytophthora root rot resistance locus from the donor parent Kingwa to the recurrent parent Williams [142]. Wm82.a1, Wm82.a2, and Wm82.a4 are different versions of Williams82. The number of sequences in Wm82.a1, Wm82.a2, and Wm82.a4 are 73,320, 88,647, and 88,256, respectively.

Lee.a1: In the southern United States and Brazil, the Lee.a1 strain is cultivar soy-

[1] http://rice.uga.edu/
[2] https://www.soybase.org/

bean, which is a hybrid between the Chinese lines CNS and S-100, is employed as a parent in numerous breeding programmes. The details about Lee.a1 can be found in Wysmierski et al. [143]. The number of sequences in this dataset are 71,358.

ZH13.a1: The Glycine max ZH13.a1 is an assembly of high quality chinese cultivated soybean. The details of ZH13.a1 are given by Shen et al. [144]. The number of sequences in this dataset are 57,978.

PI483463.a1: The Glycine soja accession PI483463.a1 [145] contains Illumina sequence from NRGene. This line was chosen due of its high degree of genetic dissimilarity to cultivated soybean. It comes from Shanxi Province, which is in the middle of northern China. The number of sequence in this dataset are 55,161.

W05.a1: As a reference genome assembly, the genome of Glycine Soja accession W05.a1 [146], a salt-tolerant wild soybean, was constructed. In genetic studies, the W05.a1 affiliation has been utilised to examine a variety of characteristics, including uncertainty, seed size, pod count per plant, and seed colour. The number of sequences in this dataset are 89,477.

Splice: This dataset contains the primate splice-junction gene sequences [48] having three types of labels, named EI, IE, and N. We converted these symbolic labels to the numerical form of 0, 1, and 2.

Promoter: This dataset contains promoter gene sequences [48] having two types of labels, positive and negative. We converted these symbolic labels to the numerical form of 0 and 1.

The genome datasets discussed in this subsection are used for feature extraction and clustering. Clustering plant genome sequences can help to identify unique and new genes to improve crop production with higher yields, drought resistance, improved crop quality, and provide better suggestions to find a cluster of diseases. In the next subsection, we discussed about the SNP datasets used in our experiments.

2.6.2 Rice SNP Dataset Description

SNP [147] refers to a variation (deletion/addition) in a single nucleotide at a specific position in the genome and is sometimes abbreviated as SNP, snip, or snips. The

details of these datasets are discussed subsequently.

SNP-seek rice: SNP-seek rice data includes rice chromosomes from 1 to 12[3]. In order to perform clustering on a large SNP dataset, we integrated all of the rice chromosomes from ch1-12 into a single file. Detailed analysis of the SNP-seek rice data is given by Mansueto et al. [148]. The size of this dataset is 16.3 MB.

MAGIC-rice: The MAGIC-rice dataset is composed of SNP sequences[4]. The MAGIC rice dataset contains 1,411 samples organized in 12 files (for each chromosome). To perform clustering on a massive SNP dataset, we integrated all chromosomes from 1 to 12 to create the MAGIC-rice dataset. The details of Magic-rice dataset can be found in Bandillo et al. [149]. The size of this dataset is 1.03 GB.

248Entries rice: The 248Entries rice[5] consists of 248 data samples composed of indica and aus genotypes. The details of 248Entries rice dataset is given by Dilla-Ermita et al. [150]. The size of this dataset is 30.8 MB. This dataset is made up of 40,840 SNPs of rice crop.

The SNP datasets discussed in this subsection are used for feature selection and incremental clustering purpose. Moreover, to perform the analysis of the protein datasets we collected the soybean amino acid sequences. The details of protein dataset is as follows.

2.6.3 Soybean Protein Dataset Description

Protein sequence [151] is made up of twenty amino acids, which includes Alanine (A), Cysteine (C), Aspartic acid (D), Glutamic acid (E), Phenylalanine (F), Glycine (G), Histidine (H), Isoleucine (I), Lysine (K), Leucine (L), Methionine (M), Asparagine (N), Proline (P), Glutamine (Q), Arginine (R), Serine (S), Threonine (T), Valine(V), Tryptophan (W), and Tyrosine (Y). The amino acids inside a sequence can be joined together in any arrangement, and the protein sequences can vary in

[3] https://s3-ap-southeast-1.amazonaws.com/oryzasnp-atcg-irri-org/osnp_legacy/diversity_rice31.oryzasnp.hapmap.tar.gz

[4] https://s3-ap-southeast-1.amazonaws.com/oryzasnp-atcg-irri-org/pub-data/MAGIC-Raw-genotype-data-Raghavan-2017.zip

[5] https://s3-ap-southeast-1.amazonaws.com/oryzasnp-atcg-irri-org/pub-data/248Entries_40840SNPs_inorder_21May2015_v2.zip

length. The process of clustering protein sequences is crucial for the identification of functional linkages, the grouping of proteins with comparable characteristics, and the understanding of evolutionary trends. The description of the protein dataset used in our experiments is as follows.

Glycine Soja accession W05 protein dataset: The genome of Glycine Soja accession W05, a salt-tolerant wild soybean, was created to serve as a reference genome assembly. In genetic studies, the W05 affiliation has been used to study several qualities like uncertainty, seed size, the count of pods per plant, and seed colour. The details of W05 were given by Xie et al. [146]. The number of amino acid sequences in this dataset are 66622. The protein data discussed in this section is used for the feature selection.

Chapter 3

A Novel Clustering-Based Hybrid Feature Selection Approach using Ant Colony Optimization

In this chapter, a novel clustering-based hybrid feature selection approach using Ant Colony Optimization (NCHFS-ACO) to handle the high dimensionality issue of unlabeled data is proposed. This novel approach selects features randomly and uses K-means clustering to assign the fitness of features in terms of Silhouette Index (SI) along with the Laplacian score in the feature selection process. The proposed feature selection approach allows random selection of features, which allows a better exploration of feature space and thus avoids the problem of being trapped in a local optimal solution and generates a global optimal solution. Experimental results indicate that the proposed method outperforms other state-of-the-art methods on ten benchmark datasets extracted from the UCI machine learning repository in terms of SI (internal evaluation measure) and Jaccard Index (JI) (external evaluation measure).

3.1 Introduction

In today's world, various fields such as bioinformatics [3], medicine [2], financial studies [4], environmental monitoring [6], and many more [8, 9] generate high-dimensional data. In these contexts, feature selection becomes indispensable for iden-

tifying the most important and relevant features while discarding redundant and unimportant ones. Feature selection assists in enhancing prediction accuracy, reducing computation time, and creating more comprehensible models. In feature selection, each feature has two possibilities, either it would be taken for computation or not, which implies for n number of features, there are 2^n possible feature subsets. So, identifying a relevant feature subset in a reasonable amount of time is an NP-hard problem [152], but by using an approximation algorithm, a near-optimal solution can be achieved. However, many of the feature selection algorithms use a sequential search strategy to select relevant features, which adds or removes features from the dataset sequentially and leads to being trapped into a local optimum solution. In addition, many approaches in hybrid feature selection employ a filter measure during the initial filtering stage and a wrapper measure in subsequent stages. While this method often yields satisfactory results, it may not always produce an optimal solution. Furthermore, some approaches adopt a greedy strategy for feature selection, prioritizing features based solely on their individual fitness without considering the broader context. However, this myopic approach can lead to suboptimal solutions by prematurely discarding potentially valuable features and converging to local optima without exploring the full feature space. To overcome this lacunae, we proposed the novel approach named NCHFS-ACO to select the important and relevant features. The proposed approach combines the Laplacian score as well as the SI to measure the relevancy of a feature rather than using the Laplacian score in the filter stages and then the SI in the wrapper stage separately. The combination of the Laplacian score and the SI facilitates the selection of more relevant features by adopting the characteristics of both measures at the same time. In addition, proposed approach follows the behaviour of Temnothorax Albipennis ant species, which facilitates the preservation of most promising features throughout the computation without losing them. Moreover, the proposed approach introduces an element of randomness by selecting a subset of features at random during each iteration. This strategic inclusion of random features helps to prevent the algorithm from becoming ensnared in local optima, allowing it to explore a wider range of feature combinations and potentially discover more globally

optimal solutions. Since, the proposed approach selects features using clustering, it is applicable to both labeled and unlabeled data. The detailed methodology of proposed NCHFS-ACO is presented subsequently.

3.2 Proposed Clustering-Based Hybrid Feature Selection Approach using Ant Colony Optimization

The proposed approach is based on Ant Colony Optimization (ACO); hence, the brief overview of ACO is presented first in preliminaries. After that, we discussed the proposed approach.

3.2.1 Preliminaries

ACO [25] mimics the foraging behavior of ants when searching for a route between a food source and an ant colony. It was initially designed to address the well-known "travelling salesman" problem. Later, it was used to solve a variety of complex optimization problems like feature selection [75].

In this algorithm, when ants find food, they spray a smelly substance called pheromone on the ground to show where they are going. When other ants are on the lookout for food, they detect the pheromone and decide to follow the same path. A wandering ant also spreads pheromones along this path, which makes the path stronger and draws other ants to follow it. If ants have to choose between multiple routes, they choose the routes with high pheromones first. This means that more ants have moved from that route. This is because ants prefer shorter routes to feed their colonies and thus shorter routes receive more pheromones.

Another ant species, Temnothorax Albipennis [153], follows a tandem run technique in which an ant acts as a leader called a master, and other ants will be followers known as slaves. The leader ant knows the location of food and thus it controls the direction and speed of other slave ants. Using this approach, follower ant finds food

more quickly than searching alone. While following the leader, slave ants also collect surrounding information. If a follower ant explores a new path shorter than the leader ant path by using surrounding information, then in the next iteration follower will act as leader.

The simulation model is expressed by a completely connected undirected graph G = (V, E) having a one-to-one mapping between vertices and features. Hence the number of vertices (v_n) equals the number of features (f_n). V denotes the set of vertices as v_1, v_2, v_3....v_n and E denotes the set of edges (e_1, e_2, e_3....$e_{\frac{n(n-1)}{2}}$) joining any two vertices in the graph. In this model, the number of artificial ants (N_{ant}) is taken the same as the number of features (f_n) to avoid being trapped in the local optimum, so $f_n = v_n = N_{ant}$. In ACO, each artificial ant i constructs a feature subset (F_i). The N denotes the count of all feature subsets created by ants and n_{max} indicates the maximum number of features possible in each subset, then $N_{ant} = N$ and $0 \leq n_{max} \leq n$.

Each ant begins at a vertex and then proceeds to traverse different vertices to create a feature subset. The initial pheromone value (γ) for each feature is set to a constant. The notation n_{tan} represents the features selected by tandem run strategy.

3.2.2 Proposed Method

In this study, we proposed a novel clustering-based hybrid feature selection technique using ACO that uses the tandem run strategy to choose the best feature subset.

In this approach, the selection of n_{max} features is achieved in three steps. In the first step, n feature subsets are created by choosing n_{max} features randomly, and then on these subsets, we applied the K-means clustering. The efficacy of these subsets is evaluated in terms of SI value. The subset with the highest score is the leader subset ($g_{bestset}$). In the second step, n feature subsets are created differently, and the selection of n_{max} features is achieved in three parts. Some features are chosen randomly (n_{random}), whereas some features ($n_{arbitary}$) are selected with high pheromone and low Laplacian scores. On the other hand, some features (n_{tan}) are selected from the leader subset having high pheromone and low Laplacian scores. Later, in the

third step, these subsets are again applied to K-means clustering and the efficacy of these subsets is evaluated in terms of SI value. The subset with the highest SI value is known as localbest ($l_{bestset}$), and if localbest is greater than globalbest, then this localbest becomes globalbest ($g_{bestset}$) for the further iterations. Steps two to three are repeated till max iteration (max_{iter}). After executing all iterations, the globalbest set is the best subset generated that has the maximum SI value. The Pseudo-code for proposed NCHFS-ACO approach is described in Algorithm 3.1. The block diagram of the proposed algorithm is shown in Figure 3.1.

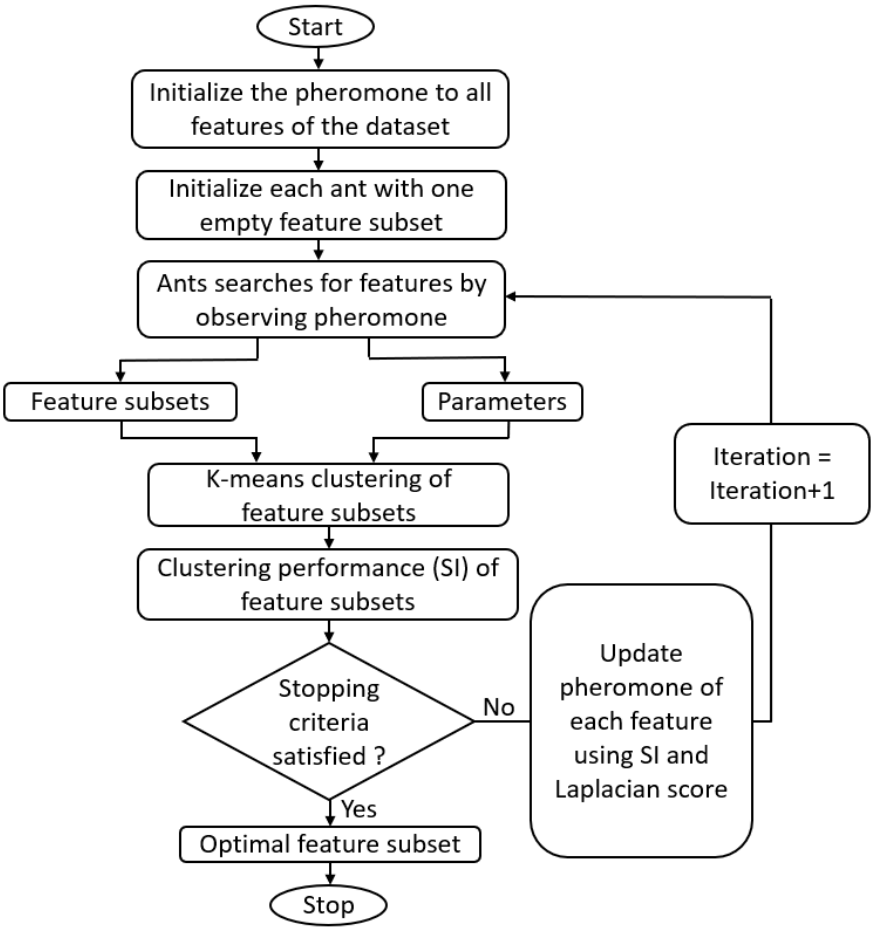


Figure 3.1: Flow diagram of NCHFS-ACO

The proposed NCHFS-ACO uses Laplacian score as filter measure. In terms of local preservation, the Laplacian score [154] is a good indicator. Many feature selection methods use a Laplacian score to rank features. A low Laplacian score indicates that a feature is superior. The weight matrix $W = \{w_{11}, w_{12}, w_{13}, ...w_{ij}, ...w_{mm}\}$ of size

Algorithm 3.1 NCHFS-ACO

Input: Dataset, n_{max}, max_{iter}
Output: Leader subset ($g_{bestset}$)

▷ **Initialization**

Equalize the number of ants with the number of features in the Dataset. So, $N_{ant} \leftarrow n$.
In ACO each artificial ant constructs a feature subset so $N \leftarrow N_{ant}$.
Initialize fitness of each feature with 0. So, $fitness_{f_i} = 0$ where $i \in [1, n]$.
Start all N_{ant} ants with N random solutions each with n_{max} features.
Initialize $t = 1$.
Initialize initial pheromone (γ_i) of each feature by $1/n$, where $i \in [1, n]$.

▷ **Calculation of heuristic value**

Calculate the huristic value for each feature h_i as given in Eq. (3.4). Where $i \in [1, n]$.

▷ **Feature subset construction and fitness evaluation**

while max_{iter} **do**
  for $p = 1$ to N_{ant} **do**
    $Fitness_{F_p} \leftarrow SI_p$ (SI_p is SI value of subset F_p using K-means clustering.)
    for $q = 1$ to n **do**
      if $f_q \in F_p$ **then**
        $Infitness_{f_q} \leftarrow fitness_{F_p}/n_{max}$. ($Infitness_{f_q}$ a variable corresponding to each f_q.)
      else
        $Infiness_{f_q} \leftarrow 0$
      end if
      $fitness_{f_q} \leftarrow fitness_{f_q} + Infitness_{f_q}$
    end for
  end for

▷ **Pheromone updation**

Refresh the pheromone values for each of the n features by $\gamma_i(t+1) = \gamma_i(t) + fitness_{f_i}$.

▷ **Current best subset $g_{bestset}$ computation**

Update the localbest set $l_{bestset} \leftarrow$ Subset with max(SI_p).
if $SI_{l_{bestset}} > SI_{g_{bestset}}$ **then**
  $g_{bestset} \leftarrow l_{bestset}$
end if

▷ **Tandem run Recruitment Strategy for Finding Features**

Set $n_{remain} = n_{max} - n_{tan}$
for $p = 1$ to N_{ant} **do**
  $F_p \leftarrow NULL$ (Empty Subset)
  for $q = 1$ to n_{random} **do**
    Randomly pick a feature f_q from the n available features.
    $F_p = F_p \cup f_q$
  end for
  for $q = n_{random} + 1$ to n_{remain} **do**
    Select a feature f_q that has the highest pheromone, the highest heuristic value, and hasn't yet been added to the subset F_p of features.
    $F_p = F_p \cup f_q$
  end for
  Set $uniquefeature \leftarrow$ features present in leader subset but not in partially constructed subset F_p.
  for $q = n_{remain} + 1$ to n_{max} **do**
    Select a feature f_q from $uniquefeature$ that has the highest pheromone, the highest heuristic value.
    $F_p = F_p \cup f_q$
  end for
end for
$t = t + 1$
end while
Return the best subset $g_{bestset}$

$m * m$ is used to construct a similarity graph for a dataset with m instances, where each edge connecting instances x_i to x_j represents similarity in the form of a weight w_{ij}. The Laplacian matrix L is computed using Eq. (3.1).

$$L = D - W, \tag{3.1}$$

where D and W are the diagonal and weight matrix, respectively.

Let's f_r is the r^{th} feature in all m instances then $f_r = (f_{r1}, f_{r2}, f_{r3}, f_{r4}, f_{r5}, ...f_{rm})^T$ where $r \in [1, n]$. Laplacian score of f_r is calculated as given in Eq. (3.2).

$$L_r = \frac{\tilde{f_r}^T L \tilde{f_r}}{\tilde{f_r}^T D \tilde{f_r}}, \tag{3.2}$$

where $\tilde{f_r}$ denotes the f_r vector's deviation from the mean and calculated as defined in Eq. (3.3).

$$\tilde{f_r} = f_r - (\frac{f_r^T D1}{1^T D1}), \tag{3.3}$$

where D is the diagonal matrix and $1 = [1, ..., 1]^T$. f_r^T is the transpose of f_r.

After getting the Laplacian score, heuristic value (h_r) is computed as given in Eq. (3.4).

$$h_r = 1/L_r. \tag{3.4}$$

The proposed NCHFS-ACO approach is then experimented with various benchmark datasets. The Experimental findings of these datasets are discussed in detail in subsequent section.

3.3 Experimental Evaluation

To perform experiments, we collected ten benchmark datasets from the UCI machine learning repository [48]. The details of these datasets are presented subsequently.

3.3.1 Dataset Details

In the experimental study, we used ten benchmark datasets, i.e., Iris, Sonar, Vehicle silhouettes, Ionosphere, Pima, Wine, Wdbc, Parkinsons, Pendigits, and Waveform (noise). The preprocessing of these datasets involves removing missing values. Following that step, standard scaling is applied to scale the data (except for Iris) such that each feature is normalized to achieve a mean of 0 and unit variance. The reason for this scaling is that the results of feature selection and clustering algorithms are affected by the fact that these datasets (except Iris) have a range of values with distinct scales. For all datasets, class labels are removed and not taken into account during the feature selection and clustering process. After preprocessing, the details of datasets are presented in Table 3.1.

Table 3.1: Benchmark datasets from UCI machine learning repository

Dataset name	Count of instances	Count of features	Count of classes
Iris	150	4	3
Sonar	208	60	2
Vehicle silhouettes	813	18	3
Ionosphere	351	33	2
Pima	768	8	2
Wine	178	13	3
Wdbc	569	30	2
Parkinsons	195	22	2
Pendigits	7494	16	10
Waveform (noise)	5000	40	3

3.3.2 Evaluation Measures

The performance of any feature selection approach is evaluated by using a clustering or classification method to determine how the feature selection improves clustering or classification performance. Since, the proposed approach is a clustering based hybrid feature selection approach that deals with the labeled and unlabeled datasets and uses K-means clustering to evaluate the qualities of features in terms of SI in the

wrapper stage, the K-means clustering is applied to evaluate the performance of the proposed feature selection approach. In the proposed approach, K-means clustering performance is evaluated using the SI as an internal evaluation measure and JI as an external evaluation measure. The reason behind taking these measures are they are widely used evaluation measures in various unsupervised feature selection approaches. The details of these measures are briefly discussed in Section 2.5. We also used one visualizer to visualize the clustering performance. The details of the visualizer is as follows:

Silhouette Visualizer:

The Silhouette Visualizer displays the silhouette coefficient for each sample on a per-cluster basis to show which clusters are dense and which are not. Additionally, it displays the number of clusters achieving an average SI value.

3.3.3 Hyperparameter Settings for the Proposed Approach

In order to conduct experiments, different n_{max} values are taken for all datasets, and the parameter settings for various variables are shown in Table 3.2.

Table 3.2: Hyperparameter settings for NCHFS-ACO

Variable Name	Value
max_{iter}	50
n_{random}	40% of n_{max}
$n_{arbitary}$	30% of n_{max}
n_{tan}	30% of n_{max}

3.3.4 Experimental Analysis

Experiments are conducted on the datasets listed in Table 3.1, and the NCHFS-ACO approach is compared to a hybrid feature selection approach developed by Solorio et al. [1], because both approaches used a similar strategy to obtain the best feature subset and quantified the results using the JI and SI. The NCHFS-ACO approach is performed by taking different-different n_{max} values and results are presented only for

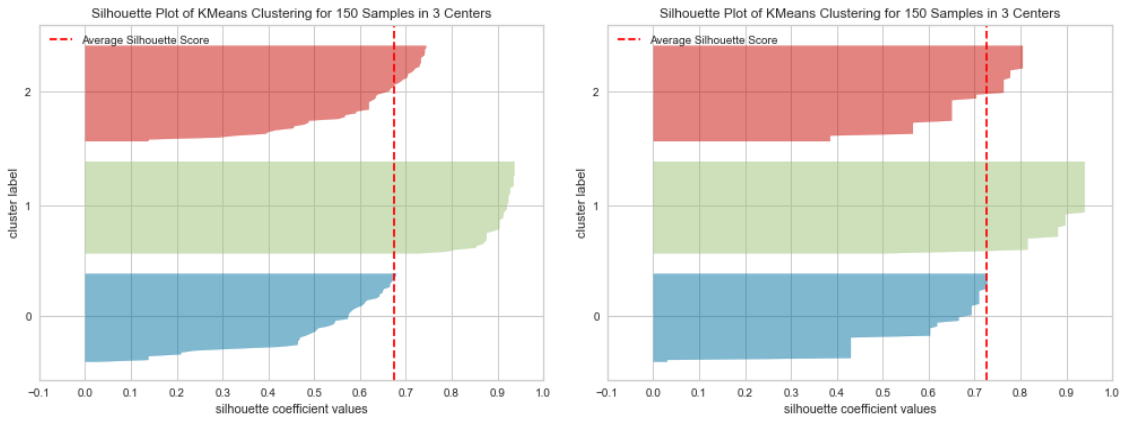
a few best performing n_{max} values. In the experiment, to speed up the computation, we took the number of clusters equal to the number of classes of a dataset.

Results on Iris dataset:

Table 3.3 shows the results of the NCHFS-ACO on the Iris dataset for various n_{max} values. When one feature is getting selected, the NCHFS-ACO gives the same JI but a higher SI than the Solorio et al. [1] approach. The Silhouette Visualizer obtained from NCHFS-ACO presented in Figure 3.2(b) shows that all clusters have a better average SI value than the Silhouette Visualizer shown in Figure 3.2(a) which is obtained from Solorio et al. [1].

Table 3.3: Results on Iris dataset

Technique used	No. of features selected	JI	SI
Solorio et al.	2	0.8575	0.6736
NCHFS-ACO	**1**	**0.8575**	**0.7259**
NCHFS-ACO	3	0.6418	0.5385
NCHFS-ACO	2	0.8575	0.6736



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 3.2: Results on Iris dataset

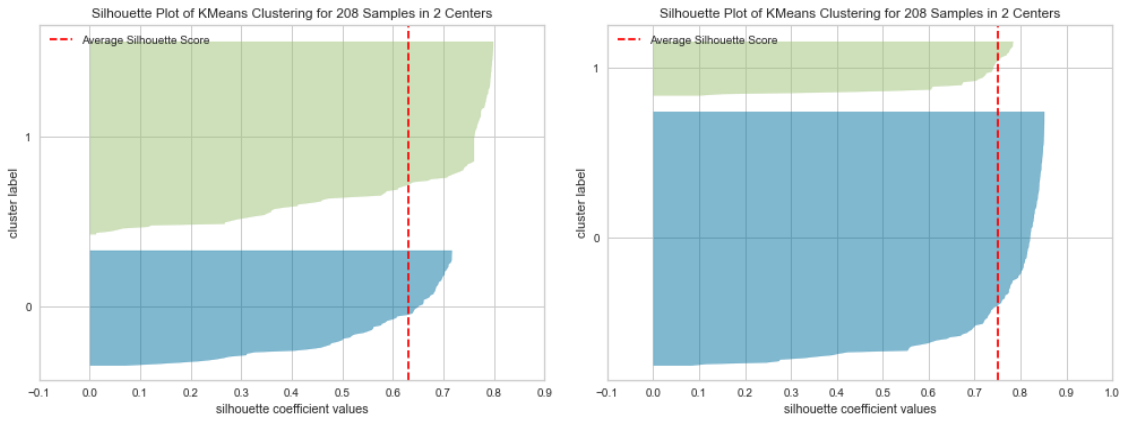
Results on Sonar dataset:

The NCHFS-ACO has been applied to Sonar dataset for different values of n_{max} and the combined results are shown in Table 3.4. The NCHFS-ACO is performing better

than Solorio et al. [1] approach when 1 and 3 number of features are getting selected but with 5 number of features it is performing better only in terms of JI. The NCHFS-ACO gives the highest SI values when 1 feature is getting selected. The Silhouette Visualizer obtained from NCHFS-ACO is presented in Figure 3.3(b) which shows that all clusters have a better average SI value than the Silhouette Visualizer shown in Figure 3.3(a) which is obtained from Solorio et al. [1].

Table 3.4: Results on Sonar dataset

Technique used	No. of features selected	JI	SI
Solorio et al.	1	0.3448	0.6304
NCHFS-ACO	**1**	**0.4273**	**0.7501**
NCHFS-ACO	3	0.4473	0.6319
NCHFS-ACO	5	0.4287	0.5121



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 3.3: Results on Sonar dataset

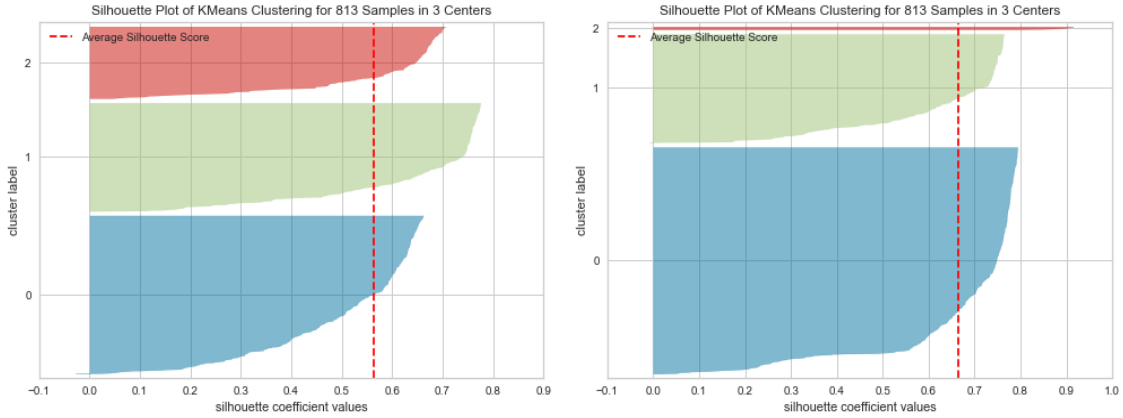
Results on Vehicle silhouettes dataset:

The NCHFS-ACO has been applied to the Vehicle silhouettes dataset for different values of n_{max} and combined results are presented in Table 3.5. The NCHFS-ACO approach gives increased JI and SI values in comparison to Solorio et al. [1] approach in all cases but it gives the highest SI value on 4 features. Also, the performance of NCHFS-ACO and Solorio et al. [1] is investigated on 5 number of features. However,

it has been found that NCHFS-ACO selects more relevant features and thus it outperforms in comparison to Solorio et al. [1] approach. The Silhouette Visualizer obtained from Solorio et al. [1] and NCHFS-ACO approach are also presented in Figure 3.4(a) and Figure 3.4(b) to observe the clustering results.

Table 3.5: Results on Vehicle silhouettes dataset

Technique used	No. of features selected	JI	SI
Solorio et al.	5	0.2935	0.5635
NCHFS-ACO	5	0.3162	0.6603
NCHFS-ACO	**4**	**0.3150**	**0.6650**
NCHFS-ACO	3	0.3148	0.6562
NCHFS-ACO	1	0.3319	0.6516



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 3.4: Results on Vehicle silhouettes dataset

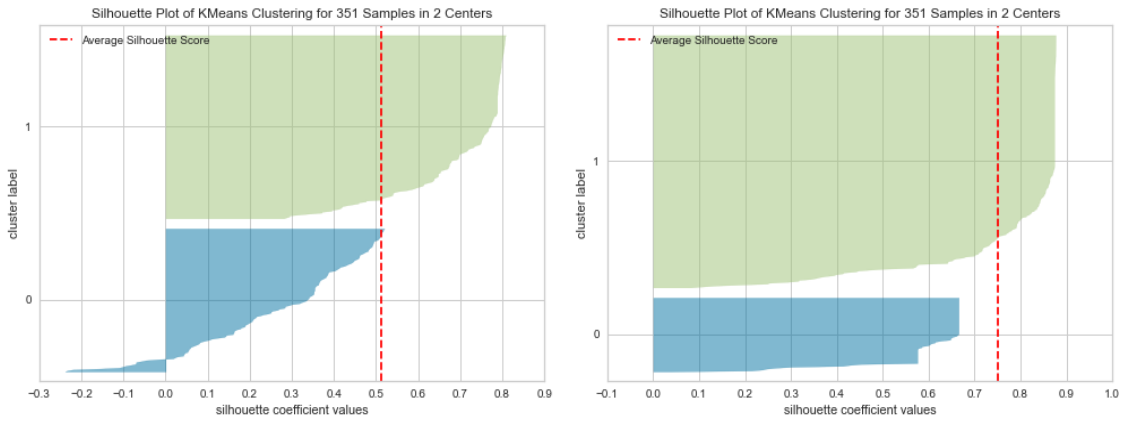
Results on Ionosphere dataset:

Table 3.6 shows the results of the NCHFS-ACO on the Ionosphere dataset for various n_{max} values. The NCHFS-ACO approach gives increased JI and SI values in comparison to Solorio et al. [1] approach in all cases but it gives the highest SI and JI values on 1 feature. Also, the performance of NCHFS-ACO and Solorio et al. [1] is investigated on 7 number of features. However, it has been found that NCHFS-ACO selects more relevant features and thus it outperforms in comparison to Solorio et al. [1] approach.

The Silhouette Visualizer presented in Figure 3.5(a) illustrates the results of Solorio et al. [1]. It shows that within the blue-colored cluster, some data points have negative silhouette coefficient values, which degrade the clustering result. On the other hand, The Silhouette Visualizer presented in Figure 3.5(b) illustrates the results of the NCHFS-ACO approach, showing that all clusters have positive silhouette coefficients, leading to an improved clustering result.

Table 3.6: Results on Ionosphere dataset

Technique used	No. of features selected	JI	SI
Solorio et al.	7	0.4376	0.5131
NCHFS-ACO	**1**	**0.6132**	**0.7506**
NCHFS-ACO	5	0.4486	0.5438
NCHFS-ACO	7	0.4589	0.5681



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 3.5: Results on Ionosphere dataset

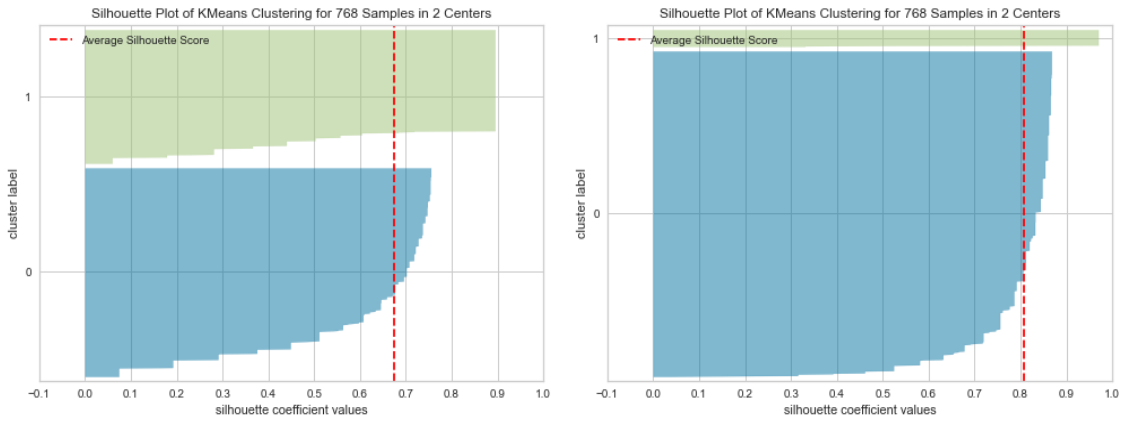
Results on Pima dataset:

Table 3.7 shows the results of the NCHFS-ACO on the Pima dataset for various n_{max} values. The NCHFS-ACO approach with one feature gives the far better JI and SI values than the Solorio et al. [1] approach while with 2 and 3 features it gives better result only in terms of JI but not in terms of SI. The Silhouette Visualizer obtained from Solorio et al. [1] and NCHFS-ACO approach are also presented in Figure 3.6(a)

and Figure 3.6(b) to observe the clustering results. The Silhouette Visualizer obtained from NCHFS-ACO shows that all clusters have a better average SI value than the Silhouette Visualizer obtained from Solorio et al. [1].

Table 3.7: Results on Pima dataset

Technique used	No. of features selected	JI	SI
Solorio et al.	1	0.3620	0.6736
NCHFS-ACO	**1**	**0.5200**	**0.8075**
NCHFS-ACO	2	0.5200	0.6570
NCHFS-ACO	3	0.4155	0.4329



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

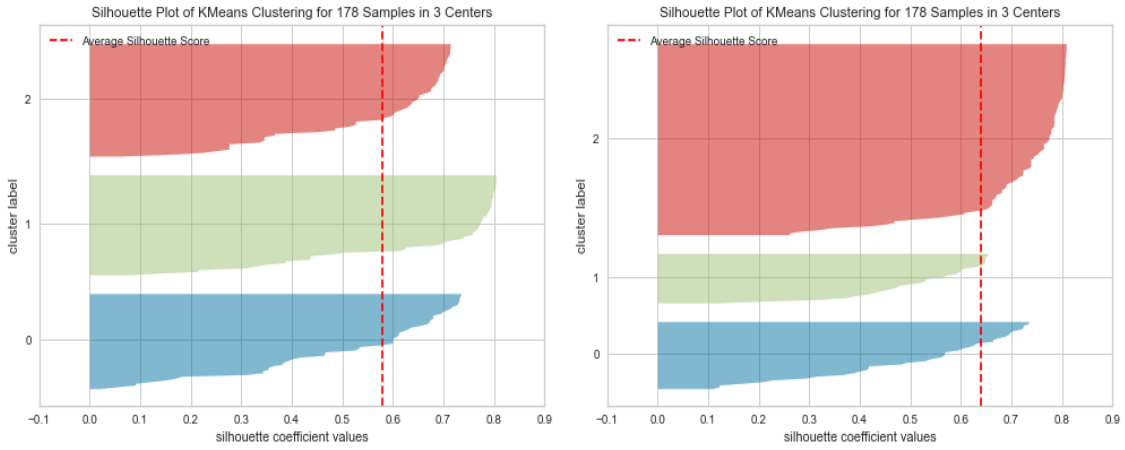
Figure 3.6: Results on Pima dataset

Results on Wine dataset:

Table 3.8 shows the results of the NCHFS-ACO on the Wine dataset for various n_{max} values. The NCHFS-ACO approach is giving higher SI and JI values when one feature is selected. While with 3 and 5 features it is performing better in terms of JI but not in terms of SI. The Silhouette Visualizer obtained from Solorio et al. [1] and NCHFS-ACO approach feature is also presented in Figure 3.7(a) and Figure 3.7(b) to observe the clustering results. It can be seen from the Silhouette Visualizer obtained from NCHFS-ACO that all clusters are having better average SI value than the Silhouette Visualizer obtained from Solorio et al. [1] approach.

Table 3.8: Results on Wine dataset

Technique used	No. of features selected	JI	SI
Solorio et al.	1	0.4953	0.5795
NCHFS-ACO	**1**	**0.5180**	**0.6386**
NCHFS-ACO	3	0.7079	0.4940
NCHFS-ACO	5	0.8174	0.4553



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 3.7: Results on Wine dataset

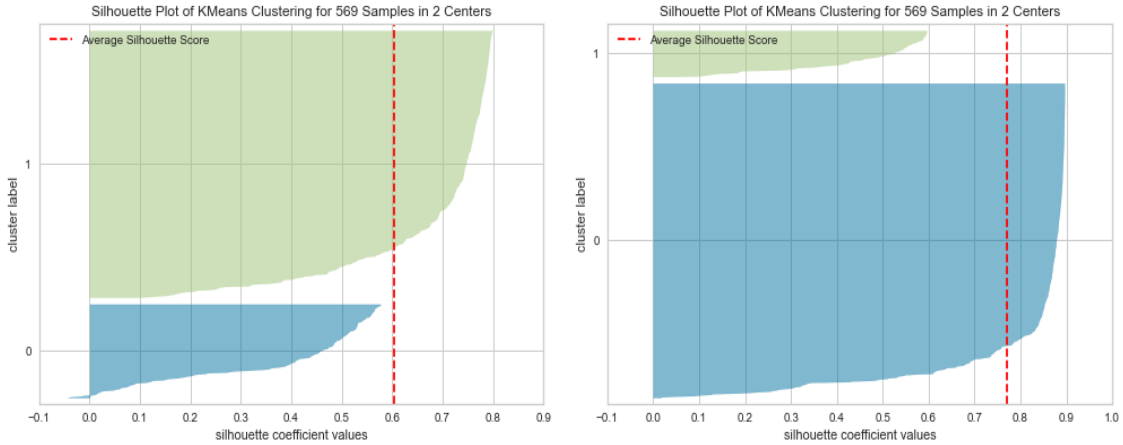
Results on Wdbc dataset:

Table 3.9 shows the results of the NCHFS-ACO on the Wdbc dataset for various n_{max} values. The NCHFS-ACO approach gives increased JI and SI values in comparison to Solorio et al. [1] approach in all cases but it gives the highest SI value when 1 feature is getting selected. Also, the performance of NCHFS-ACO and Solorio et al. [1] is investigated on 7 number of features. However, it has been found that NCHFS-ACO selects more relevant features and thus it outperforms in comparison to Solorio et al. [1] approach. The Silhouette Visualizer presented in Figure 3.8(a) displays the results of Solorio et al. [1], indicating that within the blue-colored cluster, certain data points exhibit negative silhouette coefficient values, thereby degrading the clustering result. On the other hand, the Silhouette Visualizer presented in Figure 3.8(b) displays the results of NCHFS-ACO, indicating that all data points have positive silhouette

coefficients. As a result, it achieves a better clustering outcome.

Table 3.9: Results on Wdbc dataset

Technique used	No. of features selected	JI	SI
Solorio et al.	7	0.5964	0.6040
NCHFS-ACO	**1**	**0.6251**	**0.7698**
NCHFS-ACO	3	0.6211	0.6857
NCHFS-ACO	5	0.6475	0.6672
NCHFS-ACO	7	0.6598	0.6539



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

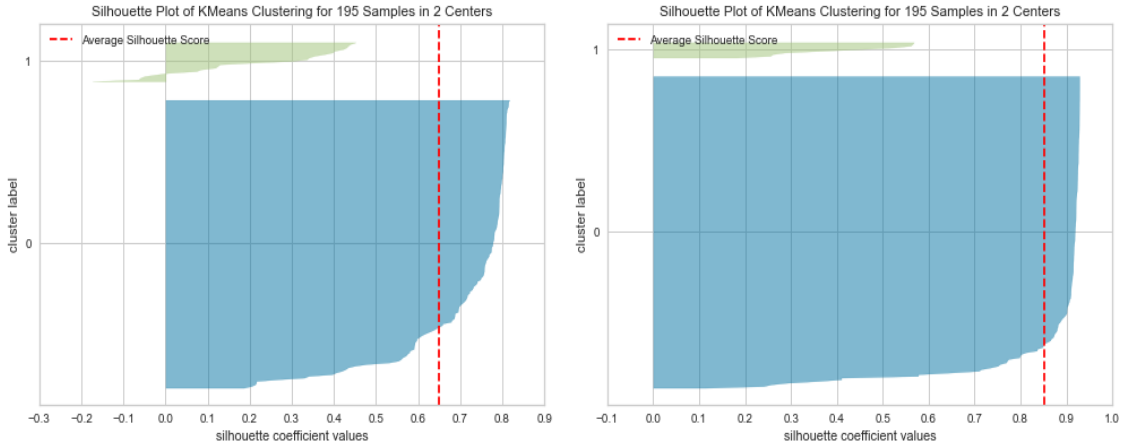
Figure 3.8: Results on Wdbc dataset

Results on Parkinsons dataset:

Table 3.10 shows the results of the NCHFS-ACO on the Parkinsons dataset for various n_{max} values. The NCHFS-ACO approach gives increased JI and SI values in comparison to Solorio et al. [1] approach in all cases but the highest SI value is attained when only 1 feature is selected. Also, the performance of NCHFS-ACO and Solorio et al. [1] is investigated on 12 number of features. However, it has been found that NCHFS-ACO selects more relevant features and thus it outperforms in comparison to Solorio et al. [1] approach. The Silhouette Visualizer obtained from Solorio et al. [1] and NCHFS-ACO approach is presented in Figure 3.9(a) and Figure 3.9(b) to observe the clustering results. It can be seen from the Silhouette Visualizer shown in Figure

Table 3.10: Results on Parkinsons dataset

Technique used	No. of features selected	JI	SI
Solorio et al.	12	0.5014	0.6495
NCHFS-ACO	**1**	**0.5758**	**0.8507**
NCHFS-ACO	5	0.5531	0.7952
NCHFS-ACO	12	0.5479	0.6860



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 3.9: Results on Parkinsons dataset

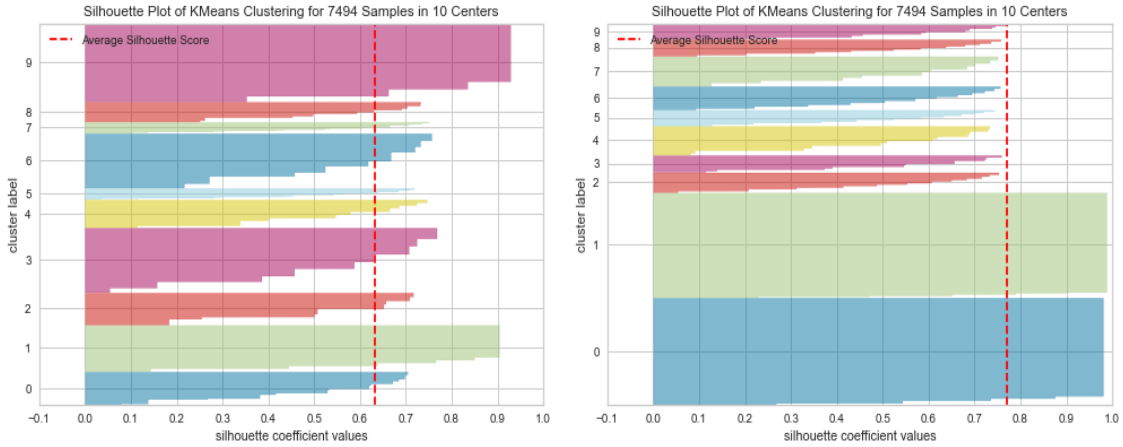
3.9(a) that in the green-colored cluster some data points are having negative silhouette coefficients value, which is degrading the clustering result. On the other side in Figure 3.9(b) all data points are having positive silhouette coefficient values due to this it achieves the better clustering result.

Results on Pendigits dataset:

Table 3.11 shows the results of the NCHFS-ACO on the Pendigits dataset for various n_{max} values. When one feature is selected, the NCHFS-ACO gives better JI and SI than the Solorio et al. [1] approach. The Silhouette Visualizer obtained from Solorio et al. [1] and NCHFS-ACO approach are also presented in Figure 3.10(a) and Figure 3.10(b) to observe the clustering results. It can be seen from the Silhouette Visualizer obtained from NCHFS-ACO that all clusters are having better average SI value than the Silhouette Visualizer obtained from Solorio et al. [1] approach.

Table 3.11: Results on Pendigits Dataset

Technique used	No. of features selected	JI	SI
Solorio et al.	1	0.1635	0.6319
NCHFS-ACO	**1**	**0.1728**	**0.7707**
NCHFS-ACO	8	0.4128	0.3892



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

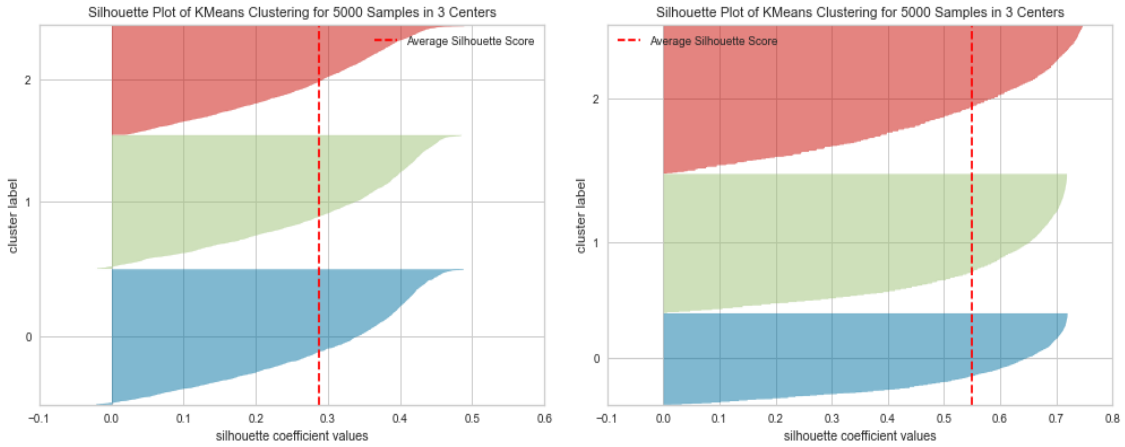
Figure 3.10: Results on Pendigits dataset

Results on Waveform dataset:

Table 3.12 shows the results of the NCHFS-ACO on the Waveform dataset for various n_{max} values. When one feature is selected, the NCHFS-ACO gives the better JI and SI than the Solorio et al. [1] approach. Also, the performance of NCHFS-ACO and Solorio et al. [1] is investigated on 15 number of features. However, it has been found that NCHFS-ACO selects more relevant features and thus it outperforms in comparison to Solorio et al. [1] approach. The Silhouette Visualizer obtained from Solorio et al. [1] and NCHFS-ACO approach is presented in Figure 3.11(a) and Figure 3.11(b) to observe the clustering results. It can be observed from the Silhouette Visualizer obtained from NCHFS-ACO that all clusters are having better average SI value than the Silhouette Visualizer obtained from Solorio et al. [1] approach.

Table 3.12: Results on Waveform dataset

Technique used	No. of features selected	JI	SI
Solorio et al.	15	0.3354	0.2881
NCHFS-ACO	**1**	**0.3554**	**0.5491**
NCHFS-ACO	15	0.3359	0.2890



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 3.11: Results on Waveform dataset

3.3.5 Comparative Performance Analysis

The comparison between the proposed NCHFS-ACO and Solorio et.al. [1] in terms of the SI, JI, and the number of features reduced is presented in Figure 3.12, Figure 3.13 and Table 3.13 respectively. From Figure 3.12, it can be seen that the NCHFS-ACO provides significantly higher SI values across all datasets, whereas Figure 3.13 demonstrates that NCHFS-ACO provides significantly higher JI values across all datasets, except the Iris dataset. In the case of the Iris dataset, it provides the same JI as Solorio et al. [1] approach. From the Table 3.13 it can be observed that in case of Iris, Vehicle silhouettes, Ionosphere, Wdbc, Parkinsons and Waveform (noise) the proposed NCHFS-ACO significantly reduced the features.

We performed another comparison in terms of SI, by taking the same number of features as computed by the Solorio et al. [1] approach shown in Figure 3.14. The figure indicates that the NCHFS-ACO provides superior SI values for all datasets

except Iris.

Table 3.13: Number of features reduced by the proposed approach in comparison with Solorio et al. [1] for the benchmark datasets

Dataset name	No. of features reduced by NCHFS-ACO	No. of features reduced by Solorio et al.
Iris	3	2
Sonar	59	59
Vehicle silhouettes	14	13
Ionosphere	32	26
Pima	7	7
Wine	12	12
Wdbc	29	23
Parkinsons	21	10
Pendigits	15	15
Waveform (noise)	39	25

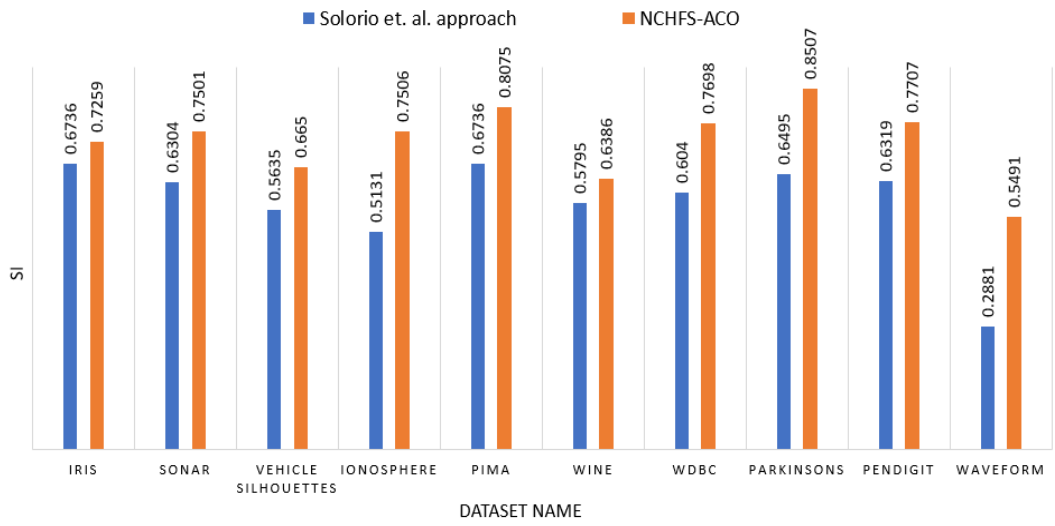


Figure 3.12: Comparison between Solorio et al. [1] and NCHFS-ACO in SI

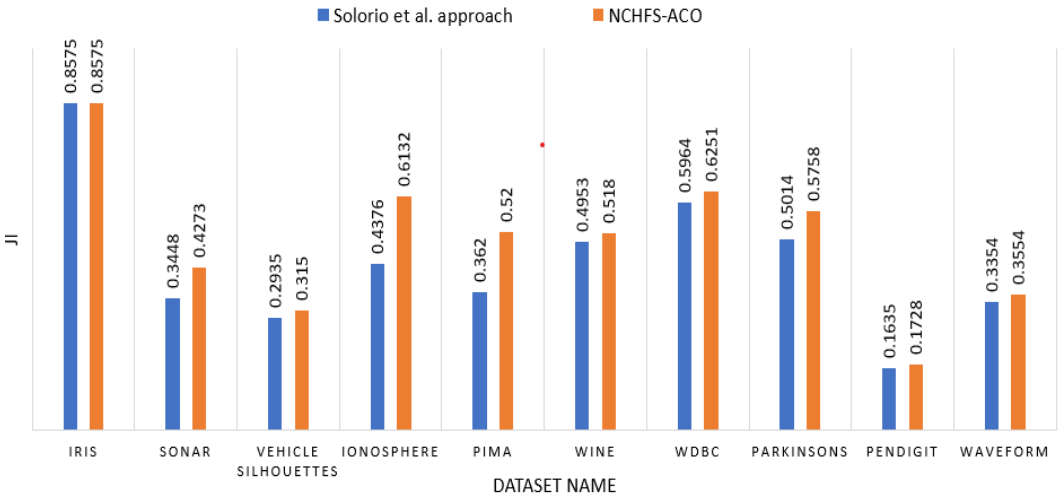


Figure 3.13: Comparison between Solorio et al. [1] and NCHFS-ACO in JI

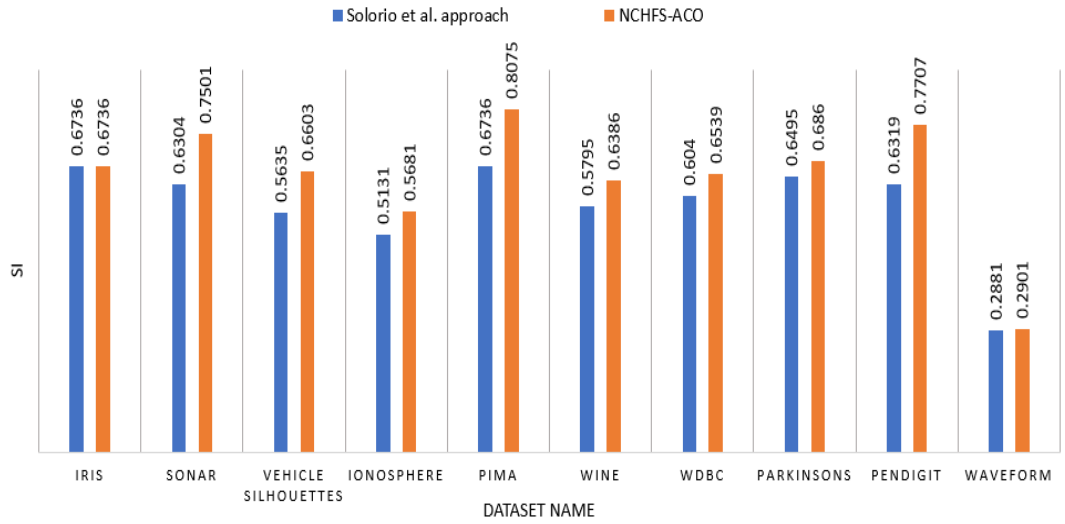


Figure 3.14: Comparison between Solorio et al. [1] and NCHFS-ACO by taking the same no. of features computed by Solorio et al. [1]

3.4 Summary

In this chapter, we proposed a novel clustering-based hybrid feature selection approach using Ant Colony Optimization abbreviated as "NCHFS-ACO". The proposed approach removes unnecessary or unimportant features which have a negative impact on model construction and thus selects the most appropriate features from large datasets. Rather than sequentially selecting features, the NCHFS-ACO selects features in random order to avoid being trapped in the local optimum, which is verified

when tested on the ten benchmark datasets. Another significant advantage of the proposed NCHFS-ACO technique is that it uses a combination of the Laplacian score and SI to measure feature relevance, making it easier to select more relevant features by integrating the characteristics of both measures at the same time. Furthermore, it simulates the behavior of the Temnothorax Albipennis ant species, employing a tandem run strategy to select the most promising features from a leader subset. This ensures that the proposed NCHFS-ACO method preserves the most promising features throughout the computation without losing them. As a result, when tested on ten benchmark datasets, it is found to be superior compared to the existing method in terms of SI and JI. Further, it is observed that our proposed method works well with small, medium, and large-sized benchmark datasets.

The proposed NCHFS-ACO is further tested on unlabeled real-life plant genome and protein datasets, i.e., Single Nucleotide Polymorphisms (SNP) and Protein sequences of rice and soybean plant species. The detailed performance investigation of these real-life plant genome datasets is reported in Chapter 7.

However, to apply feature selection to these genome datasets, there is a need to develop the feature extraction methods to transform the genome data into feature vectors (numeric values). So, we proposed two scalable feature extraction methods based on Apache Spark, which are discussed in detail in Chapters 4 and 5.

Chapter 4

A Novel Apache Spark Based 14-Dimensional Scalable Feature Extraction Approach for the Clustering of Genomics Data

The feature selection method proposed in Chapter 3 performs well for datasets with numerical features. However, to apply feature selection to real-life genome datasets, in this chapter, we presented a novel alignment-free, Apache Spark-based scalable feature extraction approach to transform the genome data into feature vectors consisting of numerical values. This novel approach overcomes the time consuming nature of alignment-based approaches and extracts significantly important features from millions of genome sequences in less computational time. It extracts features in five stages, i.e., based on the length of the sequence, the frequency of nucleotide bases, the pattern organization of nucleotide bases, the distribution of nucleotide bases, and the entropy of the sequence, to generate a fixed-length numeric vector consisting of only 14 dimensions to describe each genome sequence uniquely. This approach efficiently extracts context based features in terms of pattern organization and distribution and also removes the drawback of extracting the same features for dissimilar sequences using a novel power method. The feature extracted with the proposed scalable feature extraction approach is applied to K-means and Fuzzy c-means clustering techniques.

The experimental results show that the proposed method is highly successful and efficient in terms of computing time in comparison to other state-of-the-art approaches.

4.1 Introduction

Feature extraction plays a crucial role in bioinformatics by converting genome sequences into feature vectors (numeric values), thereby facilitating the clustering [155] [156] of similar genome sequences. By facilitating clustering, it aids in the classification of genes, regulatory elements, and functional regions by categorising similar sequences. Traditionally, biologists relied on alignment-based methods to identify similarities and homologies across sequences, enabling the classification of new biological sequences into established families or classes [116]. However, this approach is time-consuming, especially considering the vast amount of data present in genomics. One of the significant challenges in applying such a method, for example, in metagenomics, is that between 25% and 65% of the sequences have no homolog (orphan sequences) in the databases, rendering these sequences useless [157]. In response to such challenges, adopting alignment-free feature extraction techniques [50, 158] emerges as a viable solution. These methods offer efficient and scalable solutions for analyzing large-scale genomic datasets without the need for computationally intensive pairwise sequence alignments, reducing bias and enabling broad applicability to diverse genomic analysis tasks.

In feature extraction, each genome sequence can be determined by several features, yielding a vector of numerical values obtained by a description function that binds sequences and features. However, designing of an appropriate feature extraction method is crucial for a trustworthy knowledge process, as this stage directly impacts the final result. Nowadays, numerous feature extraction approaches are available, yet the majority lack scalability [86, 88, 90], hindering their efficiency in handling vast amounts of genomic data. Moreover, some approaches extract identical features from dissimilar sequences [78, 88], leading to inappropriate clustering. Consequently, developing an appropriate and scalable strategy for feature extraction remains a challenging

task, crucial for efficiently processing thousands of sequences and addressing clustering difficulties.

Based on the above discussion, we can say that feature extraction process directly impacts the accuracy of clustering. In addition, the scalability saves a huge amount of time in the processing of large scale genome sequences. So, in this chapter, we proposed a novel Apache Spark based 14-dimensional scalable feature extraction technique (14d-SFET) which extracts features based on length, frequency, modified total distance, distribution, and entropy to describe the genome sequence uniquely. The detailed methodology of proposed approach is presented subsequently.

4.2 Novel 14-Dimensional Scalable Feature Extraction Approach

The proposed 14d-SFET extracts five distinct types of sequence features in five stages: sequence length, frequency-based features, modified total distance-based features, distribution-based features, and sequence entropy. It extracts context-based features in the form of modified total distance and distribution. Moreover, the proposed 14d-SFET addresses the drawback of having the same total distance and same distribution for a nucleotide in two dissimilar sequences by applying a novel power mechanism. We used the Apache Spark framework to make the proposed approach scalable. The architecture of the proposed approach is shown in Figure 4.1. The five different types of features extracted in the proposed approach are discussed in details subsequently.

Table 4.1: Example of genome sequence with a novel power method

	4^0	4^1	4^2	4^3	4^4	4^5	4^6	4^7	4^8
	1	2	3	4	5	6	7	8	9
Seq 1	A	G	T	C	T	A	T	G	C

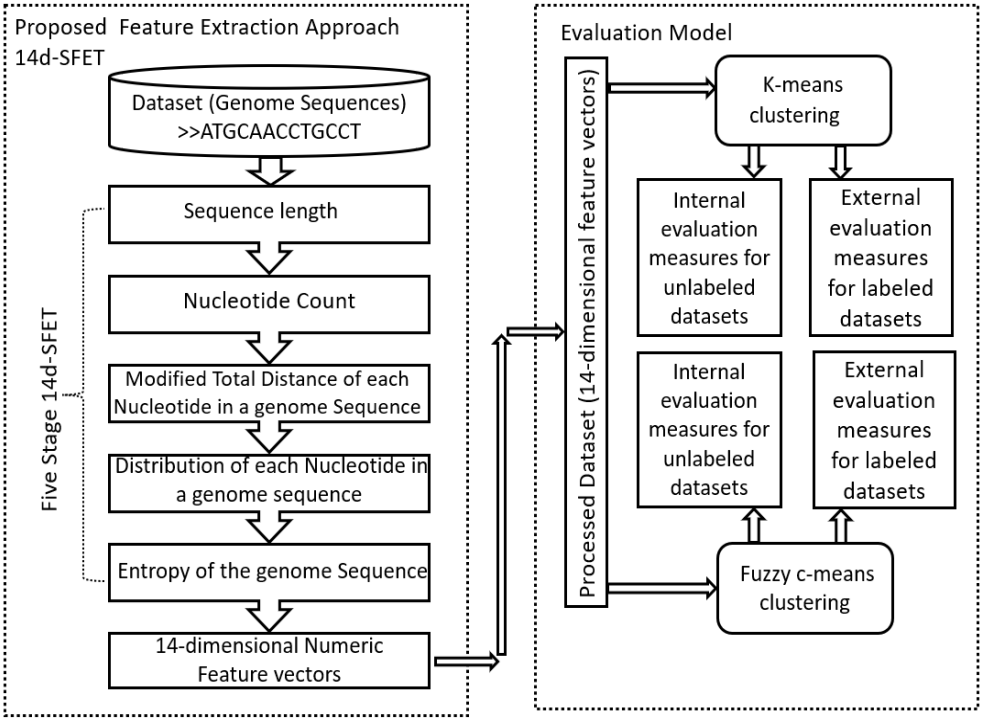


Figure 4.1: Proposed 14d-SFET architecture

(I) Sequence length:

Genome length varies from organism to organism, providing helpful information for the purpose of clustering. So, we used sequence length as a feature in the proposed method. In a genome sequence, sequence length (Seq_{len}) describes the total number of nucleotide present in that sequence. For example, a genome sequence is presented in Table 4.1. For this sequence, the value of Seq_{len} will be 9.

(II) Features based on the frequency:

In this type of feature, for each nucleotide n, the count of each nucleotide (len_n) is calculated. In the sequence presented in Table 4.1, the total count of A, T, G, and C are 2, 3, 2, and 2, so the value of len_A, len_T, len_G and len_C are 2, 3, 2, and 2, respectively.

(III) Features based on modified total distance:

In this type of feature, we removed the lacunae of having same total distances for a nucleotide n in non similar sequences by employing a novel power method, in which we multiplied the distances (d_i) by its place values, and named these type of features

as features based on the modified total distance. In order to comprehend the total distance problem of 12d-FET [50], if A appears at positions 3 and 4 in one sequence and positions 2 and 5 in another sequence, the total distance = 7 will be identical for both sequences. Owing to this, the feature based on total distance cannot correctly differentiate between two distinct genome sequences. So by using a novel power method, for each nucleotide n, the modified total distance from the first nucleotide (MTD_n) is calculated using Eq. (4.1).

$$MTD_n = \sum_{i=1}^{len_n} d_i * 4^{(d_i-1)}, \tag{4.1}$$

where d_i is the distance of the i^{th} nucleotide from the first nucleotide. In this technique, we selected four as a base number because, in any genome sequence, there are only four characters, i.e., A, T, G, and C. For example, in the sequence represented in Table 4.1, the nucleotide A appears at positions 1 and 6, so the modified total distance of A will be $MTD_A = 1 * 4^0 + 6 * 4^5 = 6145$. Using this approach, there is very little probability that we will get the same MTD_n for a nucleotide n in two nonsimilar sequences.

(IV) Features based on the distribution:

In this type of feature, we extracted the features based on distribution, by using the features based on frequency (len_n) and the features based on modified total distance (MTD_n). For a nucleotide n, the distribution (dis_n) is calculated by using using Eq. (4.2) and (4.3).

$$Avgd_n = MTD_n/len_n, \tag{4.2}$$

$$dis_n = \sum_{i=1}^{len_n} \frac{(d_i - Avgd_n)^2}{len_n}, \tag{4.3}$$

where $Avgd_n$ is the average of modified total distance of a nucleotide n.

(V) Entropy:

Entropy [159] is used to describe the degree of randomness in a given sequence. Let S

be a sequence of nucleotide bases such that $S = B_1, B_2, B_3....B_n$ and let $B_1, B_2, B_3...B_n$ occurs in S with probabilities $P(B_1), P(B_2), P(B_3)....P(B_n)$ then the entropy of the sequence (En_S) is calculated using Eq. (4.4).

$$En_S = -\sum_{i=1}^{n} P(B_i) log P(B_i). \tag{4.4}$$

So, by merging these five types of features, the feature vector consists of fourteen features, like $< seq_{len}$, len_A, MTD_A, dis_A, len_G, MTD_G, dis_G, len_T, MTD_T, dis_T, len_C, MTD_C, dis_C, $En_S >$. The feature vector corresponding to genome sequence shown in Table 4.1 will be < 9, 2, 6145, 9418767.25, 2, 131080, 4294836234, 3, 30000, 99900027.66, 2, 590080, 87044766128.5, 1.36 >. The implementation of the proposed 14d-SFET on Apache Spark cluster is discussed next.

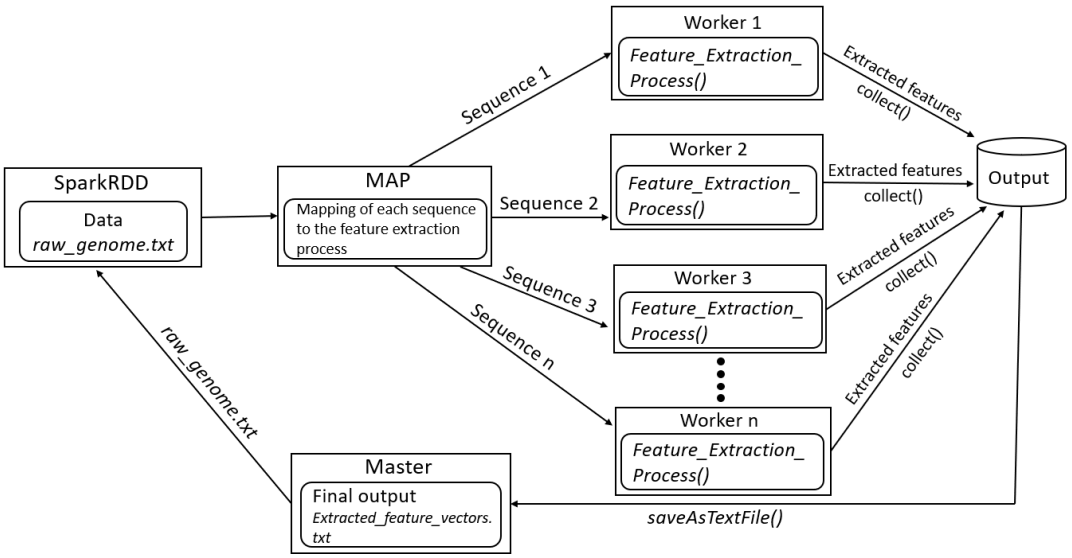


Figure 4.2: Implementation of proposed 14d-SFET on Apache Spark cluster

Scalable 14d-SFET on Apache Spark cluster:

To implement the proposed 14d-SFET on the Apache Spark cluster, first the dataset is made in Resilient Distributed Dataset (RDD) form so that it can be parallelized on various nodes to perform distributed computing, as shown in steps 1 and 2 of Algorithm 4.1. After that, in the third step, each sequence in RDD is mapped with the feature extraction process to extract features from each sequence. The feature extraction process contains the program for feature extraction. In the fourth step, the

extracted features are collected from all worker nodes. Finally, in the last step, all extracted features are saved in vector form to perform clustering of these sequences. The implementation of the proposed algorithm in Apache Spark cluster is shown in Figure 4.2. The pseudo code of proposed 14d-SFET is given in Algorithm 4.1.

Algorithm 4.1 14d-SFET

Input: $raw\_genome.txt$

Output: $Extracted\_features\_vectors.txt$

1: $x_1 = SparkContext.textFile(raw\_genome.txt)$

2: $x_1 = x_1.map(lambda\ y : numpy.array(y))$

3: $x_1 = x_1.map(lambda\ y : Feature\_Extraction\_Process(y))$

4: $x_1 = x_1.collect()$

5: $x_1.saveAsTextFile(Extracted\_features\_vectors.txt)$

Algorithm 4.2 $Feature\_Extraction\_Process()$

Input: Grid of characters; $x : [A, T, G, C]$

Output: seq_{len}, len_n, MTD_n, dis_n, En_s

1: Initialize the seq_{len} and len_n with 0, where $n \in \{A, T, G, C\}$.

2: Let n is a input character.

3: **for** n in x **do**

$seg_{len} = seq_{len} + 1$ (Increase the seq_{len} by 1)

$len_n = len_n + 1$ (Increase the count of nucleotide n by 1)

4: **end for**

5: Calculate the MTD_n for all nucleotides using Eq. (4.1).

6: Compute the dis_n for all nucleotides using Eq. (4.2) and (4.3).

7: Calculate the En_s using Eq. (4.4).

The feature extraction process is presented in Algorithm 4.2. In the first step of Algorithm 4.2, we initialized the seq_{len} and frequency of each nucleotide (len_n) with 0. In the second step, the input sequence characters are read one by one, and we assumed that the character under process is denoted by n. From the third to the fourth step, the sequence length (Seq_{len}) and the frequency-based features are calculated. The modified total distance (MTD_n) is calculated in the fifth step. Subsequently, the

sixth and seventh steps compute the distribution-based features (dis_n) and Entropy (En_S).

4.3 Experimental Evaluation

In this study, we collected nine genome datasets consisting of real-life plant genomes and benchmark datasets. To evaluate the performance of proposed 14d-SFET, we used two clustering techniques named K-means [160] and Fuzzy c-means [161]. The reason for using two clustering models is to judge the performance of the proposed method in general irrespective of any of the learning models used. To decide the number of clusters (k) in all datasets, the experiments are performed by taking various k values. The proposed approach performance is evaluated on a standalone Dell Workstation consist of Intel Xeon W-2102 processor having RAM of 64 GB and 4 Cores. The proposed approach implemented on Apache Spark and the scalability of the proposed approach is checked by varying the number of cores i.e. by using one core, two cores, three cores, and four cores. The detailed description of Apache Spark is presented in Section 2.4.

This section is divided into four subsections. The first subsection briefs about the dataset used for the experiments. The second subsection discusses the various evaluation measures used in experiments to evaluate the performance of the proposed method. The third subsection discusses the parameter settings for the K-means and Fuzzy c-means clustering techniques used in the study. Finally, the fourth subsection explains the experimental findings on various real-life plant genome and benchmark datasets.

4.3.1 Dataset Details

We used seven real-life plant genome unlabeled datasets of soybean crop, i.e., Williams82(Wm82).a1, Wm82.a2, Wm82.a4, Lee.a1, ZH13.a1, PI483463.a1, and W05.a1 taken from soybase repository[1] [49] and two benchmark labeled datasets, i.e.,

[1] https://www.soybase.org/

Splice and Promoter taken from UCI machine learning repository[2] [48], to perform the experiments. Details of these datasets are given in Section 2.6.1.

4.3.2 Evaluation Measures

To evaluate the performance of the proposed method on unlabeled datasets, we employed two internal evaluation measures, Silhouette Index (SI) and Davies-Bouldin Index (DBI). On the other hand, to evaluate the performance of the proposed approach on the labeled datasets, we used two external evaluation measures, Jaccard Index (JI) and Rand Index (RI). The details of these measures are as presented in Section 2.5.

4.3.3 Parameter Settings for Evaluation Models

Various parameters used in evaluation models to perform the experiments are listed here, and their values are presented in Table 4.2.

Table 4.2: Parameter Settings for evaluation models

K-means	*n_init*	10
	Maximum iteration	300
Fuzzy c-means	*error*	0.005
	Maximum iteration	1000

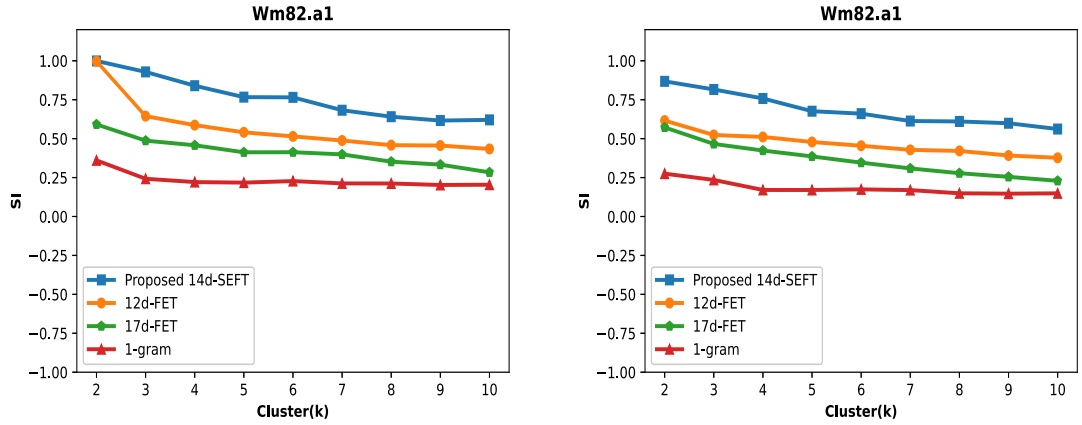
4.3.4 Experimental Analysis

We performed experiments on the datasets for different number of clusters (k) and presented the experimental results only for few best-performing k values in comparison with 1-gram [78], 12d-FET [50] and 17d-FET [90] in terms of SI and DBI for the unlabeled datasets and in terms of JI and RI for the labeled datasets. To evaluate the performance of distributed computing of proposed scalable approach, we conducted trials on a single core, two cores, three cores, and four cores and reported the time required to extract features presented in Table 4.12 for each configuration.

[2] https://archive.ics.uci.edu/

Results on Wm82.a1:

The experimental findings of the proposed 14d-SFET on the Wm82.a1 using Fuzzy c-means and K-means with $k = [2, 10]$ are displayed in Table 4.3. In both clustering methods, the proposed 14d-SFET gives better SI and DBI values than 12d-FET [50], 17d-FET [90], and 1-gram [78] for all values of k. In both the clustering methods, the proposed 14d-SFET yields the maximum SI at $k = 2$ and the lowest DBI at $k = 2$. The comparison on SI values between the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] using K-means and Fuzzy c-means is shown in Figure 4.3(a) and Figure 4.3(b), respectively. It can be observed that the proposed 14d-SFET produces the higher SI values in comparison to all approaches for all k values. The comparison on DBI between the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] using K-means and Fuzzy c-means is shown in Figure 4.4(a) and Figure 4.4(b), respectively. The proposed approach gives the lower DBI for all k in comparison to all approaches.



(a) Using K-means (b) using Fuzzy c-means

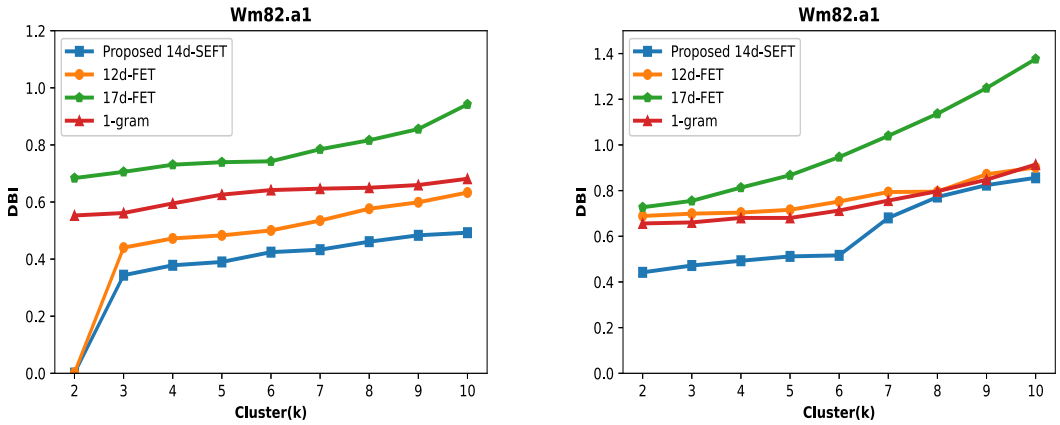
Figure 4.3: SI on Wm82.a1

Results on Wm82.a2:

The experimental results of the proposed 14d-SFET on the Wm82.a2 using Fuzzy c-means and K-means with $k = [2, 10]$ are presented in Table 4.4. In both clustering methods, the proposed 14d-SFET gives better SI and DBI values than 12d-FET [50],

Evaluation measures	Clusters	Algorithms							
		K-means				Fuzzy c-means			
		Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]	Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]
SI	2	**0.9990**	0.9969	0.5915	0.3606	**0.8680**	0.616	0.572	0.2751
	3	**0.9286**	0.6445	0.4871	0.2423	**0.8158**	0.5234	0.466	0.2348
	4	**0.8399**	0.5867	0.4575	0.2211	**0.7580**	0.5109	0.4236	0.1704
	5	**0.7665**	0.5406	0.4123	0.2175	**0.6762**	0.4784	0.3858	0.1702
	6	**0.7651**	0.5145	0.4129	0.2275	**0.6605**	0.4543	0.3458	0.1741
	7	**0.6825**	0.4879	0.3988	0.2125	**0.6131**	0.4278	0.3089	0.1696
	8	**0.6409**	0.4581	0.3521	0.2120	**0.6104**	0.4212	0.278	0.149
	9	**0.6162**	0.4558	0.3331	0.2024	**0.5989**	0.3915	0.2546	0.1466
	10	**0.6205**	0.4337	0.2836	0.2041	**0.5616**	0.377	0.229	0.1488
DBI	2	**0.0007**	0.0021	0.6838	0.5528	**0.4421**	0.6888	0.7271	0.6561
	3	**0.3436**	0.4402	0.7056	0.5614	**0.4718**	0.6990	0.7549	0.6606
	4	**0.3783**	0.4724	0.7306	0.5950	**0.4927**	0.7037	0.8131	0.6802
	5	**0.3900**	0.4831	0.7394	0.6258	**0.5118**	0.7158	0.8672	0.6803
	6	**0.4243**	0.5004	0.7426	0.6418	**0.5164**	0.7522	0.9468	0.7126
	7	**0.4329**	0.5349	0.7847	0.6467	**0.6804**	0.7937	1.0396	0.7562
	8	**0.4610**	0.5764	0.8163	0.6500	**0.7718**	0.7954	1.1365	0.7971
	9	**0.4833**	0.5990	0.8555	0.6595	**0.8239**	0.8714	1.2490	0.8477
	10	**0.4926**	0.6332	0.9421	0.6814	**0.8560**	0.9013	1.3763	0.9141

Table 4.3: Results on Wm82.a1



(a) Using K-means (b) using Fuzzy c-means

Figure 4.4: DBI on Wm82.a1

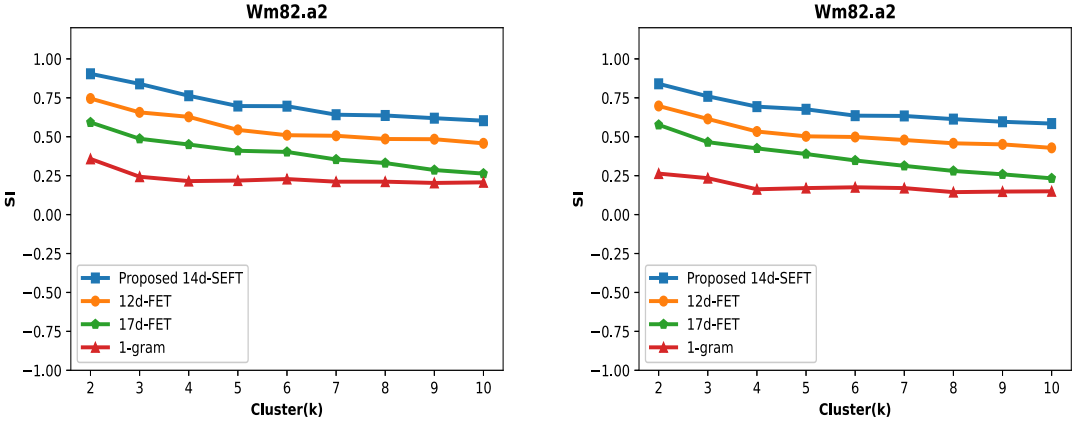
17d-FET [90], and 1-gram [78] for all values of k. The proposed 14d-SFET yields the highest SI and the lowest DBI at $k = 2$ among both clustering methods. Figure 4.5(a) and Figure 4.5(b) depict the comparison of SI values among the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] using K-means and Fuzzy c-means, respectively. Observably, the proposed 14d-SFET yields the highest SI values compared to all other approaches for all k values. The comparison of the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] on DBI using K-means and Fuzzy c-means is depicted in Figures 4.6(a) and (b), respectively. The proposed method yields the lowest DBI values compared to all other methods.

Results on Wm82.a4:

The experimental findings of the proposed 14d-SFET on the Wm82.a4 using Fuzzy c-means and K-means with $k = [2, 10]$ are shown in Table 4.5. The proposed 14d-SFET outperformed 12d-FET [50], 17d-FET [90], and 1-gram [78] for all k values. In both clustering methods, the proposed 14d-SFET produces the highest SI and lowest DBI at $k = 2$. Using K-means and Fuzzy c-means, Figure 4.7(a) and Figure 4.7(b) illustrate the comparison of SI values among the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78]. For all k values, it can be seen that the proposed 14d-SFET yields higher SI values than all other approaches. The comparison of the

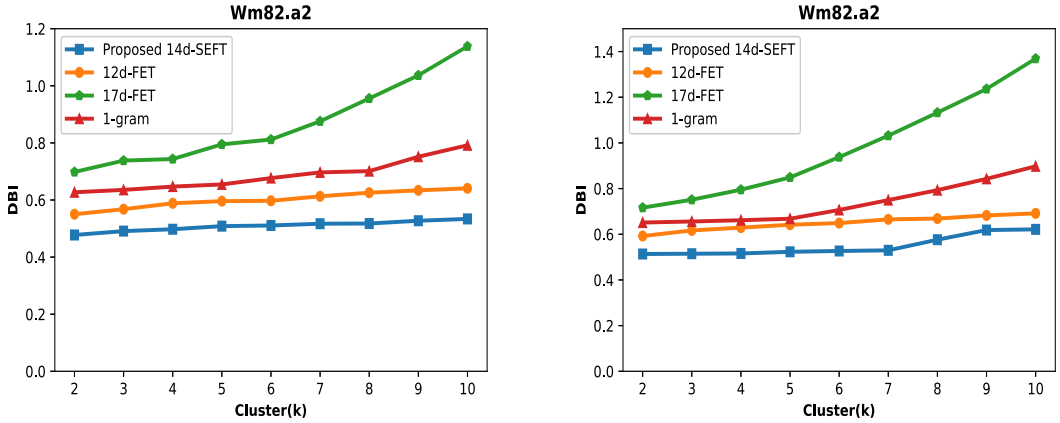
Evaluation measures	Clusters	Algorithms							
		K-means				Fuzzy c-means			
		Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]	Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]
SI	2	**0.9043**	0.7454	0.592067	0.3576	**0.8399**	0.6974	0.576356	0.2633
	3	**0.8395**	0.6565	0.4870	0.2431	**0.7595**	0.6143	0.4651	0.2339
	4	**0.7633**	0.6278	0.4499	0.2151	**0.6933**	0.5335	0.4251	0.1625
	5	**0.6971**	0.544	0.4098	0.2184	**0.6763**	0.5026	0.3890	0.1702
	6	**0.6962**	0.5098	0.4023	0.2282	**0.6353**	0.4983	0.3477	0.1754
	7	**0.6414**	0.5061	0.3538	0.2113	**0.6335**	0.479	0.3131	0.1702
	8	**0.6365**	0.4855	0.3310	0.2114	**0.6132**	0.4578	0.2803	0.1442
	9	**0.6191**	0.4841	0.2861	0.2032	**0.5961**	0.4509	0.2585	0.1478
	10	**0.603**	0.4576	0.2631	0.2072	**0.5846**	0.4286	0.2328	0.1495
DBI	2	**0.4774**	0.5501	0.6982	0.6270	**0.5135**	0.5921	0.7164	0.6513
	3	**0.4910**	0.5678	0.7380	0.6354	**0.5147**	0.6167	0.7512	0.6559
	4	**0.4975**	0.5884	0.7434	0.6470	**0.5159**	0.6290	0.7951	0.6614
	5	**0.5084**	0.5960	0.7949	0.6545	**0.5232**	0.6417	0.8488	0.6679
	6	**0.5106**	0.5972	0.8118	0.6767	**0.5267**	0.6489	0.9377	0.7063
	7	**0.5169**	0.6129	0.8757	0.6966	**0.5296**	0.6652	1.0317	0.7496
	8	**0.5176**	0.6255	0.9557	0.7010	**0.5762**	0.6686	1.1326	0.7931
	9	**0.5273**	0.6338	1.0365	0.7515	**0.6183**	0.6823	1.2362	0.8426
	10	**0.5337**	0.6408	1.1382	0.7916	**0.6214**	0.6914	1.3691	0.8975

Table 4.4: Results on Wm82.a2



(a) Using K-means (b) Using Fuzzy c-means

Figure 4.5: SI on Wm82.a2



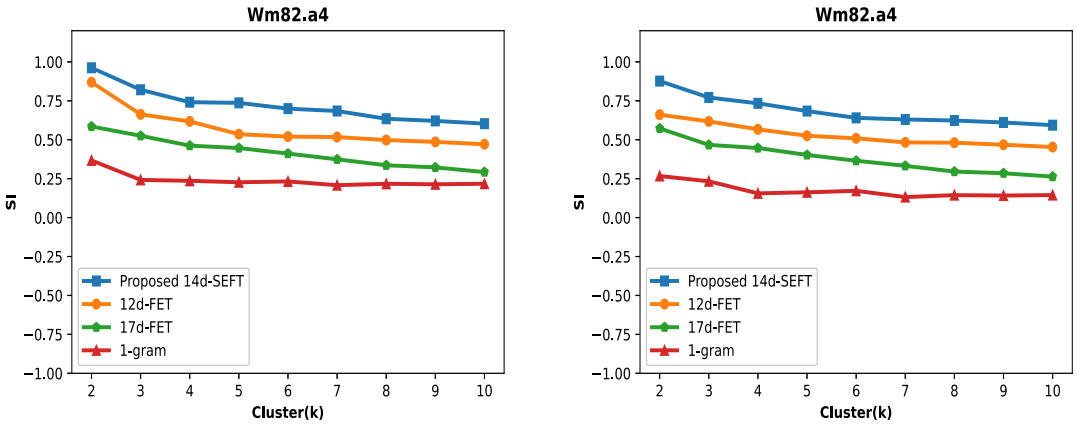
(a) Using K-means (b) Using Fuzzy c-means

Figure 4.6: DBI on Wm82.a2

proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] on DBI using K-means and Fuzzy c-means is illustrated in Figures 4.8(a) and (b), respectively. The proposed method yields the lowest DBI values compared to all other methods.

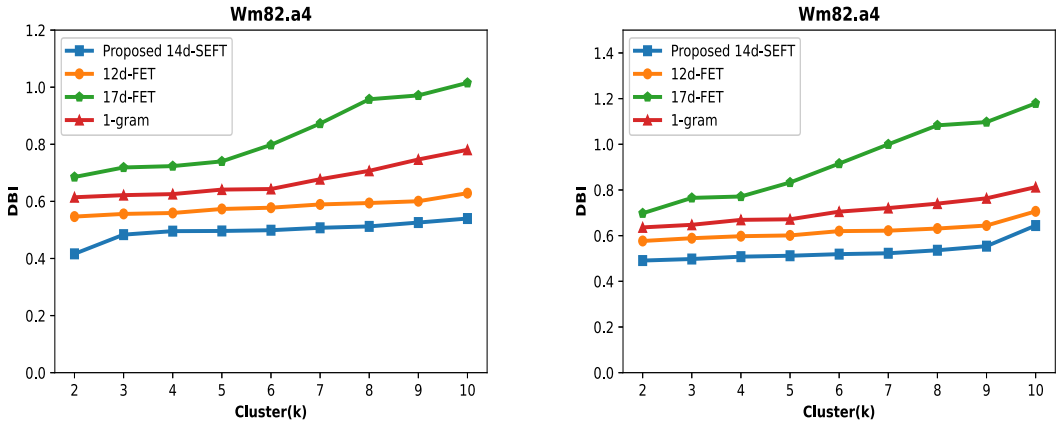
Table 4.5: Results on Wm82.a4

Evaluation measures	Clusters	Algorithms									
		K-means					Fuzzy c-means				
		Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]		Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]	
SI	2	**0.9621**	0.8696	0.5850	0.3673		**0.8766**	0.6609	0.5728	0.2670	
	3	**0.8211**	0.6630	0.5253	0.2419		**0.7712**	0.6176	0.4670	0.2331	
	4	**0.7410**	0.6177	0.4621	0.2361		**0.7337**	0.5668	0.4467	0.1556	
	5	**0.7368**	0.5356	0.4465	0.2268		**0.6841**	0.5254	0.4023	0.1623	
	6	**0.6996**	0.5200	0.4110	0.2321		**0.6404**	0.5091	0.3650	0.1718	
	7	**0.6845**	0.5171	0.3740	0.2080		**0.6303**	0.4829	0.3323	0.1318	
	8	**0.6347**	0.4979	0.3360	0.2171		**0.6233**	0.4809	0.2954	0.1441	
	9	**0.6209**	0.4860	0.3222	0.2138		**0.6108**	0.4677	0.2849	0.1416	
	10	**0.6032**	0.4711	0.2921	0.2174		**0.5935**	0.4532	0.2628	0.1447	
DBI	2	**0.4162**	0.5467	0.6854	0.6144		**0.4909**	0.5765	0.6979	0.6360	
	3	**0.4835**	0.5561	0.7187	0.6218		**0.4976**	0.5887	0.7652	0.6474	
	4	**0.4957**	0.5593	0.7236	0.6255		**0.5079**	0.5975	0.7714	0.6690	
	5	**0.4963**	0.5734	0.7401	0.6413		**0.5119**	0.6009	0.8327	0.6715	
	6	**0.4990**	0.5778	0.7981	0.6433		**0.5190**	0.6198	0.9154	0.7050	
	7	**0.5074**	0.5893	0.8726	0.6776		**0.5228**	0.6217	0.9998	0.7207	
	8	**0.5123**	0.5943	0.9577	0.7070		**0.5360**	0.6311	1.0829	0.7401	
	9	**0.5258**	0.6005	0.9712	0.7467		**0.5539**	0.6442	1.0972	0.7634	
	10	**0.5399**	0.6287	1.0152	0.7806		**0.6441**	0.7063	1.1793	0.8123	



(a) Using K-means (b) Using Fuzzy c-means

Figure 4.7: SI on Wm82.a4



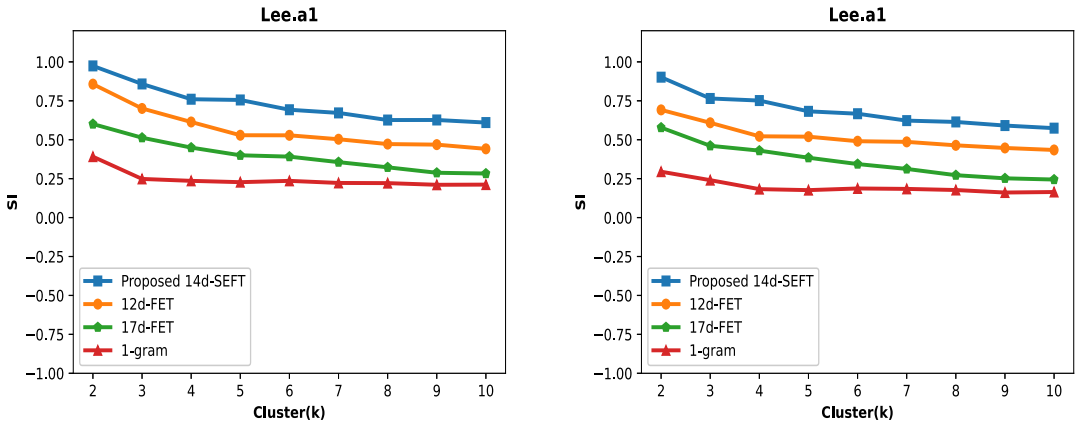
(a) Using K-means (b) Using Fuzzy c-means

Figure 4.8: DBI on Wm82.a4

Results on Lee.a1:

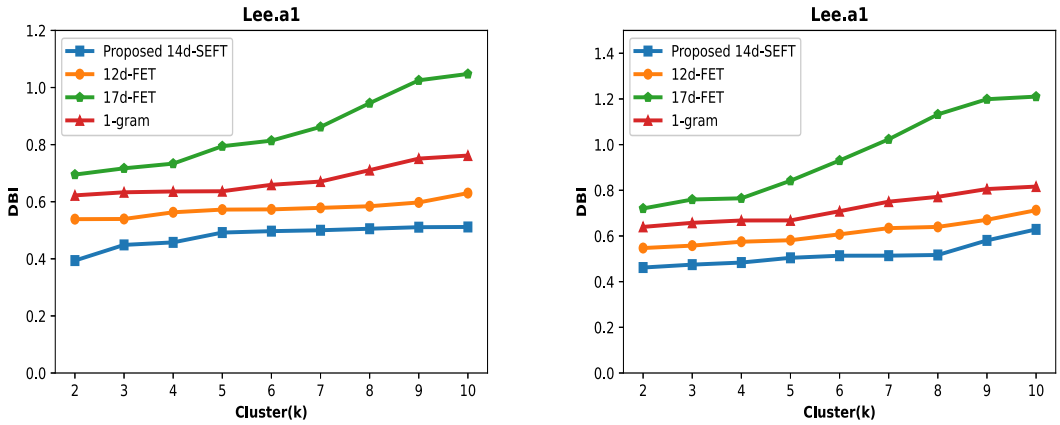
The experimental findings of the proposed 14d-SFET on the Lee.a1 using Fuzzy c-means and K-means with $k = [2, 10]$ are displayed in Table 4.6. The proposed 14d-SFET outperformed 12d-FET [50], 17d-FET [90], and 1-gram [78] for all k values. The proposed 14d-SFET gives the highest SI and the lowest DBI at $k = 2$ in both clustering approaches. Figures 4.9(a) and (b) show a comparison of SI values for the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] using K-means

and Fuzzy c-means, respectively. It can be seen that the proposed 14d-SFET delivers greater SI values than all other techniques for all k values. The DBI comparison of the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] using K-means and Fuzzy c-means is illustrated in Figures 4.10(a) and (b). In comparison to all other approaches, the proposed 14d-SFET has the lowest DBI for all k.



(a) Using K-means (b) Using Fuzzy c-means

Figure 4.9: SI on Lee.a1



(a) Using K-means (b) Using Fuzzy c-means

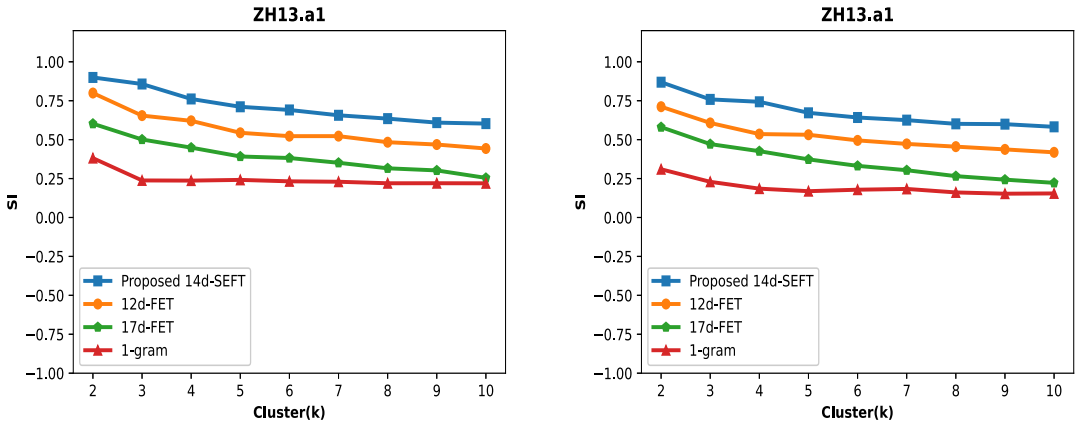
Figure 4.10: DBI on Lee.a1

Table 4.6: Results on Lee.a1

Evaluation measures	Clusters	Algorithms							
		K-means				Fuzzy c-means			
		Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]	Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]
SI	2	**0.9741**	0.8571	0.600789	0.3912	**0.9017**	0.6917	0.578082	0.2947
	3	**0.8582**	0.7008	0.5129	0.2482	**0.7649**	0.6088	0.4611	0.2404
	4	**0.7601**	0.6139	0.4495	0.2359	**0.7515**	0.5220	0.4299	0.1827
	5	**0.7553**	0.5288	0.4000	0.2274	**0.6823**	0.5196	0.3844	0.1763
	6	**0.6925**	0.5283	0.3914	0.2358	**0.6670**	0.4904	0.3437	0.1869
	7	**0.6722**	0.5031	0.3560	0.2224	**0.6227**	0.4862	0.3124	0.1842
	8	**0.6264**	0.4721	0.3227	0.2213	**0.6142**	0.4641	0.2722	0.1770
	9	**0.6268**	0.4685	0.2878	0.2106	**0.5910**	0.4471	0.2521	0.1609
	10	**0.6102**	0.4414	0.2828	0.2117	**0.5743**	0.4342	0.2443	0.1642
DBI	2	**0.3938**	0.5388	0.6950	0.6222	**0.4617**	0.5471	0.7203	0.6397
	3	**0.4487**	0.5395	0.7169	0.6329	**0.4744**	0.5577	0.7597	0.6576
	4	**0.4575**	0.5629	0.7334	0.6359	**0.4836**	0.5747	0.7649	0.6678
	5	**0.4919**	0.5724	0.7944	0.6366	**0.5043**	0.5813	0.8419	0.6681
	6	**0.4968**	0.5731	0.8138	0.6592	**0.5135**	0.6071	0.9305	0.7084
	7	**0.5000**	0.5785	0.8620	0.6706	**0.5137**	0.6342	1.0234	0.7502
	8	**0.5053**	0.5841	0.9453	0.7106	**0.5167**	0.6397	1.1324	0.7714
	9	**0.5107**	0.5974	1.0252	0.7510	**0.5806**	0.6708	1.1988	0.8054
	10	**0.5115**	0.6301	1.0475	0.7616	**0.6287**	0.7127	1.2102	0.8162

Results on ZH13.a1:

The experimental results of the proposed 14d-SFET on the ZH13.a1 using Fuzzy c-means and K-means with $k = [2, 10]$ are presented in Table 4.7. The proposed 14d-SFET outperformed 12d-FET [50], 17d-FET [90], and 1-gram [78] for all k values. The proposed 14d-SFET yields the highest SI at $k = 2$ and the lowest DBI at $k = 2$ among both clustering methods. Figure 4.11(a) and Figure 4.11(b) depict the comparison of SI values between the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] using K-means and Fuzzy c-means, respectively. Observably, the proposed 14d-SFET yields the highest SI values compared to all other approaches for all k values. The comparison of the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] on DBI using K-means and Fuzzy c-means is depicted in Figures 4.12(a) and (b), respectively. The proposed method yields the lowest DBI values compared to all other methods.



(a) Using K-means (b) Using Fuzzy c-means

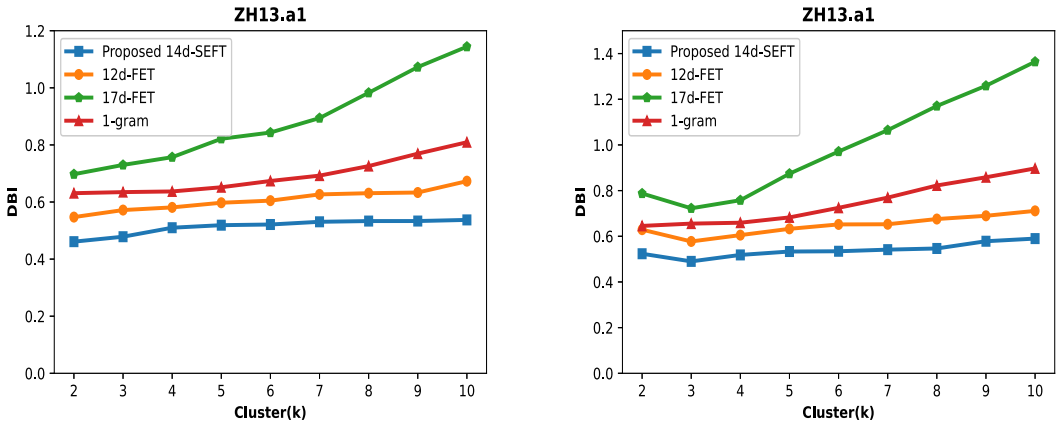
Figure 4.11: SI on ZH13.a1

Results on PI483463.a1:

The experimental outcomes of the proposed 14d-SFET on the PI483463.a1 using Fuzzy c-means and K-means with $k = [2, 10]$ are shown in Table 4.8. For all k values, the proposed 14d-SFET provides superior SI values than 12d-FET [50], 17d-FET [90], and 1-gram [78] for both clustering approaches. In both clustering methods, the

Evaluation measures	Clusters	Algorithms							
		K-means				Fuzzy c-means			
		Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]	Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]
SI	2	**0.8993**	0.7992	0.6022	0.3813	**0.8689**	0.7115	0.5802	0.3097
	3	**0.8572**	0.6543	0.5009	0.2379	**0.7584**	0.6072	0.4713	0.2291
	4	**0.7612**	0.6208	0.4486	0.2369	**0.7430**	0.5357	0.4258	0.1852
	5	**0.7109**	0.5439	0.3918	0.2414	**0.6722**	0.5313	0.3732	0.1689
	6	**0.6906**	0.5223	0.3819	0.2322	**0.6421**	0.4951	0.3316	0.1784
	7	**0.6563**	0.5225	0.3513	0.2295	**0.6252**	0.4725	0.3035	0.1834
	8	**0.6349**	0.4834	0.3158	0.2198	**0.6014**	0.4550	0.2655	0.1609
	9	**0.6092**	0.4685	0.3023	0.2200	**0.5994**	0.4373	0.2429	0.1528
	10	**0.6029**	0.4430	0.2543	0.2195	**0.5822**	0.4184	0.2224	0.1545
DBI	2	**0.4611**	0.5471	0.6974	0.6309	**0.5235**	0.6289	0.7873	0.6454
	3	**0.4784**	0.5717	0.7299	0.6346	**0.4900**	0.5773	0.7224	0.6554
	4	**0.5099**	0.5811	0.7570	0.6370	**0.5183**	0.6050	0.7579	0.6593
	5	**0.5188**	0.5974	0.8215	0.6516	**0.5332**	0.6326	0.8740	0.6826
	6	**0.5212**	0.6048	0.8433	0.6737	**0.5343**	0.6519	0.9711	0.7245
	7	**0.5307**	0.6266	0.8940	0.6924	**0.5414**	0.6526	1.06469	0.7692
	8	**0.5332**	0.6309	0.9827	0.7258	**0.5466**	0.6754	1.1704	0.8223
	9	**0.5333**	0.6332	1.0727	0.7693	**0.5782**	0.6896	1.2590	0.8583
	10	**0.5373**	0.6729	1.1440	0.8093	**0.5898**	0.7112	1.3643	0.8971

Table 4.7: Results on ZH13.a1



(a) Using K-means (b) Using Fuzzy c-means

Figure 4.12: DBI on ZH13.a1

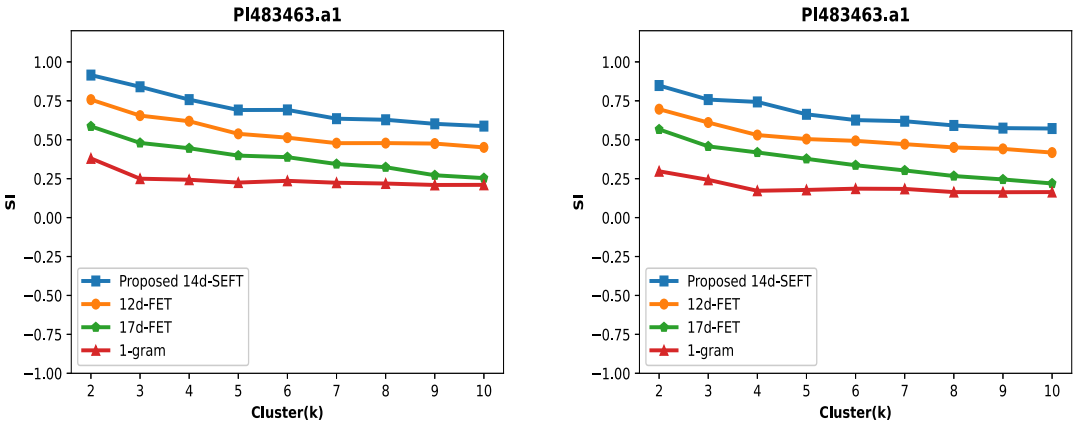
proposed 14d-SFET produces the highest SI and lowest DBI at $k = 2$. Using K-means and Fuzzy c-means, Figure 4.13(a) and Figure 4.13(b) illustrate the comparison of SI values among the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78]. For all k values, it can be seen that the proposed 14d-SFET yields higher SI values than all other approaches. The comparison of the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] on DBI using K-means and Fuzzy c-means is illustrated in Figures 4.14(a) and (b), respectively. The proposed method yields the lowest DBI values compared to all other methods.

Results on W05.a1:

The experimental results of the proposed 14d-SFET on the W05.a1 using Fuzzy c-means and K-means with $k = [2, 10]$ are displayed in Table 4.9. In both clustering methods, the proposed 14d-SFET gives better SI values than 12d-FET [50], 17d-FET [78], and 1-gram [78] for all values of k. The proposed 14d-SFET gives the highest SI at $k = 2$ and the lowest DBI at $k = 2$ in both clustering approaches. Figures 4.15(a) and (b) show a comparison of SI values for the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] using K-means and Fuzzy c-means, respectively. It can be seen that the proposed 14d-SFET delivers greater SI values than all other techniques for all k values. The DBI comparison of the proposed 14d-SFET, 12d-FET

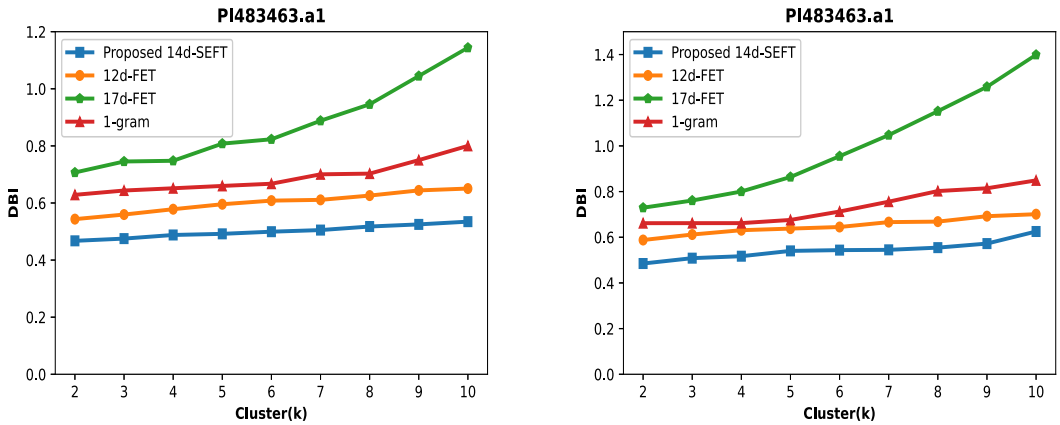
Evaluation measures	Clusters	Algorithms							
		K-means				Fuzzy c-means			
		Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]	Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]
SI	2	**0.9152**	0.7576	0.5860	0.3802	**0.8481**	0.6960	0.5663	0.2974
	3	**0.8401**	0.6549	0.4795	0.2498	**0.7578**	0.6105	0.457433	0.2425
	4	**0.7576**	0.6188	0.4453	0.2430	**0.7428**	0.5301	0.418382	0.1722
	5	**0.6908**	0.5379	0.3980	0.2248	**0.6633**	0.5042	0.377405	0.1776
	6	**0.6915**	0.5133	0.3880	0.2361	**0.6262**	0.4928	0.336311	0.1857
	7	**0.6352**	0.4780	0.3439	0.2237	**0.6189**	0.4716	0.30284	0.1842
	8	**0.6282**	0.4784	0.3235	0.2188	**0.5917**	0.4506	0.266954	0.1637
	9	**0.6021**	0.4754	0.2718	0.2095	**0.5746**	0.4413	0.244938	0.1629
	10	**0.5875**	0.4508	0.2535	0.2102	**0.5719**	0.4175	0.219177	0.1638
DBI	2	**0.4674**	0.5434	0.7068	0.6285	**0.4849**	0.5878	0.7293	0.6616
	3	**0.4752**	0.5594	0.7453	0.6436	**0.5084**	0.6118	0.7608	0.6617
	4	**0.4879**	0.5781	0.7478	0.6515	**0.5169**	0.6308	0.8003	0.6618
	5	**0.4919**	0.5956	0.8079	0.6597	**0.5402**	0.6378	0.8638	0.6757
	6	**0.4994**	0.6082	0.8231	0.6674	**0.5436**	0.6448	0.9549	0.7125
	7	**0.5051**	0.6109	0.8881	0.7004	**0.5448**	0.6662	1.0473	0.7557
	8	**0.5175**	0.6259	0.9457	0.7028	**0.5545**	0.6684	1.1519	0.8022
	9	**0.5250**	0.6440	1.0447	0.7502	**0.5722**	0.6921	1.2587	0.8144
	10	**0.5346**	0.6505	1.1439	0.8000	**0.6251**	0.7009	1.3989	0.8490

Table 4.8: Results on PI483463.a1



(a) Using K-means (b) Using Fuzzy c-means

Figure 4.13: SI on PI483463.a1



(a) Using K-means (b) Using Fuzzy c-means

Figure 4.14: DBI on PI483463.a1

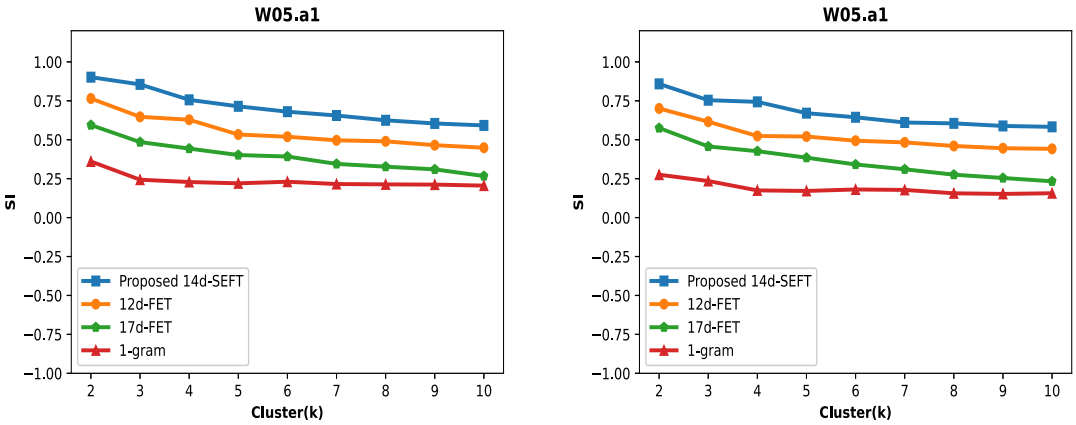
[50], 17d-FET [90], and 1-gram [78] using K-means and Fuzzy c-means is illustrated in Figures 4.16(a) and (b). In comparison to all other approaches, the proposed 14d-SFET has the lowest DBI for all k.

Results on Splice dataset:

The experimental outcomes of the proposed 14d-SFET on the Splice dataset using Fuzzy c-means and K-means with $k = [2, 10]$ are shown in Table 4.10. The proposed 14d-SFET outperformed 12d-FET [50], 17d-FET [90], and 1-gram [78] for all k values.

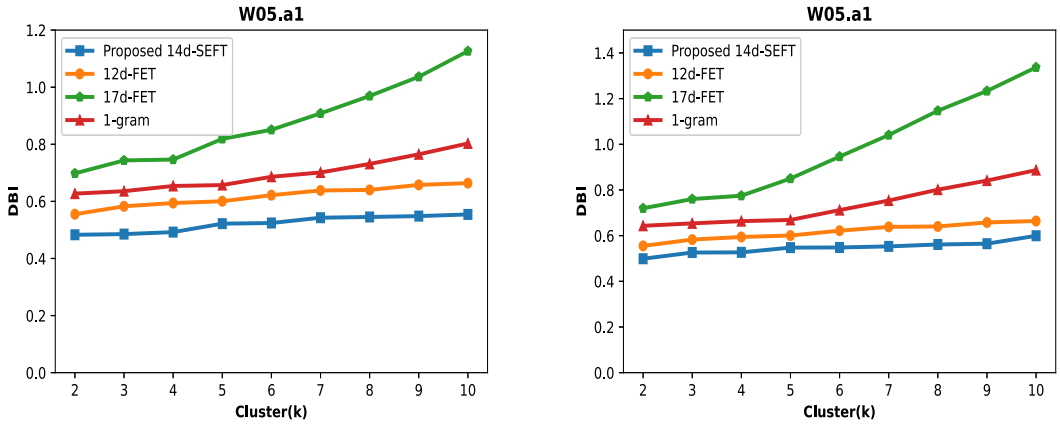
Evaluation measures	Clusters	Algorithms							
		K-means				Fuzzy c-means			
		Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]	Propoosed 14d-SFET	12d-FET [50]	17d-FET [90]	1-gram [78]
SI	2	**0.9018**	0.7648	0.5943	0.3605	**0.8588**	0.7007	0.5754	0.2754
	3	**0.8554**	0.6470	0.4850	0.2433	**0.7539**	0.6157	0.4568	0.2349
	4	**0.7563**	0.6284	0.4432	0.2284	**0.7433**	0.5241	0.4264	0.1746
	5	**0.7145**	0.5334	0.4017	0.2200	**0.6706**	0.5205	0.3847	0.1711
	6	**0.6801**	0.5193	0.3921	0.2302	**0.6443**	0.4936	0.3411	0.1805
	7	**0.6556**	0.4963	0.3448	0.2155	**0.6105**	0.4828	0.3103	0.1778
	8	**0.6249**	0.4898	0.3269	0.2132	**0.6051**	0.4600	0.2756	0.1559
	9	**0.6046**	0.4650	0.3097	0.2118	**0.5885**	0.4454	0.2543	0.1521
	10	**0.5919**	0.4486	0.2667	0.2053	**0.5828**	0.4415	0.2331	0.1563
DBI	2	**0.4829**	0.5550	0.6980	0.6271	**0.4988**	0.5550	0.7196	0.6430
	3	**0.4853**	0.5827	0.7435	0.6357	**0.5260**	0.5827	0.7602	0.6537
	4	**0.4923**	0.5942	0.7464	0.6540	**0.5268**	0.5942	0.7748	0.6634
	5	**0.5219**	0.6004	0.8189	0.6571	**0.5476**	0.6004	0.8498	0.6687
	6	**0.5242**	0.6217	0.8507	0.6861	**0.5482**	0.6217	0.9461	0.7117
	7	**0.5428**	0.6383	0.9082	0.7013	**0.5529**	0.6383	1.0405	0.7536
	8	**0.5454**	0.6401	0.9693	0.7312	**0.5613**	0.6401	1.1469	0.8012
	9	**0.5484**	0.6577	1.0365	0.7649	**0.5647**	0.6577	1.2333	0.8411
	10	**0.5543**	0.6639	1.1264	0.8029	**0.5994**	0.6639	1.3368	0.8868

Table 4.9: Results on W05.a1



(a) Using K-means (b) Using Fuzzy c-means

Figure 4.15: SI on W05.a1



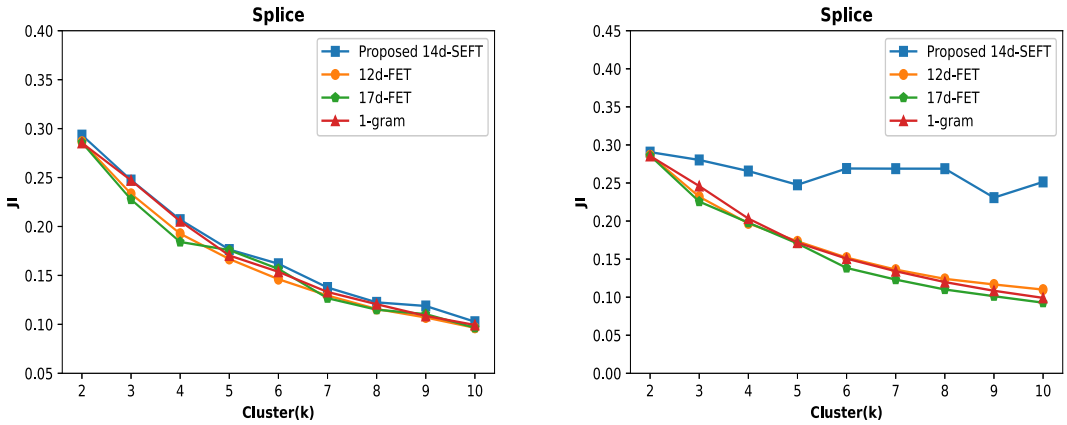
(a) Using K-means (b) Using Fuzzy c-means

Figure 4.16: DBI on W05.a1

Figure 4.17(a) and Figure 4.17(b) depict the comparison of JI values among the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] using K-means and Fuzzy c-means, respectively. Observably, the proposed 14d-SFET yields the highest JI values compared to all other approaches for all k values. The comparison of the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] on RI using K-means and Fuzzy c-means is depicted in Figures 4.18(a) and (b), respectively. The proposed method yields the highest RI values compared to all other methods.

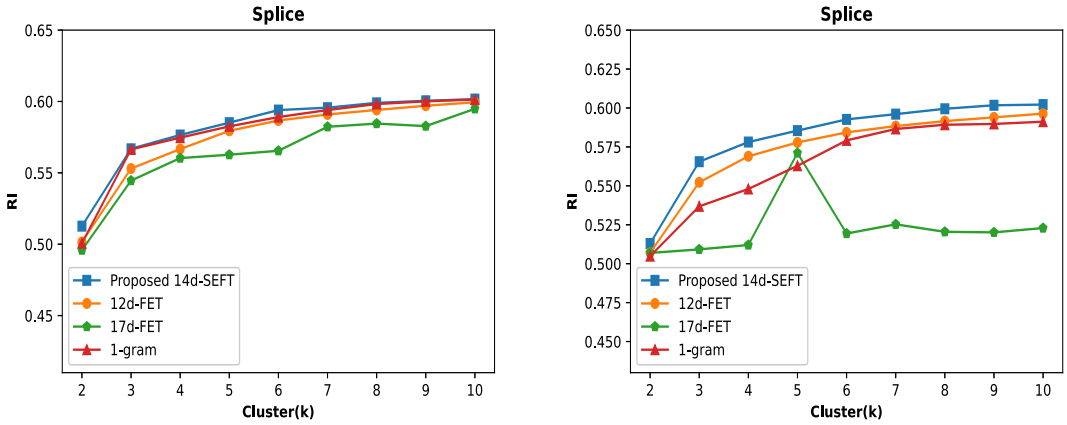
Table 4.10: Results on Splice dataset

Evaluation measures	Clusters	Algorithms							
		K-means				Fuzzy c-means			
		Propoosed 14d-SFET	12D-FET [50]	17D-FET [90]	1-gram [78]	Propoosed 14d-SFET	12D-FET [50]	17D-FET [90]	1-gram [78]
JI	2	**0.2933**	0.2866	0.2857	0.2853	**0.2905**	0.2864	0.2858	0.2854
	3	**0.2474**	0.2333	0.2275	0.2471	**0.2802**	0.2319	0.2255	0.2459
	4	**0.2069**	0.1926	0.1841	0.2053	**0.2656**	0.1968	0.1979	0.2032
	5	**0.1764**	0.1667	0.1758	0.1703	**0.2476**	0.1733	0.1705	0.1716
	6	**0.1619**	0.1461	0.1566	0.1536	**0.2690**	0.1519	0.1383	0.1506
	7	**0.1376**	0.1294	0.1267	0.1330	**0.2688**	0.1361	0.1230	0.1341
	8	**0.1224**	0.1156	0.1150	0.1205	**0.2687**	0.1240	0.1100	0.1198
	9	**0.1187**	0.1070	0.1104	0.1086	**0.2307**	0.1166	0.1012	0.1084
	10	**0.1026**	0.0964	0.0965	0.0993	**0.2513**	0.1099	0.0927	0.0990
RI	2	**0.5126**	0.5015	0.4958	0.5006	**0.5129**	0.5069	0.5069	0.5048
	3	**0.5668**	0.5529	0.5445	0.5664	**0.5654**	0.5522	0.5091	0.5368
	4	**0.5765**	0.5666	0.5603	0.5746	**0.5781**	0.5689	0.5119	0.5479
	5	**0.5851**	0.5794	0.5627	0.5825	**0.5854**	0.5778	0.5710	0.5628
	6	**0.5939**	0.5866	0.5654	0.5890	**0.5926**	0.5843	0.5193	0.5791
	7	**0.5956**	0.5909	0.5822	0.5939	**0.5960**	0.5883	0.5252	0.5865
	8	**0.5990**	0.5940	0.5845	0.5981	**0.5994**	0.5915	0.5204	0.5892
	9	**0.6004**	0.5970	0.5827	0.6000	**0.6017**	0.5939	0.5201	0.5897
	10	**0.6016**	0.5994	0.5948	0.6013	**0.6021**	0.5963	0.5228	0.5911



(a) Using K-means (b) Using Fuzzy c-means

Figure 4.17: JI on Splice



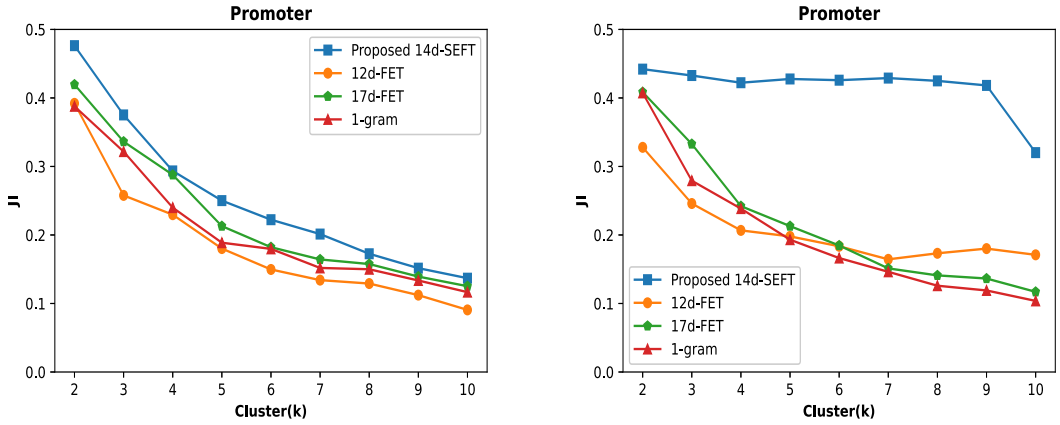
(a) Using K-means (b) Using Fuzzy c-means

Figure 4.18: RI on Splice

Results on Promoter dataset:

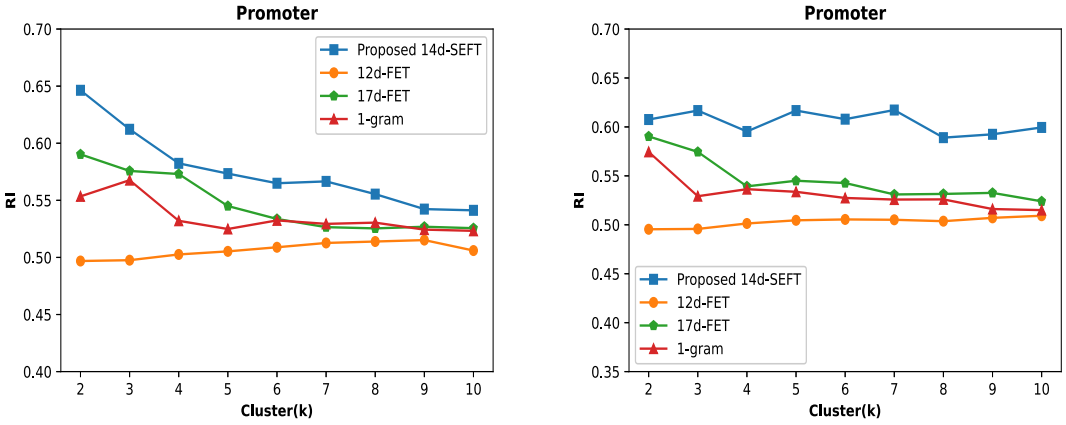
The experimental results of the 14d-SFET on the Promoter dataset using Fuzzy c-means and K-means with $k = [2, 10]$ are displayed in Table 4.11. For all k values, the proposed 14d-SFET provides superior JI values than 12d-FET [50], 17d-FET [90], and 1-gram [78] for both clustering approaches. Figure 4.19(a) and Figure 4.19(b) illustrate the comparison of JI values between the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78]. For all k values, it can be seen that the proposed

14d-SFET yields higher JI values than all other approaches. The comparison of the proposed 14d-SFET, 12d-FET [50], 17d-FET [90], and 1-gram [78] on RI using K-means and Fuzzy c-means is illustrated in Figures 4.20(a) and (b), respectively. The proposed method yields the highest RI values compared to all other methods.



(a) Using K-means (b) Using Fuzzy c-means

Figure 4.19: JI on Promoter



(a) Using K-means (b) Using Fuzzy c-means

Figure 4.20: RI on Promoter

Table 4.11: Results on Promoter dataset

Evaluation measures	Clusters	Algorithms							
		K-means				Fuzzy c-means			
		Propoosed 14d-SFET	12D-FET [50]	17D-FET [90]	1-gram [78]	Propoosed 14d-SFET	12D-FET [50]	17D-FET [90]	1-gram [78]
JI	2	**0.4763**	0.3920	0.4195	0.3877	**0.4420**	0.3279	0.4085	0.4073
	3	**0.3752**	0.2577	0.3362	0.3216	**0.4327**	0.2459	0.3327	0.2792
	4	**0.2934**	0.2296	0.2877	0.2399	**0.4222**	0.2066	0.2420	0.2384
	5	**0.2501**	0.1801	0.2129	0.1887	**0.4276**	0.1979	0.2129	0.1930
	6	**0.2222**	0.1496	0.1821	0.1796	**0.4259**	0.1835	0.1846	0.1662
	7	**0.2013**	0.1340	0.1642	0.1518	**0.4289**	0.1644	0.1512	0.1462
	8	**0.1725**	0.1291	0.1575	0.1499	**0.4249**	0.1732	0.1410	0.1259
	9	**0.1515**	0.1122	0.1392	0.1335	**0.4182**	0.1801	0.1364	0.1190
	10	**0.1369**	0.0906	0.1252	0.1165	**0.3202**	0.1710	0.1170	0.1037
RI	2	**0.6463**	0.4968	0.5902	0.5534	**0.6075**	0.4954	0.5902	0.5744
	3	**0.6122**	0.4975	0.5757	0.5676	**0.6167**	0.4957	0.5744	0.5292
	4	**0.5823**	0.5026	0.5730	0.5320	**0.5953**	0.5013	0.5392	0.5363
	5	**0.5734**	0.5053	0.5450	0.5248	**0.6167**	0.5045	0.5450	0.5336
	6	**0.5649**	0.5088	0.5336	0.5324	**0.6079**	0.5054	0.5426	0.5274
	7	**0.5665**	0.5126	0.5265	0.5293	**0.6172**	0.5051	0.5309	0.5257
	8	**0.5554**	0.5139	0.5254	0.5304	**0.5890**	0.5036	0.5315	0.5259
	9	**0.5423**	0.5151	0.5268	0.5243	**0.5924**	0.5070	0.5326	0.5160
	10	**0.5412**	0.5060	0.5256	0.5232	**0.5994**	0.5092	0.5239	0.5150

Distributed analysis of datasets:

Since the proposed approach is scalable, to check its scalability, we calculated the time required to extract features by varying the number of cores. We performed this analysis only for unlabeled datasets because these datasets contain a large number of sequences. The time required to extract features using proposed 14d-SFET on the unlabeled datasets is shown Table 4.12. It can be observed that the scalability reduces the enormous amount of time by processing the datasets on the higher number of cores. In case of Wm82.a1 and PI483463.a1 the time required to extract features on 4 cores is slightly higher than the 3 cores, this is due to because sometimes scalable approaches takes more time for data partitioning on various cores than data processing.

Table 4.12: Distributed analysis of datasets

Dataset	Time required (Seconds)			
	1 core	2 core	3 core	4 core
Williams82.a1	396	249	150	153
Williams82.a2	473	295	205	156
Williams82.a4	540	297	221	162
Lee.a1	425	276	151	150
ZH13.a1	344	196	151	151
PI483463.a1	316	168	149	151
W05.a1	531	297	237	153

4.4 Summary

This chapter offers a novel Apache Spark based 14-dimensional scalable feature extraction technique for genomics data consisting of four nucleotide bases A, T, G, and C, abbreviated as "14d-SFET". The proposed 14d-SEFT extracts most important features, i.e., the sequence length and entropy features, to increase the efficiency of the feature extraction process. In addition, it utilizes the distributed computing architecture to compute the task in parallel and also save time. Moreover, by adopting the power formulation, the proposed approach solves the problem of total distance and distribution features, which can sometimes be the same for dissimilar sequences

as identified in the existing method. Experimental results demonstrate that the proposed method generates the generalized strategy for feature extraction regardless of the evaluation method employed. For both K-means and Fuzzy c-means clustering, the proposed method performed better in terms of internal evaluation measures and external evaluation measures compared to existing methods. For all real-life plant genome datasets, the proposed method's scalability saves a substantial amount of time by distributing the computation on a higher number of cores. Therefore, we can conclude that the proposed method performs better than the other state-of-the-art methods in both K-means and Fuzzy c-means clustering and generates 14-dimensional features regardless of the evaluation model employed.

The proposed 14d-SFET extracts the most important features based solely on sequence arrangement and length. However, this method does not extract features based on nucleotide classification using chemical properties. Therefore, to further enhance the performance, we introduced another scalable feature extraction method. This method focuses on extracting important features based on the classification of nucleotides using their chemical properties, as discussed in Chapter 5.

Chapter 5

A Novel 13-Dimensional Alignment-Free Scalable Feature Extraction Method for Genomic Data Clustering

The method presented in Chapter 4 retrieves features depending on sequence layout and length. However, this method does not retrieve features based on chemical properties, which would provide additional information about genome functionality. Hence, in this chapter, we proposed a scalable, 13-dimensional feature extraction method (13dim-SFA) that extracts key features based on the chemical properties of nucleotides. The proposed method transforms the genome sequences into three novel sequences using the chemical properties of nucleotides. Furthermore, it computes the position distribution and local frequency distribution from these three novel sequences to compute the context based features in terms of entropies. In addition, this method utilizes Apache Spark to preprocess large raw genomes by distributing tasks across multiple cores. After extracting features, we clustered genome sequences with K-means and assessed performance in terms of the Silhouette Index and Davies-Bouldin Index. This approach is tested on five unlabeled real-life plant genome datasets of rice and wheat crops obtained from the rice genome library [51] and Han et al. [52], respectively. The experimental results show that the proposed approach performs well in comparison to state-of-the-art approaches.

5.1 Introduction

The proposed 13dim-SFA provides a comprehensive methodology for analyzing genome sequences by combining classification, feature extraction, and clustering techniques. Sequences are initially classified into three groups based on the presence of pyrimidine or purine classes, keto or amino groups, and strong or weak hydrogen bond groups, resulting in a foundational framework. The spatial organization and variability of nucleotides within the sequences are then captured by extracting position and distribution-based features, and quantifying them in terms of entropy. Additionally, the method includes extracting sequence length as an important feature. These features, which include 13 distinct descriptors, provide a multidimensional representation of sequence structure and distribution. Using this extensive feature set, clustering analysis is used to group sequences based on their similarity in feature space. This clustering approach sheds light on the complexity and diversity of genome sequences by revealing common structural motifs, functional properties, and evolutionary relationships. The proposed approach's integrative methodology makes it easier to classify, compare, and interpret sequence data, making it useful in genomics and bioinformatics research. The proposed approach is discussed in detail in subsequent section.

5.2 Proposed 13-Dimensional Alignment-Free Scalable Feature Extraction Method

The proposed 13dim-SFA extracts the features from genome sequences in terms of sequence length and entropy values. The proposed approach performs feature extraction in six steps, which are discussed as follows. The flow diagram of the proposed approach is presented in Figure 5.1.

First step:

In this step, we extracted the length of the sequence (L). The length of the sequence varies from organism to organism and thus provides useful information for clustering. Hence, we considered the length of the sequence as a feature. In the example presented

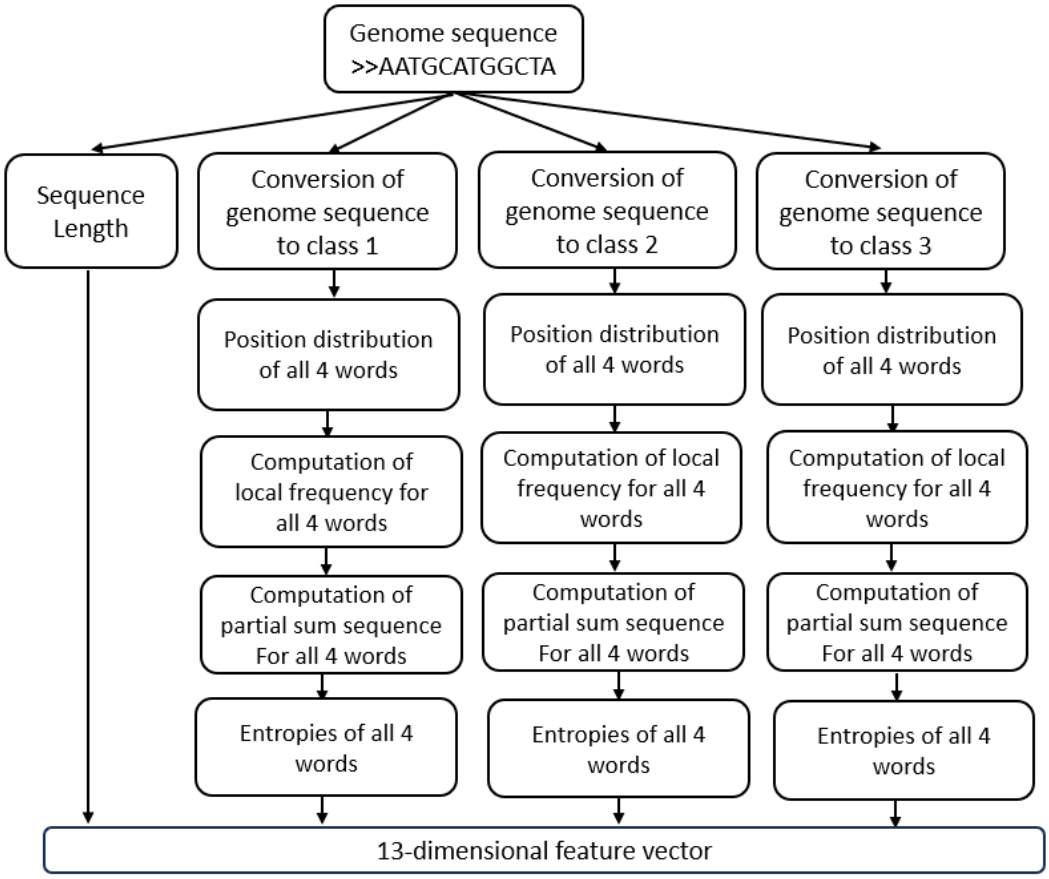


Figure 5.1: Flow diagram of proposed approach

in Figure 5.1, the total number of nucleotides are 12. So, the value of L will be 12.

Second step:

In this step, we performed the classification and transformation of nucleotides [81] in three categories using their chemical properties. In the first category, the nucleotides are classified in pyrimidine or purine class. In the second category, the nucleotides are classified in keto or amino class, and in the third category, the nucleotides are classified on the basis of hydrogen bond, i.e., strong hydrogen bond group or weak hydrogen bond group. The classification of these sequences are presented subsequently.

- **Pyrimidine and purine class (class 1):** The pyrimidine class (D) consists of two nucleotides named "C" and "T". On the other hand, the purine class (R) consists of two nucleotides named "A" and "G".

- **Keto and amino class (class 2):** The keto class (E) has two nucleotides

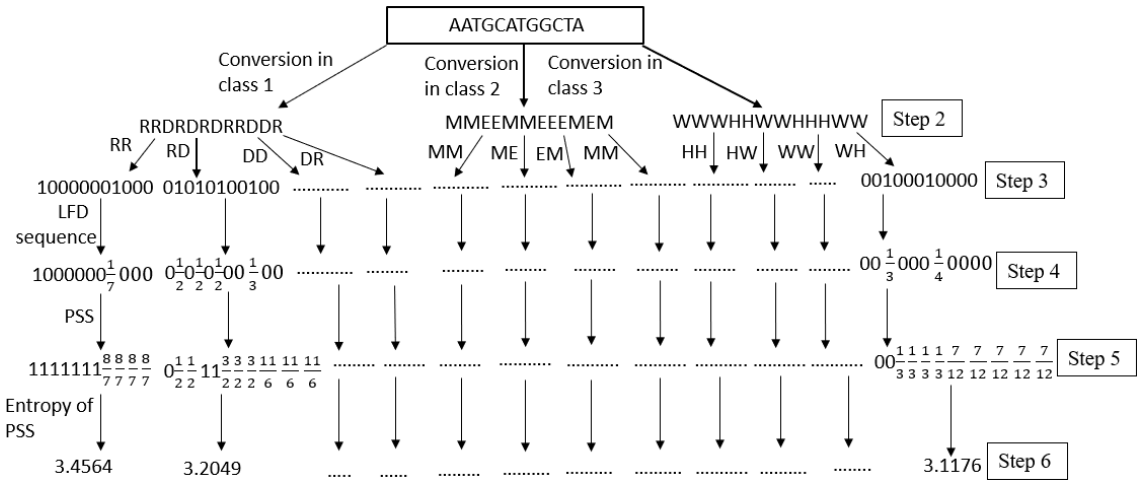


Figure 5.2: Example of genome sequence

named "G" and "T" where as the amino class (M) consists of two nucleotides "A" and "C".

- **Strong hydrogen bond and weak hydrogen bond class (class 3):** The strong hydrogen bond class (H) has "C" and "G". On the other hand, weak hydrogen bond group (W) consists of "A" and "T".

After conversion of nucleotides into these three classes, we obtained the three transformed sequences. After that, we created the words of length $(k) = 2$. So, for each class, there will be 2^k words. For example, class 1 produces the words RR, RD, DD, DR. Similarly, the other two classes will also produce the four words each. Therefore a total of 12 words are created by the three classes. An example sequence is depicted in Figure 5.2. In this figure, the process of converting an example sequence into three distinct classes is depicted in step 2.

Third step:

In this step, we calculated the position distribution sequence (PDS) for each word within every class. With the help of this process, we obtained the positional information for a particular word, enabling the provision of context-based information for sequences. To derive the position distribution, we utilized a sliding window spanning two characters, initially positioning it at the onset of the transformed sequence. If the particular word appeared at that position, we denoted it as 1; otherwise, we marked

the position as 0. Subsequently, we shifted the sliding window by one character and reassessed the presence of the specific word. This process iterated until the sequence concluded. For example, a sequence is presented in Figure 5.2, and its PDS is shown in step 3.

Fourth step:

In this step, we created a new sequence by using the local frequency distribution (LFD) with the help of the previously calculated PDS. Wei et al. [12] described that local frequency considers both the position and density of a word in a text. They suggested that, the single local frequency distribution may not provide the complete global context of a word, but a sequence of these distributions gives a more accurate and clear depiction of the word's overall distribution when compared to a global frequency analysis. The local frequency is calculated for every occurrence of a word by computing the distance between positions using Eq. (5.1).

$$LF_i^f = \frac{1}{p_i^f - p_{i-1}^f}, \tag{5.1}$$

where LF_i^f is the local frequency, p_i^f is the position of the i^{th} occurrence and p_{i-1}^f denotes the position of $i-1^{th}$ occurrence of the word f. The value of p_0^f is considered to be 0. For example, consider the first PDS presented in step 3 in Figure 5.2. The LFD sequence corresponding to that sequence is presented in the first sequence of step 4.

Fifth step:

In this step, the partial sum sequence (PSS) is computed with the help of LFD sequence. The procedure for computation of partial sum sequence is as follows. Let LFD sequence be $S = Z_1, Z_2, Z_3, ..., Z_n$. So its PSS will be $PSS_S = Y_1, Y_2, Y_3, ..., Y_n = (Z_1), (Z_1 + Z_2), (Z_1 + Z_2 + Z_3), ..., \sum_{k=1}^{n}(Z_k)$. For example, consider the first LFD presented in step 4 in Figure 5.2. The PSS sequence corresponding to that sequence is presented in the first sequence of step 5.

Sixth step:

In this step, the entropy (E) of a given PSS is computed using Eq. (5.2).

$$E = -\sum_{k=1}^{n} p_k log p_k, \tag{5.2}$$

where p_k denotes the discrete probability at position k and computed using Eq. (5.3).

$$p_k = \frac{Z_k}{T}, \tag{5.3}$$

where k ranges from 1 to n. n represents the length of the PSS. T denotes the sum of PSS sequence, and Z_k represents the value at position k in PSS. For example, the entropy values of an example sequence shown in Figure 5.2 are presented in step 6.

Finally, we make a feature vector by combing the entropy values of all 12 words and the length of the sequence (L). So we get a feature vector having 13 dimensions. In the following subsection, we will examine the implementation of proposed 13dim-SFA on Apache Spark to make it scalable.

Scalable 13dim-SFA on Apache Spark cluster:

To deploy the proposed 13dim-SFA on the Apache Spark cluster, we initially converted the dataset into Resilient Distributed Dataset (RDD) format, as shown in steps 1 and 2 of Algorithm 5.1. This conversion allows the dataset to be partitioned across multiple nodes, facilitating parallel processing for distributed computing objectives. After that, each sequence in the RDD is mapped to the feature extraction procedure using the *map*() function in order to extract features from them. The procedure for feature extraction includes the implementation of an algorithm designed to extract features. In the third step, the extracted features are collocated from every worker node using the *collect*() function. Finally, in the last stage, all the accumulated features in a vectorized format are saved in an output file in order to facilitate the clustering process of these sequences. Figure 5.3 depicts the execution of the proposed 13dim-SFA within an Apache Spark cluster. The pseudo code of proposed 13dim-SFA is presented in Algorithm 5.1.

In Algorithm 5.2, the pseudo code for the *Feature_Extraction_Procedure*() is described. During the initial phase, the sequence length (L) is set to 0 and an empty array of feature vector (V) is generated. We measured the L in the second through

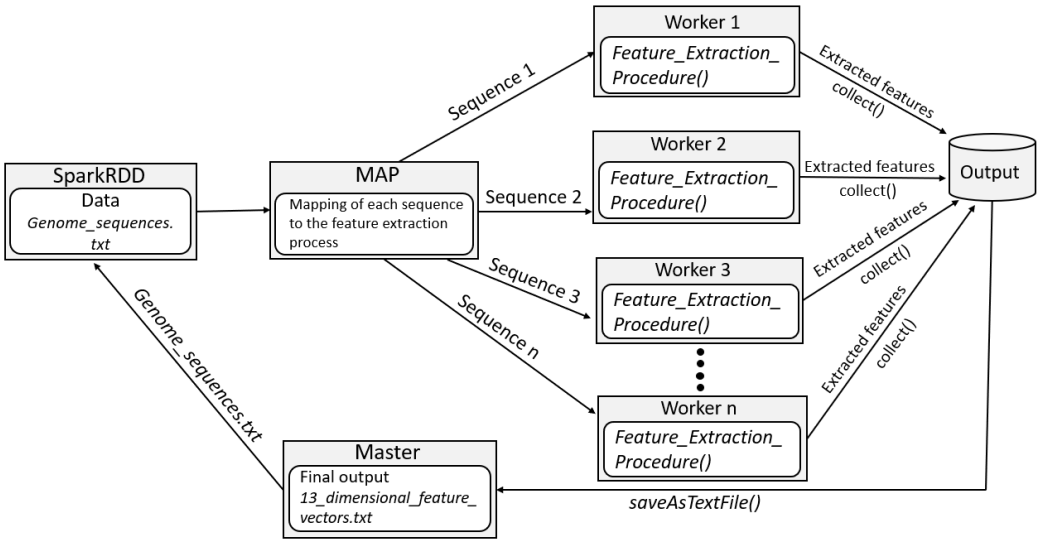


Figure 5.3: Execution of proposed 13dim-SFA on Apache Spark

Algorithm 5.1 13dim-SFA

Input: $Genome\_sequences.txt$

Output: $13\_dimensional\_feature\_vectors.txt$

1: $x_1 = SparkContext.textFile(Genome\_sequences.txt)$

2: $x_1 = x_1.map(lambda\ y : numpy.array(y))$

3: $x_1 = x_1.map(lambda\ y : Feature\_Extraction\_Procedure(y))$

4: $x_1 = x_1.collect()$

5: $x_1.saveAsTextFile(13\_dimensional\_feature\_vector.txt)$

fifth steps. In the sixth and seventh steps, the sequence transforms into three classes. In eighth to twelfth steps, we computed the PDS, LFD, PSS, and entropy for every word. Finally, in the last stage, the 13-dimensional feature vector is obtained. The proposed 13dim-SFA is applied on various real-life plant genome datasets and the experimental findings are presented subsequently.

5.3 Experimental Evaluation

In order to assess the effectiveness of the suggested 13dim-SFA, the K-means [162] clustering method is employed. In order to determine the optimal number of clusters (k) for every dataset, several experiments are conducted by considering alternative

Algorithm 5.2 *Feature_Extraction_Procedure*()

Input: Genome sequence (G) having characters A, T, G, and C

Output: 13_dimensional_feature_vector

Initialize the length of the sequence (L) with 0 and an empty feature vector (V).
for Each character in G **do**
  $L = L + 1$ (Increment the length of the sequence by 1)
end for
Append L to feature vector V.
for Every class (C) **do**
  $C(G) \leftarrow Genomesequence(G)$ (Transforming the G into class)
  for Every word $(w) \in C(G)$ **do**
    Create positional distribution sequence $PDS_w \leftarrow C(G)$.
    Create local frequency distribution sequence $LFD_w \leftarrow PDS_w$.
    Calculate partial sum sequence $PSS_w \leftarrow LFD_w$.
    Calculate entropy E_w of PSS_w using Eq. (5.3).
    Append E_w to feature vector V.
  end for
end for

values of k. The performance of the suggested technique is assessed on a Dell Workstation, which is equipped with an Intel Xeon W-2102 CPU, 64 GB of RAM, and 4 cores. The suggested methodology was executed on Apache Spark to assess its scalability. The detailed description of Apache Spark is presented in Section 2.4.

This section comprises of four subsections. The initial subsection presents a comprehensive overview of the experimental datasets. The performance evaluation metrics are described in the second subsection. The subsequent section provides an analysis of the parameter configurations pertaining to the various parameters. The experimental results for the real-life plant genome datasets are outlined in the fourth subsection.

5.3.1 Dataset Details

We performed the experiments using five real-life plant genome datasets of rice and wheat crops obtained from rice genome library [51] and from Han et al. [52], respectively. In the rice dataset, there are 12 chromosomes (ch) present. We merged three sets of chromosomes and created four datasets of rice named Rice 1, Rice 2, Rice 3, and Rice 4. The detailed description of these datasets are given in Section 2.6.1.

5.3.2 Evaluation Measures

To evaluate the performance of the proposed method, we employed two metrics, Silhouette Index (SI), Davies-Bouldin Index (DBI). The details of these metrics are given in Section 2.5.

5.3.3 Hyperparameter Settings for Evaluation Model

Table 5.1 presents a comprehensive list of the parameters utilized in the experiments, together with their corresponding values.

Table 5.1: Hyperparameter settings for evaluation models

Evaluation Model	**Hyperparameter**	**Parameter Value**
K-means	*n_init*	10
K-means	*Maximum iteration*	300

5.3.4 Experimental Analysis

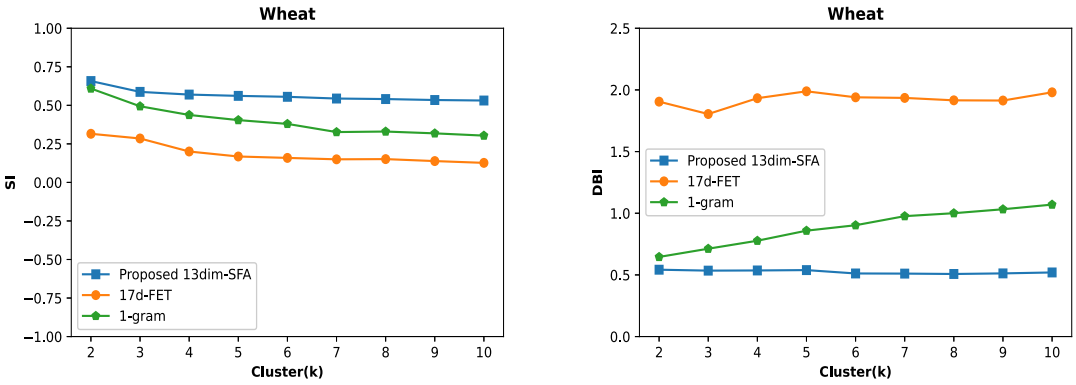
We ran trials on the datasets with varying numbers of clusters (k) and only gave the experimental findings for a few of the best-performing k values in terms of SI and DBI in contrast to 1-gram [78], 17d-FET [90]. In order to assess the efficacy of the proposed scalable approach in distributed computing, we conducted experiments on several configurations, including single core, two cores, three cores, and four cores. The time taken to extract features for each configuration was recorded and displayed in Table 5.7.

Experimental findings on Wheat dataset:

Table 5.2 shows the experimental results on the Wheat data on $k = 2$ to 10. It can be observed that the proposed 13dim-SFA gives a higher SI and lower DBI for all k values in comparison to 17d-FET [90] and 1-gram [78]. The comparison graphs on SI and DBI values among the proposed 13dim-SFA, 17d-FET, and 1-gram are presented in Figures 5.4(a) and (b), respectively. It can be seen from the graphs that proposed approach gives better results in comparison to other approaches.

Table 5.2: Results on Wheat Dataset

Count of clusters	SI			DBI		
	Proposed 13dim-SFA	17d-FET	1-gram	Proposed 13dim-SFA	17d-FET	1-gram
2	**0.6575**	0.3154	0.6090	**0.5422**	1.9048	0.6457
3	**0.5873**	0.2850	0.4939	**0.5347**	1.8045	0.7124
4	**0.5698**	0.2005	0.4376	**0.5361**	1.9328	0.7771
5	**0.5614**	0.1681	0.4042	**0.5390**	1.9888	0.8587
6	**0.5553**	0.1590	0.3796	**0.5117**	1.9401	0.9028
7	**0.5441**	0.1498	0.3266	**0.5106**	1.9352	0.9764
8	**0.5407**	0.1512	0.3300	**0.5075**	1.9156	1.0003
9	**0.5345**	0.1384	0.3182	**0.5123**	1.9139	1.0323
10	**0.5312**	0.1268	0.3036	**0.5199**	1.9806	1.0701



(a) SI (b) DBI

Figure 5.4: Results on Wheat dataset

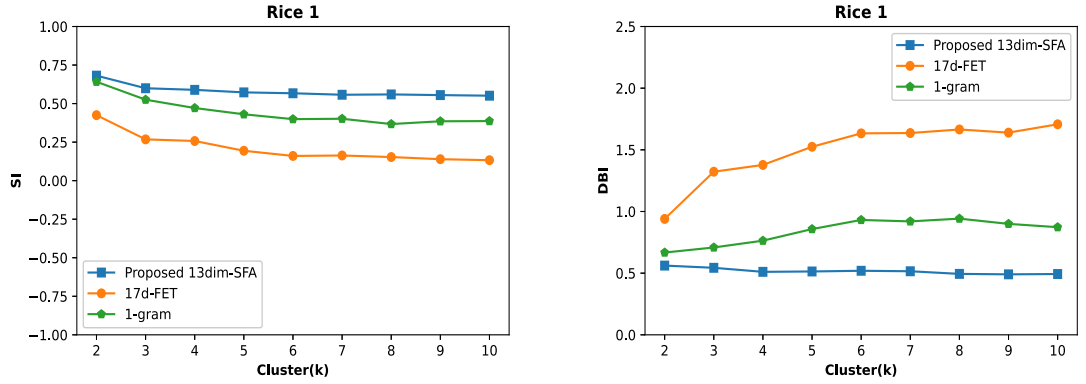
Experimental findings on Rice 1:

The experimental findings on the Rice 1 data for values of k ranging from 2 to 10 are

presented in Table 5.3. The results indicate that the 13dim-SFA method consistently yields greater SI values and lower DBI values across all values of k, when compared to the 17d-FET [90] and 1-gram [78] methods. Graphs comparing the SI and DBI values of the 13dim-SFA, 17d-FET, and 1-gram are shown in Figures 5.5(a) and (b), respectively. The graphs demonstrate that the proposed method yields superior results in comparison to other methods.

Table 5.3: Results on Rice 1 Dataset

Count of clusters	SI			DBI		
	Proposed 13dim-SFA	17d-FET	1-gram	Proposed 13dim-SFA	17d-FET	1-gram
2	**0.6813**	0.4250	0.6416	**0.5614**	0.9398	0.6671
3	**0.5995**	0.2679	0.5253	**0.5434**	1.3226	0.7083
4	**0.5891**	0.2573	0.4712	**0.5107**	1.3776	0.7634
5	**0.5727**	0.1940	0.4306	**0.5137**	1.5248	0.8574
6	**0.5669**	0.1603	0.3991	**0.5193**	1.6336	0.9315
7	**0.5573**	0.1633	0.4015	**0.5154**	1.6363	0.9197
8	**0.5594**	0.1532	0.3672	**0.4941**	1.6651	0.9422
9	**0.5550**	0.1393	0.3852	**0.4904**	1.6394	0.8996
10	**0.5512**	0.1328	0.3865	**0.4930**	1.7072	0.8727



(a) SI (b) DBI

Figure 5.5: Results on Rice 1 dataset

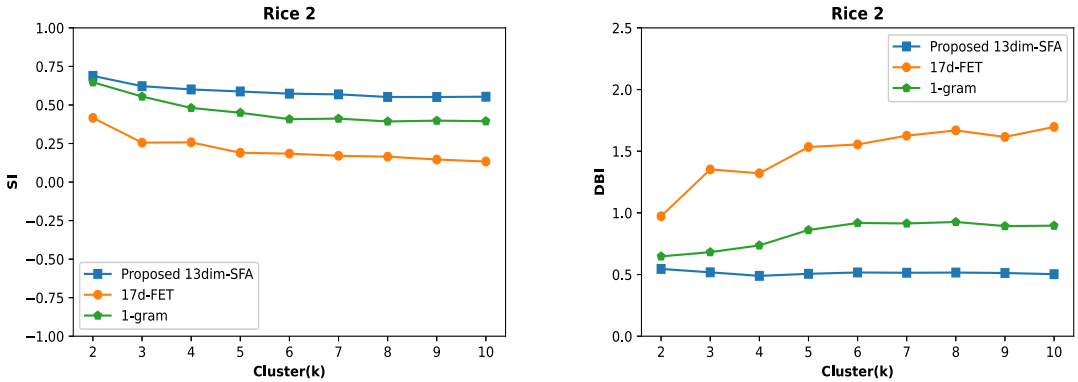
Experimental findings on Rice 2:

Table 5.4 shows the results of experiments with Rice 2 data for k = 2–10. Compared

to 17d-FET [90] and 1-gram [78], the proposed 13dim-SFA gives the highest SI and the lowest DBI for all k values. Figures 5.6(a) and (b) depict comparisons of SI and DBI values for the 13dim-SFA, 17d-FET, and 1-gram. It is evident from the graphs that the proposed method yields superior results as compared to other methods.

Table 5.4: Results on Rice 2 Dataset

Count of clusters	SI			DBI		
	Proposed 13dim-SFA	17d-FET	1-gram	Proposed 13dim-SFA	17d-FET	1-gram
2	**0.6886**	0.4164	0.6473	**0.5454**	0.9728	0.6474
3	**0.6219**	0.2559	0.5547	**0.5181**	1.3524	0.6815
4	**0.6007**	0.2577	0.4808	**0.4891**	1.3211	0.7364
5	**0.5876**	0.1898	0.4495	**0.5058**	1.5342	0.8609
6	**0.5734**	0.1843	0.4078	**0.5168**	1.5548	0.9182
7	**0.5688**	0.1704	0.4115	**0.5143**	1.6259	0.9143
8	**0.5518**	0.1647	0.3930	**0.5161**	1.6687	0.9261
9	**0.5516**	0.1463	0.3984	**0.5124**	1.6156	0.8929
10	**0.5535**	0.1331	0.3948	**0.5030**	1.6972	0.8963



(a) SI (b) DBI

Figure 5.6: Results on Rice 2 dataset

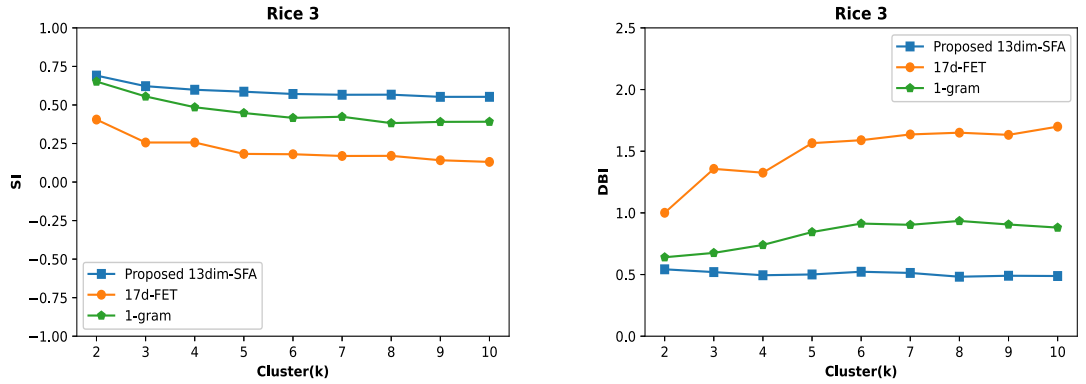
Experimental findings on Rice 3:

Table 5.5 presents the results of the experiments conducted on Rice 3 data ranging from k = 2 to 10. When compared to 17d-FET [90] and 1-gram [78], the suggested 13dim-SFA provides a greater SI and a lower DBI for all of the different k values.

The comparison graphs on SI and DBI values among the proposed 13dim-SFA, 17d-FET, and 1-gram are presented in Figures 5.7(a) and (b), respectively. It can be seen from the graphs that proposed approach gives better results in comparison to other approaches.

Table 5.5: Results on Rice 3 Dataset

Count of clusters	SI			DBI		
	Proposed 13dim-SFA	17d-FET	1-gram	Proposed 13dim-SFA	17d-FET	1-gram
2	**0.6906**	0.4053	0.6518	**0.5424**	1.0007	0.6402
3	**0.6219**	0.2565	0.5553	**0.5199**	1.3568	0.6754
4	**0.5987**	0.2566	0.4853	**0.4943**	1.3259	0.7400
5	**0.5861**	0.1826	0.4478	**0.5007**	1.5654	0.8444
6	**0.5713**	0.1804	0.4163	**0.5226**	1.5893	0.9134
7	**0.5662**	0.1691	0.4236	**0.5130**	1.6360	0.9032
8	**0.5668**	0.1699	0.3822	**0.4823**	1.6502	0.9346
9	**0.5526**	0.1414	0.3903	**0.4902**	1.6326	0.9058
10	**0.5528**	0.1306	0.3913	**0.4880**	1.6992	0.8807



(a) SI (b) DBI

Figure 5.7: Results on Rice 3 dataset

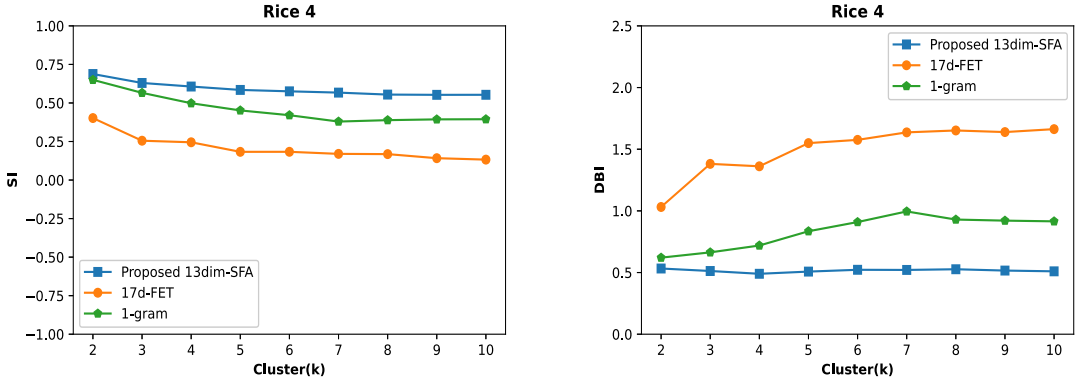
Experimental findings on Rice 4:

The experimental findings on the Rice 4 data for values of k ranging from 2 to 10 are presented in Table 5.6. The results indicate that the 13dim-SFA method consistently yields greater SI values and lower DBI values across all values of k, when compared

to the 17d-FET [90] and 1-gram [78] methods. Graphs comparing the SI and DBI values of the 13dim-SFA, 17d-FET, and 1-gram are shown in Figures 5.8(a) and (b), respectively. The graphs demonstrate that the proposed method yields superior results in comparison to other methods.

Table 5.6: Results on Rice 4 Dataset

Count of clusters	SI			DBI		
	Proposed 13dim-SFA	17d-FET	1-gram	Proposed 13dim-SFA	17d-FET	1-gram
2	**0.6878**	0.4021	0.6499	**0.5327**	1.0315	0.6207
3	**0.6298**	0.2551	0.5658	**0.5125**	1.3812	0.6636
4	**0.6066**	0.2450	0.4983	**0.4899**	1.3615	0.7191
5	**0.5846**	0.1830	0.4517	**0.5077**	1.5495	0.8348
6	**0.5755**	0.1832	0.4206	**0.5223**	1.5758	0.9094
7	**0.5672**	0.1698	0.3794	**0.5211**	1.6366	0.9955
8	**0.5544**	0.1682	0.3885	**0.5267**	1.6520	0.9296
9	**0.5527**	0.1418	0.3936	**0.5160**	1.6390	0.9209
10	**0.5530**	0.1325	0.3945	**0.5098**	1.6628	0.9151



(a) SI (b) DBI

Figure 5.8: Results on Rice 4 dataset

Run time performance:

To test the scalability of the proposed method, we calculated the time required to extract features by altering the number of cores. Table 5.7 displays the time required to extract features from unlabeled datasets using the proposed 13dim-SFA algorithm.

It can be seen that by processing datasets on a greater number of cores, scalability significantly reduces the amount of time required.

Table 5.7: Distributed analysis of datasets

Dataset	Time required (Seconds)			
	1 core	2 core	3 core	4 core
Wheat	1277	772	516	467
Rice 1 (ch1, 2, and 3)	1821	965	631	621
Rice 2 (ch4, 5, and 6)	1518	773	546	527
Rice 3 (ch7, 8, and 9)	1215	640	467	437
Rice 4 (ch10, 11, and 12)	1190	633	469	436

5.4 Summary

In this chapter, a novel 13-dimensional alignment-free scalable feature extraction approach (13dim-SFA) for extracting features from genomic data containing the four nucleotide bases A, T, G, and C has been presented. Using chemical property-based features, the proposed 13dim-SFA is able to perform clustering efficiently and generates features that provide more information for sequence analysis. Moreover, the proposed method makes use of local frequency instead of global frequency and thus provides a more precise and clear representation of the word's overall distribution. In addition, we extracted the most important characteristic, namely the sequence length, to improve the efficiency of the proposed technique for feature extraction. The scalability of the proposed method reduces the quantity of computing time by distributing the tasks across multiple cores. The experimental results demonstrate that the proposed 13dim-SFA yields a higher SI and a lower DBI in all real-world plant genome datasets of rice and wheat crops, indicating that the proposed method outperforms other state-of-the-art approaches and performs well in the clustering of similar genome sequences. Therefore, it can be concluded that the proposed 13dim-SFA performs satisfactorily for measuring sequence similarity and clustering.

However, the proposed 13dim-SFA employs K-means clustering to group similar genome sequences. A significant drawback of K-means clustering is its inability to

handle real-time dynamic data, which is abundant in the field of plant genomics, as it generates a lot of dynamic data nowadays. Hence, to address this challenge and enable clustering of real-time dynamic data, we introduced a multi-objective based incremental clustering method in Chapter 6.

Chapter 6

A Novel Multi-Objective Based Incremental Clustering Method for Dynamic Data Analysis

The clustering methods (K-means and Fuzzy c-means) employed in Chapters 4 and 5 perform well for handling the static data. However, these methods are unable to handle the real-time dynamic data generated. To address this issue, in this chapter, we proposed a multi-objective incremental clustering method for processing dynamic data that generates and updates clusters in real-time. To improve the dynamic clustering process, the proposed method employs Euclidean distance to calculate the similarity between data points and constructs a fitness function with three primary clustering objective functions: inter-cluster distance, intra-cluster distance, and cluster density. The proposed method employs the concept of objective weighting, which allocates a weight to each objective in order to generate a single Pareto-optimal solution for the constructed fitness function. The proposed approach is tested on five labeled benchmark datasets from the UCI machine learning repository to test its efficacy.

6.1 Introduction

Nowadays, a lot of dynamic data is being generated; for example, every day, almost a billion people conduct search on Google. It is estimated that daily email traffic

is around 300 billion. Every single day, people around the world compose about 230,000,000 tweets [43]. More than 30 petabytes of data created by Facebook users are stored, accessed, and analyzed by the social media platform. The analysis of the dynamic data stream can be helpful to derive various conclusions by using the data mining approaches. Despite this, analyzing large dynamic data streams are also challenging since the samples in a stream typically depend on time, and the underlying pattern which may evolve with time. To extract knowledge from this large amount of real-time data, this data should be processed using various data mining approaches involves clustering for identifying patterns and structures. This study will be helpful in generating scientific results and making the decisions.

One method for dealing with real-time dynamic data is incremental clustering [45]. One of the biggest problems with incremental clustering is that most of it only uses a similarity-based measure and a single objective to cluster the dynamic data points together [46]. Because of this, most clustering algorithms are not strong against changes in the shape, size, dimensionality, and other properties of the clusters. Multi-objective clustering is used to deal with this problem. It breaks up a dataset into groups of similar items, optimizing multiple goals at the same time. Multi-objective clustering can be thought of as a special case of multi-objective optimization [96], which tries to find the best trade-offs between different goals while taking into account certain constraints.

In this study, we came up with a new multi-objective based incremental clustering (MOB-IC) method for analyzing dynamic data points. This method uses the Euclidean distance to find similarities between dynamic data points and simultaneously optimizes three major clustering objective functions, i.e., the distance between clusters, the distance within clusters, and the cluster density, by using objective weighting concept. The proposed method groups the dynamic data points so that the distance between the data points within each cluster is as small as possible, the distance between clusters is as large as possible, and the cluster density is as large as possible within the cluster. The detailed methodology of proposed MOB-IC is presented subsequently.

6.2 Proposed Multi-Objective Based Incremental Clustering

The proposed approach simultaneously optimizes the intra-cluster distance, inter-cluster distance, and cluster density objective functions for the clustering of dynamic data points. Hence, we provided a brief overview of these three key clustering objective functions in preliminaries. After that, we discussed the proposed approach.

6.2.1 Preliminaries

In the proposed approach, we used three objective functions intra-cluster distance, inter-cluster distance, and cluster density that needs to be optimized, which are discussed in detail subsequently.

Intra-cluster distance:

Each cluster has its own intra-cluster distance, which is found by taking the average of the distances from each data point to the center of its cluster as given by Eq. (6.1).

$$obj_1 = \frac{1}{|clus_i|} \sum_{a \in clus_i} dis(a, cen_i), \tag{6.1}$$

where $clus_i$ is the i^{th} cluster and cen_i is the center of cluster $clus_i$. The $dis(a, cen_i)$ is a function that computes the Euclidean distance between the data point a and cen_i. In a cluster, the distance between the data points and the center of the cluster should be smaller for better clustering. As a result, the proposed method seeks to minimize the intra-cluster distance as obj_1.

Inter-cluster distance:

The inter-cluster distance represents the distance between one cluster and the other. The inter-cluster distance is computed by using Eq. (6.2).

$$obj_2 = \frac{1}{|clus_i|\,|clus_j|} \sum_{a \in clus_i, b \in clus_j} dis(a, b), \tag{6.2}$$

where $clus_i$ and $clus_j$ represents the i^{th} and j^{th} cluster respectively, a and b are any

data point belonging to $clus_i$ and $clus_j$, respectively. The $dis(a, b)$ is the Euclidean distance between the data point a and b. The clustering is optimal when the different clusters are more separated. As a result, the proposed approach aims to maximize the inter-cluster distance as obj_2.

Cluster density:

To improve the cluster density, the distance of a data point from the other data points in the same cluster should be smaller. This makes the clustering more crisp than fuzzy. As given in Eq. (6.3), cluster density is formulated to be proportional to the sum of these distances, so to maximize cluster density, the objective function aimed to be minimized as obj_3.

$$obj_3 = \frac{1}{|clus_i|} \sum_{a,b \in clus_i} dis(a, b), \tag{6.3}$$

where $clus_i$ is the i^{th} cluster and a, b are two data points belonging to $clus_i$. The $dis(a, b)$ is the Euclidean distance between the data point a and b.

6.2.2 Proposed Method

In the proposed MOB-IC method, the incoming data point is put into the clustering process in three steps. In the first step, when a new data point arrives in the clustering process, its similarity with all other clusters is evaluated by measuring its distance from the cluster centers of all other existing clusters using Eq. (6.4). If no cluster exists before a new data point arrives, then a new cluster is formed by containing that data point. In the second step, the cluster with the greatest similarity to the arriving data point is identified and the fitness function value for that cluster is calculated using the Eq. (6.6). Finally, in the last step, the calculated fitness function value is compared with an optimal maximum threshold ($Maximum_{thershold}$) value. If the calculated fitness value is less than or equal to the value of the maximum threshold, then the arriving data point is assigned to the most similar cluster, and the cluster center of that cluster is updated by taking the average of all data points; otherwise, a new cluster is formed with the arriving data point. The working of the proposed

approach is given in Algorithm 6.1. The flow diagram of the proposed work is shown in Figure 6.1.

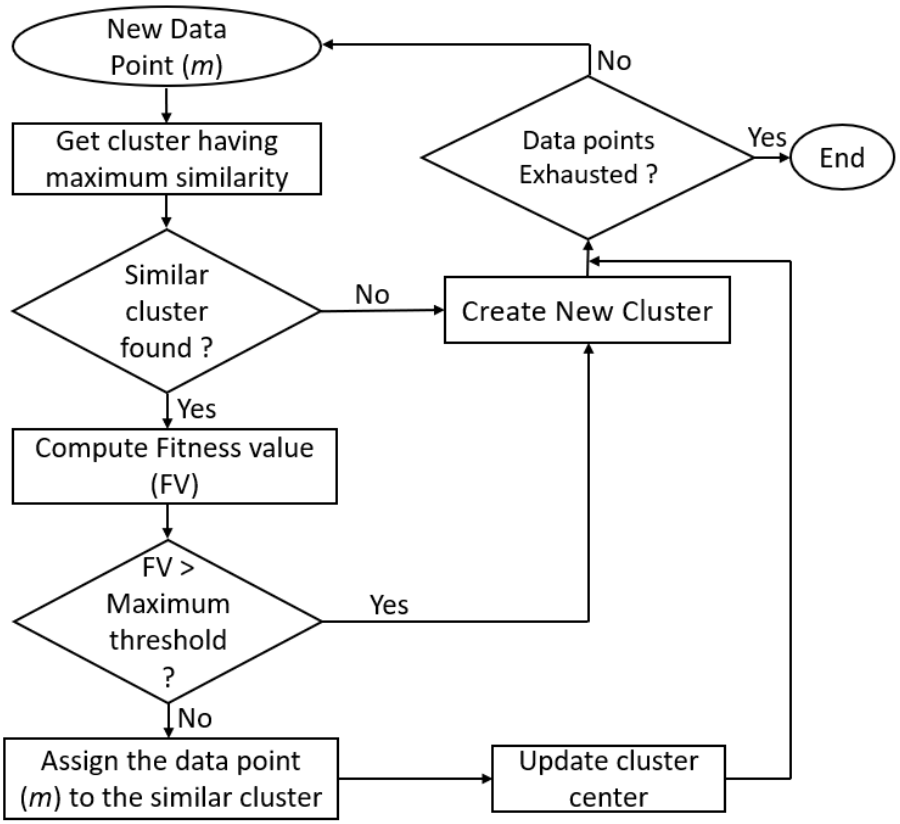


Figure 6.1: Flowchart of proposed approach

Calculation of similarity:

The similarity of a data point p from a cluster center (cen_i) is computed by using Eq. (6.4).

$$Similarity = \frac{1}{1 + dis(p, cen_i)}, \tag{6.4}$$

where $dis(p, cen_i)$ is the Euclidean distance between the data point p and i^{th} cluster center.

Algorithm 6.1 *Dynamic_Data_Clustering* $(MOB - IC)$

Input: Dynamic data points

Output: Dynamic data clustering and their analysis

Let m is a new incoming data point.
$clus_{num} = 0$ (Initialization)
for m in Input data points **do**
  $clus_i = Get\_Similar\_Cluster(m)$
  if $clus_i = 0$ **then**
    Create a new cluster for the first arriving data point.
    $cen_i \leftarrow Cluster\_Center\_Calculation(clus_i)$. (calculation of cluster center of newly created cluster).
    $clus_{num} = clus_{num} + 1$
  else
    $clus_{num} \leftarrow Update\_cluster(m, clus_i)$
  end if
end for

Algorithm 6.2 *Get_Similar_Cluster*()

Input: data point (m)

Output: Index of cluster $(clus_i)$ having greatest similarity with data point m.

$clus_i = 0$ (Initialization)
$Maximum_{similarity}$ = -∞ (Initialization)
for each of the cluster k **do**, (Where k denotes the total number of existed clusters)
  Calculate similarity of data point m with an existing cluster using Eq. (6.4).
  if $Similarity > Maximum_{similarity}$ **then**
    $Maximum_{similarity} \leftarrow Similarity$
    $clus_i \leftarrow k$ (updating the cluster index)
  end if
end for
return $clus_i$

Algorithm 6.3 *Update_cluster*()

Input: m, $clus_i$

Output: $clus_{num}$

1: $cen_i \leftarrow Cluster\_Center\_Calculation(clus_i)$

2: Compute the fitness function value (FV) for cluster $clus_i$ using Eq. 6.6.

3: **if** $FV > Maximum_{thershold}$ **then**

4: Create a new cluster for the data point m.

5: $clus_{num} = clus_{num} + 1$

6: **else**

7: Assign the data point m to the cluster with the highest similarity $clus_i$.

8: $cen_i \leftarrow Cluster\_Center\_Calculation(clus_i)$ (Updating of cluster center after including the data point m.)

9: **end if**

10: return $clus_{num}$

Algorithm 6.4 *Cluster_Center_Calculation*()

Input: $clus_i$

Output: cen_i

1: Let d_1, d_2, d_3,. . . . . d_n are data points exists in cluster having index $clus_i$.

2: The cluster center $cen_i = \frac{d_1+d_2+d_3+.....d_n}{n}$

3: return cen_i

Calculation of fitness function value:

In the proposed approach, to simultaneously optimize the aforementioned three objectives, the fitness function is formed by combining all three objective functions with weight coefficients in a single fitness function. In this way, we converted the multi-objective optimization problem into single-objective problem. However, in the proposed approach the obj_1 and obj_3 are sought to be minimized and obj_2 is sought to be maximized, so instead of directly adding these objective functions, first we transformed the obj_2 in such a way that it has to be minimized as given in Eq. (6.5). We assigned a weight coefficient α to the obj_1, weight coefficient β to obj_2, and γ to obj_3. After that, to form the fitness function, we directly added all the objective functions with positive weight coefficients as given in Eq. (6.6).

$$obj_{2new} = \frac{1}{1 + obj_2}. \tag{6.5}$$

$$Fitness function = \alpha \frac{obj_1}{N_{obj_1}} + \beta \frac{obj_{2new}}{N_{obj_{2new}}} + \gamma \frac{obj_3}{N_{obj_3}}, \tag{6.6}$$

where the value of $\alpha + \beta + \gamma = 1$. The factors N_{obj_1}, $N_{obj_{2new}}$, and N_{obj_3} represent the cluster centers containing the data points used to normalize the clustering results. By optimizing the Eq. (6.6) we will generate a Pareto-optimal solution (optimized clusters) that makes a balance between obj_1, obj_2, and obj_3.

The proposed approach gives better results facing dynamic data because we considered all three major clustering objective functions, i.e., intra-cluster distance, inter-cluster distance, and cluster density in the fitness function which helped to decide the optimal maximum threshold value efficiently. Also, the comparison of dynamic data points with optimal maximum threshold allows the data points to be clustered in most appropriate cluster such that the three major clustering objectives are optimized efficiently, thus the use of multiple objectives makes the clustering strong against changes in the shape, size, dimensionality, and other properties of the clusters. The proposed approach is applied to five benchmark datasets and the experimental findings on these datasets are presented in the subsequent section.

6.3 Experimental Findings

This section briefs about the dataset used for the experimentation, performance evaluation measures, parameter settings for the various parameters, and the results, which are discussed in detail in the following subsections.

6.3.1 Dataset Details

To evaluate the effectiveness of the proposed approach using external evaluation metrics, we used five benchmark datasets taken from the UCI machine learning repository [48] named Iris, Glass, Wine, Sonar, and Parkinsons. The details of benchmark datasets are presented in Table 6.1.

Table 6.1: Benchmark datasets

Dataset	Number of data points	Number of features
Iris	150	8
Glass	214	10
Wine	178	13
Sonar	208	60
Parkinsons	195	22

6.3.2 Performance Evaluation Measures

We used two external evaluation measures named Rand Index and Normalized Mutual Information (NMI) to measure the effectiveness of the proposed approach. These evaluation metrics are discussed in detail in Section 2.5.

6.3.3 Parameter Settings

In this study, the experiments are conducted with $\alpha = 0.3$, $\beta = 0.3$, and $\gamma = 0.4$ for each dataset. To decide the parameter $Maximum_{thershold}$ value, extensive experiments are performed by taking different $Maximum_{thershold}$ values for each dataset, and the optimal $Maximum_{thershold}$ value for each dataset is presented in Table 6.2.

Table 6.2: Parameter settings for the proposed MOB-IC

Dataset name	$Optimal\ Maximum_{thershold}$
Iris	1.40
Glass	1.39
Wine	1.35
Sonar	1.41
Parkinsons	1.46

6.3.4 Experimental Analysis

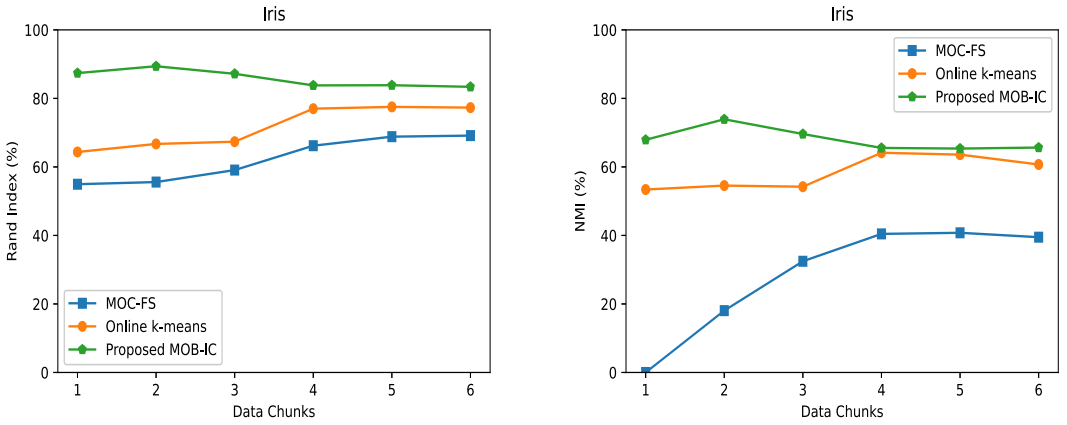
The proposed approach dynamically creates and updates clusters which is extremely beneficial for the constantly evolving datasets, however, to evaluate the performance of the proposed approach in dynamic manner on the fixed-size datasets, first we divided the datasets into 6 chunks of approximate sizes 50%, 10%, 10%, 10%, 10%, 10%, and then provided these chunks in incremental manner as input. The experiments are performed on the datasets listed in Table 6.1 and results are compared with the two state-of-the-art approches: MOC-FS [54] proposed by Sivadi et al. [54], and online K-means proposed by Abernathy et al. [55], in terms of Rand Index and NMI.

Results on Iris dataset:

The proposed MOB-IC is applied to the Iris dataset and experimental findings for the different chunks are presented in Table 6.3. It can be seen that the suggested method performs much better than MOC-FS [54] and online K-means [55]. Also, in the clustering phase for the first data chunk, the suggested method constructs an adequate number of clusters (more than one). Thus the proposed MOB-IC outperforms the MOC-FS method in terms of NMI. The comparison among the proposed method, MOC-FS, and online K-means in terms of Rand Index and NMI are depicted in Figure 6.2(a) and Figure 6.2(b), respectively. It can be seen that the proposed MOB-IC outperforms in comparison to MOC-FS and online K-means for all chunks of the Iris dataset.

Table 6.3: Results on Iris dataset

Data Chunks (DC)	DC Size	Rand Index (%)			NMI (%)		
		MOC-FS	Online K-means	Proposed MOB-IC	MOC-FS	Online K-means	Proposed MOB-IC
1	75	54.95	64.36	**87.39**	0.00	53.40	**67.92**
2	15	55.58	66.71	**89.39**	18.05	54.55	**73.91**
3	15	59.08	67.36	**87.18**	32.46	54.22	**69.61**
4	15	66.19	77.00	**83.80**	40.45	64.13	**65.54**
5	15	68.83	77.53	**83.85**	40.77	63.59	**65.36**
6	15	69.14	77.31	**83.38**	39.49	60.71	**65.65**



(a) Rand Index (b) NMI

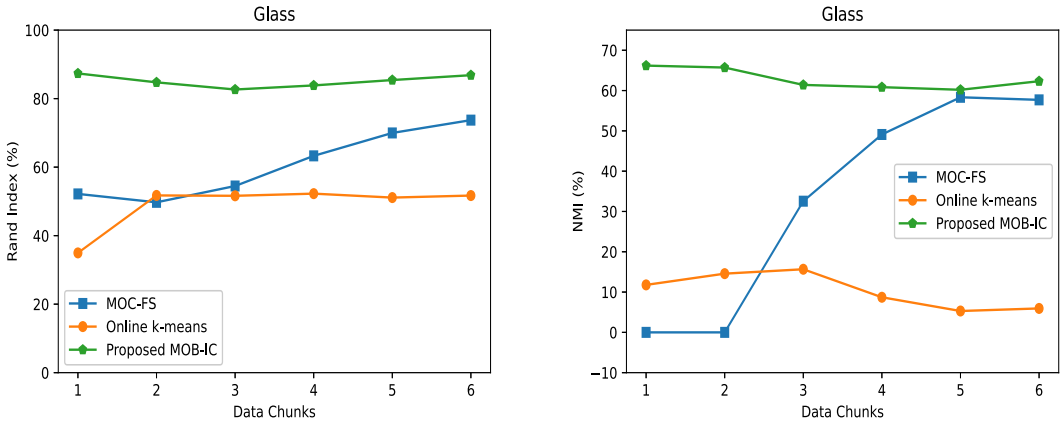
Figure 6.2: Comparison on Iris data

Results on Glass dataset:

The proposed MOB-IC is applied to the Glass dataset, and experimental results for the various chunks are shown in Table 6.4. It is observed that during the clustering of first two chunks, the proposed MOB-IC creates more than one cluster and that is why it is giving better NMI than the MOC-FS [54]. Also for all chunks of data the proposed approach outperforms the MOC-FS [54] and online K-means [55] in terms of Rand Index and NMI. The comparison among proposed approach, MOC-FS, and online K-means in terms of Rand Index and NMI is shown in Figure 6.3(a) and Figure 6.3(b), respectively.

Table 6.4: Results on Glass dataset

Data Chunks (DC)	DC Size	Rand Index (%)			NMI (%)		
		MOC-FS	Online K-means	Proposed MOB-IC	MOC-FS	Online K-means	Proposed MOB-IC
1	114	52.18	34.96	**87.35**	0.00	11.78	**66.19**
2	20	49.73	51.74	**84.75**	0.00	14.57	**65.71**
3	20	54.49	51.63	**82.66**	32.55	15.66	**61.40**
4	20	63.29	52.26	**83.85**	49.10	8.71	**60.84**
5	20	69.98	51.11	**85.42**	58.33	5.30	**60.19**
6	20	73.72	51.70	**86.84**	57.69	5.94	**62.32**



(a) Rand Index (b) NMI

Figure 6.3: Comparison on Glass data

Results on Wine dataset:

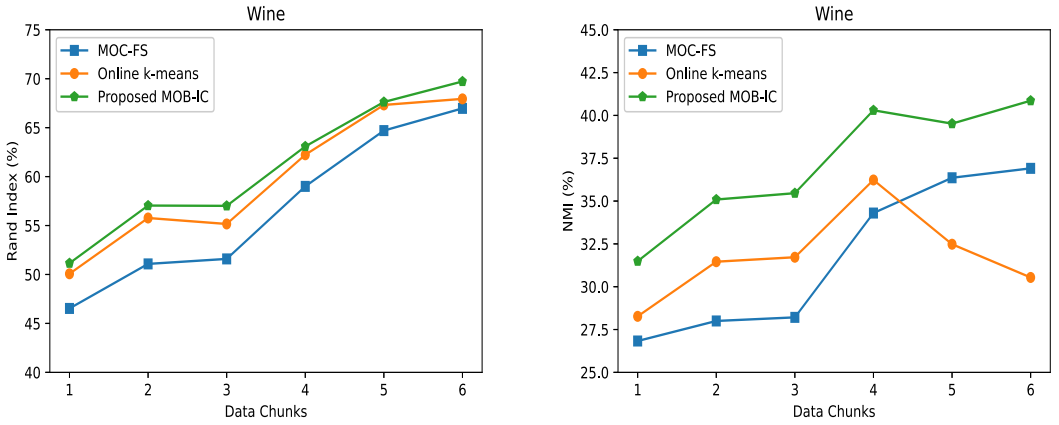
The proposed MOB-IC is applied to the Wine dataset, and experimental results for the various chunks are shown in Table 6.5. It can be seen that proposed approach significantly outperforms in comparison to MOC-FS [54] and online K-means [55] in terms of Rand Index and NMI. The comparison graph of Rand Index and NMI are shown in Figure 6.4(a) and Figure 6.4(b), respectively.

Results on Sonar dataset:

The proposed method is applied to the Sonar dataset, and the experimental outcomes for the various chunks are presented in Table 6.6. The suggested technique greatly outperforms MOC-FS [54] in terms of Rand Index and NMI, as evidenced by the results. On the other hand, in comparison to online K-means [55], the proposed

Table 6.5: Results on Wine dataset

Data Chunks	DC Size	Rand Index (%)			NMI (%)		
		MOC-FS	Online K-means	Proposed MOB-IC	MOC-FS	Online K-means	Proposed MOB-IC
1	89	46.53	50.07	**51.15**	26.83	28.27	**31.49**
2	18	51.08	55.77	**57.04**	28.00	31.46	**35.09**
3	18	51.59	55.16	**57.01**	28.21	31.72	**35.46**
4	18	59.00	62.23	**63.08**	34.30	36.23	**40.30**
5	18	64.70	67.32	**67.62**	36.36	32.48	**39.52**
6	17	66.97	67.94	**69.72**	36.91	30.54	**40.86**



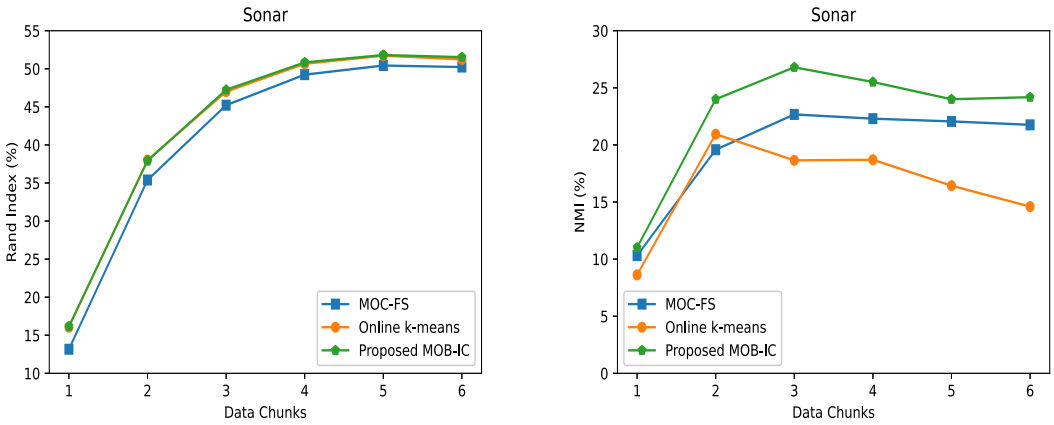
(a) Rand Index (b) NMI

Figure 6.4: Comparison on Wine dataset

MOB-IC performs better for all data chunks except the second data chunk in terms of Rand Index. The Rand Index and NMI comparison graphs are displayed in Figure 6.5(a) and Figure 6.5(b), respectively.

Table 6.6: Results on Sonar dataset

Data Chunks	DC Size	Rand Index (%)			NMI (%)		
		MOC-FS	Online K-means	Proposed MOB-IC	MOC-FS	Online K-means	Proposed MOB-IC
1	104	13.16	16.05	**16.17**	10.31	8.62	**11.01**
2	21	35.39	**38.01**	37.92	19.58	20.94	**24.00**
3	21	45.22	47.00	**47.24**	22.67	18.65	**26.80**
4	21	49.23	50.66	**50.84**	22.30	18.70	**25.51**
5	21	50.43	51.72	**51.81**	22.06	16.43	**24.00**
6	20	50.23	51.24	**51.53**	21.76	14.60	**24.18**



(a) Rand Index (b) NMI

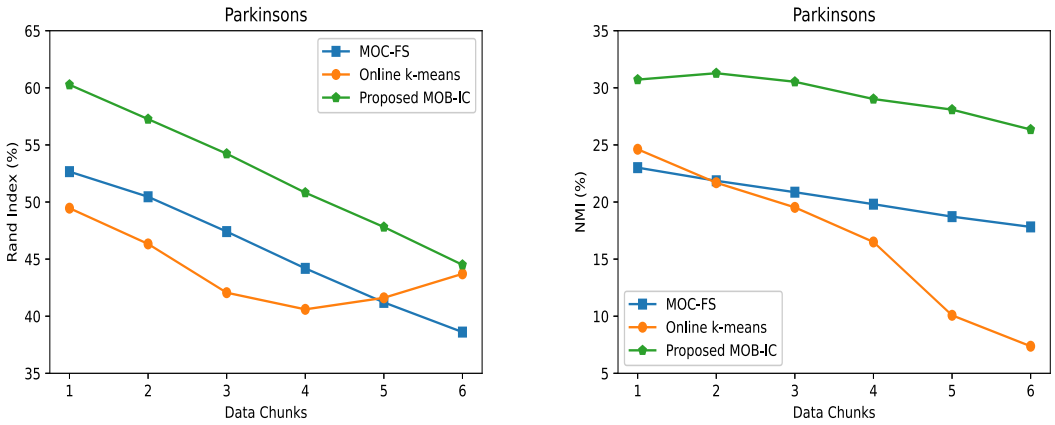
Figure 6.5: Comparison on Sonar dataset

Results on Parkinsons dataset:

The proposed MOB-IC is applied to the Parkinsons dataset and experimental findings for the different chunks are presented in Table 6.7. The results show that the proposed MOB-IC significantly outperforms the MOC-FS [54] and online K-means [55] in terms of the Rand Index and the NMI. Figure 6.6(a) and Figure 6.6(b) depict a comparison of the Rand Index and the NMI, respectively.

Table 6.7: Results on Parkinsons dataset

Data Chunks (DC)	DC Size	Rand Index (%)			NMI (%)		
		MOC-FS	Online K-means	Proposed MOB-IC	MOC-FS	Online K-means	Proposed MOB-IC
1	97	52.66	49.46	**60.27**	23.01	24.62	**30.72**
2	20	50.46	46.33	**57.26**	21.86	21.69	**31.28**
3	20	47.41	42.06	**54.23**	20.86	19.53	**30.53**
4	20	44.19	40.60	**50.82**	19.81	16.50	**29.02**
5	19	41.21	41.61	**47.81**	18.72	10.08	**28.09**
6	19	38.61	43.71	**44.50**	17.82	7.37	**26.35**



(a) Rand Index (b) NMI

Figure 6.6: Comparison on Parkinsons dataset

6.4 Time Complexity Analysis of the Proposed Approach

Since the proposed MOB-IC is using three functions, i.e., *Get_Similar_Cluster*(), *Update_cluster*(), and *Cluster_Center_Calculation*() to perform the clustering of incoming data point as shown in Algorithm 6.1. So, to analyze the time complexity of proposed MOB-IC, consider, that n data objects and k clusters have previously been encountered and generated. Now that the $(n+1)^{th}$ point has been reached, we will evaluate the complexity of each function of the proposed MOB-IC algorithm as follows:

The *Get_Similar_Cluster*() function contains a for loop that runs from 1 to k (total number of clusters formed till now). Therefore, the time complexity of this function is $\mathcal{O}(k)$.

The *Update_cluster*() function has no loops and does only one comparison. However, it calculates the value of the fitness function, which consists of three computations of distinct objectives: i.e., in obj_1 (intra-cluster distance), we calculated the distance of all the data points of the cluster to its cluster center. Therefore, the complexity of obj_1 is $\mathcal{O}(n)$. In the second objective obj_2 (inter-cluster distance), we figured out the

distance of the $(n+1)^{th}$ data point to the data points of all other clusters except the one with which the $(n+1)^{th}$ point had the most similarity. Therefore, the complexity of obj_2 is also $\mathcal{O}(n)$. The obj_3 (cluster density) computes the distance of one point to other point with in one cluster so the time complexity of obj_3 is $\mathcal{O}(n)$. So the overall time complexity of $Update\_cluster()$ is $\mathcal{O}(n+n+n)$ which can be reduced to $\mathcal{O}(n)$.

The $Cluster\_Center\_Calculation()$ function computes the cluster center by taking the mean of all data points, so its time complexity is constant, i.e., $\mathcal{O}(1)$.

So, the overall time complexity of the proposed approach is $\mathcal{O}(k+n+1)$ which can be simply reduced to $\mathcal{O}(k+n)$. Table 6.8 compares the time complexity of the proposed MOB-IC with the MOC-FS and online K-means approaches.

Table 6.8: Comparison of the proposed MOB-IC time complexity with MOC-FS and online K-means

Approach	**Time complexity**
$Get\_Similar\_Cluster()$	$\mathcal{O}(k)$
$Update\_cluster()$	$\mathcal{O}(n)$
$Cluster\_Center\_Calculation()$	$\mathcal{O}(1)$
Overall time complexity of proposed MOB-IC	$\mathcal{O}(k+n)$
MOC-FS approach	$\mathcal{O}(n^2)$
Online K-means	$\mathcal{O}(nk)$

As shown in Table 6.8, MOC-FS approach [54] has a time complexity of $\mathcal{O}(n^2)$ since they used two loops to calculate pair-wise distances to calculate cluster density. The online K-means [55] have the time complexity of $\mathcal{O}(nk)$. Therefore, it can be concluded that the proposed approach takes less time to cluster the dynamic data in comparison to MOC-FS and online K-means.

6.5 Summary

In this chapter, we proposed a novel multi-objective based incremental clustering (MOB-IC) for the analysis of dynamic data. The proposed MOB-IC utilizes the objective weighting concept to optimize the three objective functions of inter-cluster

distance, intra-cluster distance, and cluster density simultaneously, enabling an efficient determination of the optimal maximum threshold value. Furthermore, the comparison mechanism of the proposed MOB-IC evaluates dynamic data points against the optimal maximum threshold value, allowing the data points to be clustered in the most appropriate cluster. This ensures that the three major clustering objectives are efficiently optimized, making the clustering robust to changes in the shape, size, dimensionality, and other properties of the clusters. Additionally, the proposed method outperforms existing state-of-the-art methods in terms of external evaluation measures when tested on five labeled benchmark datasets. Moreover, for the Iris and Glass datasets, the proposed method creates many clusters in the first chunk of data and surpasses the state-of-the-art existing method MOC-FS in terms of cluster quality. Similarly, for high-dimensional Sonar and Parkinsons datasets, proposed MOB-IC also performs well and produces superior results. Furthermore, in the terms of time complexity, the proposed MOB-IC also demonstrates superior performance, requiring less time to cluster dynamic data than existing state-of-the-art techniques. Consequently, we can infer that the proposed MOB-IC performs well in terms of external evaluation measures and time complexity for all types of datasets by simultaneously optimizing the three essential clustering objectives.

The proposed MOB-IC is further evaluated on the unlabeled real-life plant Single Nucleotide Polymorphisms (SNP) datasets of rice crop. A detailed analysis of its performance on these datasets is presented in Chapter 7.

Chapter 7

Investigation of Massive Real-Life Plant SNP and Protein Datasets on Developed Feature Selection and Multi-Objective Based Incremental Clustering

In previous chapters (Chapter 3 and 6), the proposed clustering-based hybrid feature selection approach using Ant Colony Optimization (NCHFS-ACO) and multi-objective based incremental clustering (MOB-IC) strategies were primarily tested on labeled benchmark datasets taken from the UCI machine learning repository [48]. So, in this chapter, we proposed to examine the performance of these approaches on the unlabeled real-life plant Single Nucleotide Polymorphisms (SNP) and protein datasets of rice and soybean crops obtained from Indian Council of Agricultural Research-Indian Institute of Soybean Research (ICAR-IISR), Indore, India. We used a 12-dimensional feature extraction approach to preprocess the SNP sequences and a 60-dimensional feature extraction approach to preprocess the protein sequences. The performance of proposed NCHFS-ACO and MOB-IC is evaluated in terms of internal evaluation indexes.

7.1 Introduction

A fundamental task in bioinformatics is the analysis of plant genome sequences using machine learning techniques such as clustering [155] [156]. The use of clustering aids in the classification of genes, regulatory elements, and functional regions by categorizing similar sequences. However, the high dimensionality of plant genomics data poses a significant challenge to the efficiency of clustering processes. Traditional clustering methods may have trouble finding meaningful patterns in the data because they have to look at a lot of different dimensions. Consequently, the implementation of feature selection techniques becomes necessary to speed up the clustering process and extract relevant features from high-dimensional plant genomics datasets. Feature selection significantly improves the quality of clustering results by selecting only relevant features and discarding redundant ones. This refined clustering not only provides more accurate insights, but it also has enormous potential to advance plant genomics research. By identifying the most influential features, feature selection enables scientists to unravel complex biological mechanisms, identify key genetic traits, and accelerate the development of innovative solutions to agricultural challenges. In essence, the strategic selection of features increases the impact of clustering techniques, making them invaluable tools for exploring and understanding plant genomes.

In addition, another challenge in plant genomics area is handling of dynamic data. Plant genomics research produces a large amount of dynamic data as a result of environmental fluctuations, developmental stages, and genetic variations. This dynamic data captures the complex interactions between genes, traits, and environmental factors over time. Moreover, many bioinformatics applications [163] generate dynamic genomics data streams. To extract knowledge from this vast volume of real-time data, it needs to be analyzed using a variety of data mining methodologies, including clustering for pattern recognition. Analyzing this vast amount of dynamic data allows researchers to gain valuable insights into plant biology, adaptation mechanisms, and the effects of genetic modifications, thereby promoting advances in agriculture and crop breeding.

Hence, to address the aforementioned challenges, in this chapter, we examined the performance of our proposed NCHFS-ACO and MOB-IC on various real-life plant genome and protein datasets. The detailed descriptions of NCHFS-ACO and MOB-IC are presented in Chapters 3 and 6, respectively. The subsequent section provides details of the real-life plant SNP and protein datasets used for experimentation.

7.2 Dataset Details

We collected two types of real-life plant data: SNP sequences and protein sequences, to evaluate the performance of proposed approaches. A SNP [147] denotes a variation (deletion/addition) in a single nucleotide at a specific position in the genome, composed of the four nucleotides, adenine (A), cytosine (C), guanine (G), and thymine (T). It serves as a tool for investigating associations between genes and diseases or traits [164]. Conversely, a protein sequence [151] consists of twenty amino acids: Alanine (A), Cysteine (C), Aspartic acid (D), Glutamic acid (E), Phenylalanine (F), Glycine (G), Histidine (H), Isoleucine (I), Lysine (K), Leucine (L), Methionine (M), Asparagine (N), Proline (P), Glutamine (Q), Arginine (R), Serine (S), Threonine (T), Valine(V), Tryptophan (W), and Tyrosine (Y). The amino acids inside a sequence can be joined together in any arrangement, and the protein sequences can vary in length. The process of clustering protein sequences is crucial for the identification of functional linkages, the grouping of proteins with comparable characteristics, and the understanding of evolutionary trends.

In SNP category, We collected three different SNP datasets of rice plant namely MAGIC-rice [149], SNP-seek rice [148], and 248Entries rice [150]. For the protein category, we collected a huge protein sequence dataset of soybean plant namely Glycine Soja accession W05 [146]. The detailed description of SNP and protein datasets are provided in Sections 2.6.2 and 2.6.3.

To apply the above proposed approaches on the real-life plant SNP and protein data, first we need to apply the preprocessing step on this data. This preprocessing enables the transformation of biological sequences into numerical data, hence facil-

itating the analysis of such data by NCHFS-ACO and MOB-IC approaches. The preprocessing steps applied on these datasets are discussed subsequently.

7.3 Preprocessing of Plant SNP and Protein Datasets

We used an existing 12-dimensional feature extraction approach (12d-FET) for preprocessing SNP datasets and a 60-dimensional feature extraction approach (60d-FET) for processing protein datasets. The preprocessing steps for the SNP datasets and protein dataset is presented in Section 7.3.1 and Section 7.3.2, respectively.

7.3.1 Preprocessing of SNP Datasets using 12d-FET

During the preprocessing of SNP datasets, we extracted 12 numerical features from each SNP sequence using the 12d-FET approach proposed by Preeti et al. [50]. The procedure for extracting numerical features is as follows:

Table 7.1: Example of SNP sequence

	1	2	3	4	5	6	7	8	9
Seq 1	G	A	A	T	G	C	T	G	G

Step 1:

Count the length of each nucleotide ($l_{nucleotide}$) present in the SNP sequence. For example, we took one SNP sequence given in Table 7.1. In this sequence, total count of A, T, G, and C are 2, 2, 4, and 1, respectively. So the value of l_A, l_T, l_G, and l_C are 2, 2, 4, and 1, respectively.

Step 2:

Compute the sum of the distances of each nucleotide ($s_{nucleotide}$) to the first nucleotide. In this feature, we add the position values of each nucleotide present in the sequence. In seq 1, nucleotide A appears at positions 2 and 3, so the value of $(s_A) = 2 + 3 = 5$. Similarly, we calculated the value of s_T, s_G, s_C.

Step 3:

In this step, for each nucleotide, calculate the variance of distance ($d_{nucleotide}$). To extract this feature, firstly $\mu_{nucleotide}$ was computed using Eq. (7.1). For example, in seq 1 the computation of (d_G) is shown in Eq. (7.2).

$$\mu_{nucleotide} = s_{nucleotide}/l_{nucleotide} \tag{7.1}$$

So, the $\mu_G = s_G/l_G = 23/4 = 5.75$

$$\begin{aligned} d_G = [(1-5.75)^2 + (5-5.75)^2 \\ + (8-5.75)^2 + (9-5.75)^2]/4 \end{aligned} \tag{7.2}$$

So, $d_G = 9.67$

Similarly, the values of d_A d_C d_T were calculated. So after collecting the features from these three steps, the 12 dimensional feature vector is represented by

$(l_A, s_A, d_A, l_G, s_G, d_G, l_T, s_T, d_T, l_C, s_C, d_C)$

$(2, 5, 0.25, 4, 23, 9.67, 2, 11, 2.25, 1, 6, 0)$.

7.3.2 Preprocessing of Protein Datasets using 60d-FET

For the preprocessing of the protein sequence, we followed the 60d-FET approach proposed by Preeti et al. [165]. In protein sequence, there are twenty amino acids, so, for each amino acid, we calculated 3 features. So there will be 60 features for each protein sequence. The detailed description of 60d-FET is as follows:

Step 1:

In this step, for each amino acid z, the count of each amino acid (len_z) is calculated. There are twenty amino acids present in protein sequence, i.e, $\sum$={A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y}. The counts of these amino acids is defined as len_A, len_C, len_D, len_E, len_F, len_G, len_H, len_I, len_K, len_L, len_M, len_N, len_P, len_Q, len_R, len_S, len_T, len_V, len_W, len_Y.

Step 2:

In this step, for each amino acid z, the total distance from the first amino acid TD_z is calculated using Eq. (7.3).

$$TD_n = \sum_{i=1}^{len_z} d_i, \tag{7.3}$$

where d_i is the distance of i^{th} amino acid from the first amino acid. After computation the total distances of all amino acids is defined as TD_A, TD_C, TD_D, TD_E, TD_F, TD_G, TD_H, TD_I, TD_K, TD_L, TD_M, TD_N, TD_P, TD_Q, TD_R, TD_S, TD_T, TD_V, TD_W, TD_Y.

Step 3:

In this step, features based on distribution are computed using the above two types of features, i.e., count and total distance. In this feature, for each amino acid z, the distribution of each nucleotide (dis_z) is calculated in two steps. In the first step, the average of total distance is calculated using Eq. (7.4). Then, distribution is computed using Eq. (7.5).

$$Avgd_z = TD_z/len_z, \tag{7.4}$$

$$dis_z = \sum_{i=1}^{len_z} \frac{(d_i - Avgd_z)^2}{len_z}. \tag{7.5}$$

After computation the distributions of all amino acids is defined as dis_A, dis_C, dis_D, dis_E, dis_F, dis_G, dis_H, dis_I, dis_K, dis_L, dis_M, dis_N, dis_P, dis_Q, dis_R, dis_S, dis_T, dis_V, dis_W, dis_Y. So, after collecting the features from these three steps, the 60 dimensional feature vector is represented by $\langle \text{len}_A$, len_C, len_D, len_E, len_F, len_G, len_H, len_I, len_K, len_L, len_M, len_N, len_P, len_Q, len_R, len_S, len_T, len_V, len_W, len_Y, TD_A, TD_C, TD_D, TD_E, TD_F, TD_G, TD_H, TD_I, TD_K, TD_L, TD_M, TD_N, TD_P, TD_Q, TD_R, TD_S, TD_T, TD_V, TD_W, TD_Y, dis_A, dis_C, dis_D, dis_E, dis_F, dis_G, dis_H, dis_I, dis_K, dis_L, dis_M, dis_N, dis_P, dis_Q, dis_R, dis_S, dis_T, dis_V, dis_W, $dis_Y\rangle$.

After preprocessing, the details of SNP and protein datasets are shown in Table 7.2.

The experimental analysis of proposed NCHFS-ACO and MOB-IC on these processed datasets are presented subsequently.

Table 7.2: Real-life plant SNP and protein datasets

Dataset name	Count of instances	Count of features	Data type
MAGIC-rice	16932	12	SNP
SNP-seek rice	252	12	SNP
248Entries rice	248	12	SNP
W05	66622	60	Protein sequence

7.4 Experimental Analysis of Proposed Feature Selection on SNP and Protein Datasets

The proposed NCHFS-ACO (discussed in Chapter 3) is applied on the real-life plant SNP and protein datasets listed in Table 7.2, and the NCHFS-ACO approach is compared to a hybrid feature selection approach developed by Solorio et al. [1]. Since these SNP datasets are unlabeled, first of all the number of clusters are identified by K-means clustering. To identify the number of clusters, K-means clustering is performed by taking various k values and results are presented only on a few best-performing k values. Due to the unavailability of class labels, results are quantified in terms of Silhouette Index (SI) and Silhouette Visualizer only. The detailed descriptions of SI and Silhouette Visualizer are presented in Sections 2.5 and 3.3.2, respectively.

Results on MAGIC-rice:

Table 7.3 shows the SI values corresponding to a few best-performing clusters (k) using K-means clustering. It can be seen that $k = 3$ is giving the highest SI among various k values. So further analysis is performed by taking $k = 3$.

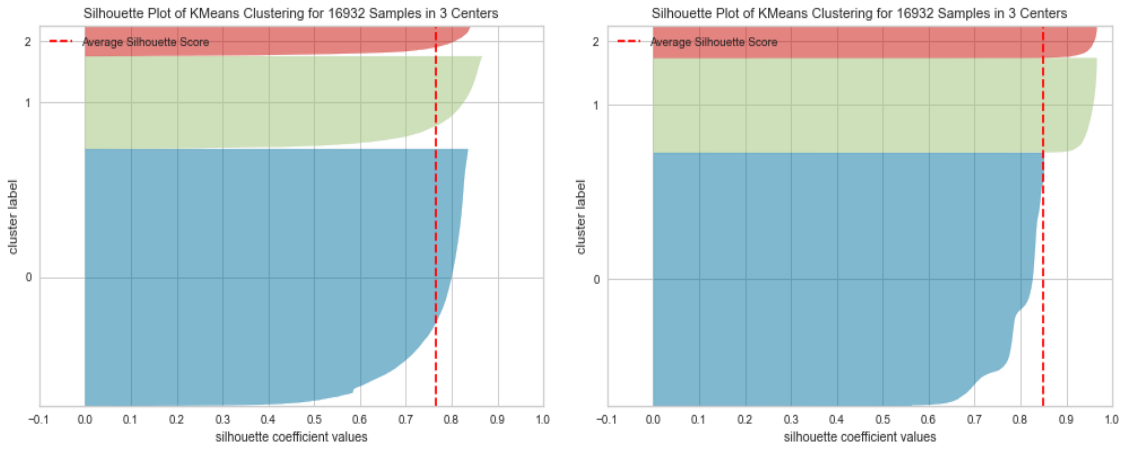
Table 7.4 shows the results of the NCHFS-ACO on the MAGIC-rice dataset for various n_{max} values by taking $k = 3$. When one feature is selected, the NCHFS-ACO gives a better SI than the Solorio et al. [1] approach. The NCHFS-ACO also

Table 7.3: Few best performing k values on MAGIC-rice

No. of clusters(k)	SI
2	0.7333
3	0.7644
4	0.6273

Table 7.4: Results on MAGIC-rice

Technique used	No. of features selected	SI
Solorio et al.	4	0.7644
NCHFS-ACO	**1**	**0.8487**
NCHFS-ACO	2	0.8411
NCHFS-ACO	4	0.8329



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 7.1: Results on MAGIC-rice

performs better than Solorio et al. [1] in the case of 4 features by selecting another set of 4 features. The Silhouette Visualizer obtained from Solorio et al. [1] and the NCHFS-ACO approach is presented in Figure 7.1(a) and Figure 7.1(b) to observe the clustering results. It can be observed from the Silhouette Visualizer obtained from NCHFS-ACO that all clusters are having better average SI value than the Silhouette Visualizer obtained from Solorio et al. [1] approach.

Results on SNP-seek rice:

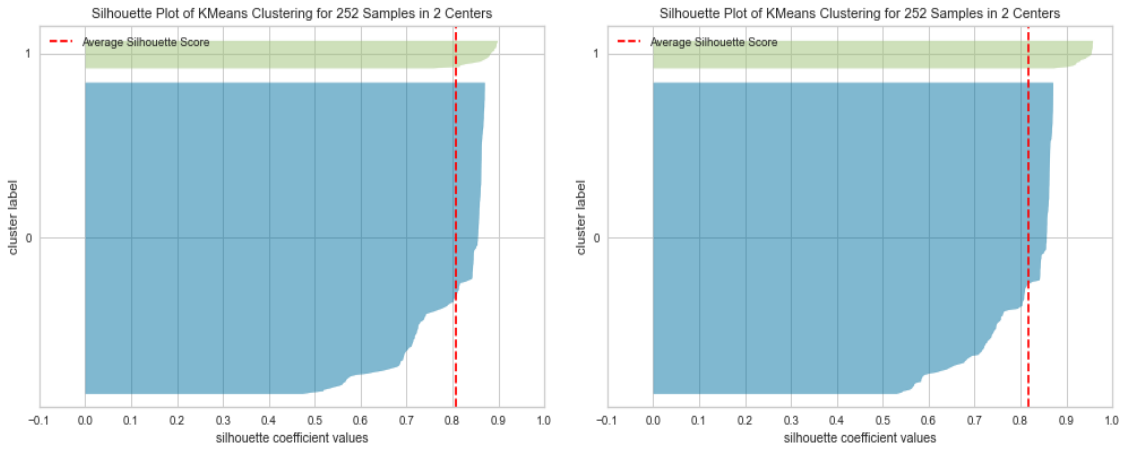
Table 7.5 shows the SI values corresponding to a few best-performing clusters (k) using K-means clustering. It can be seen that $k = 2$ is giving the highest SI among various k values. So further analysis is performed by taking $k = 2$.

Table 7.5: Few best performing k values on SNP-seek rice

No. of clusters(k)	SI
2	0.8069
3	0.7296
4	0.6252

Table 7.6: Results on SNP-seek rice

Technique used	No. of features selected	SI
Solorio et al.	6	0.8069
NCHFS-ACO	**1**	**0.8166**
NCHFS-ACO	2	0.8458
NCHFS-ACO	6	0.8069



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 7.2: Results on SNP-seek rice

Table 7.6 shows the results of the NCHFS-ACO on the SNP-seek rice dataset for various n_{max} values by taking $k = 2$. When one feature is selected, the NCHFS-ACO

gives better SI than the Solorio et al. [1] approach. The Silhouette Visualizer obtained from Solorio et al. [1] and NCHFS-ACO approach is presented in Figure 7.2(a) and Figure 7.2(b) to observe the clustering results. It can be seen from the Silhouette Visualizer obtained from the NCHFS-ACO that all clusters are having better average SI value than the Silhouette Visualizer obtained from Solorio et al. [1] approach.

Results on 248Entries rice:

Table 7.7 shows the SI values corresponding to a few best-performing k values using K-means clustering. It can be seen that $k = 2$ is giving the highest SI among various k values. So further analysis is performed by taking $k = 2$.

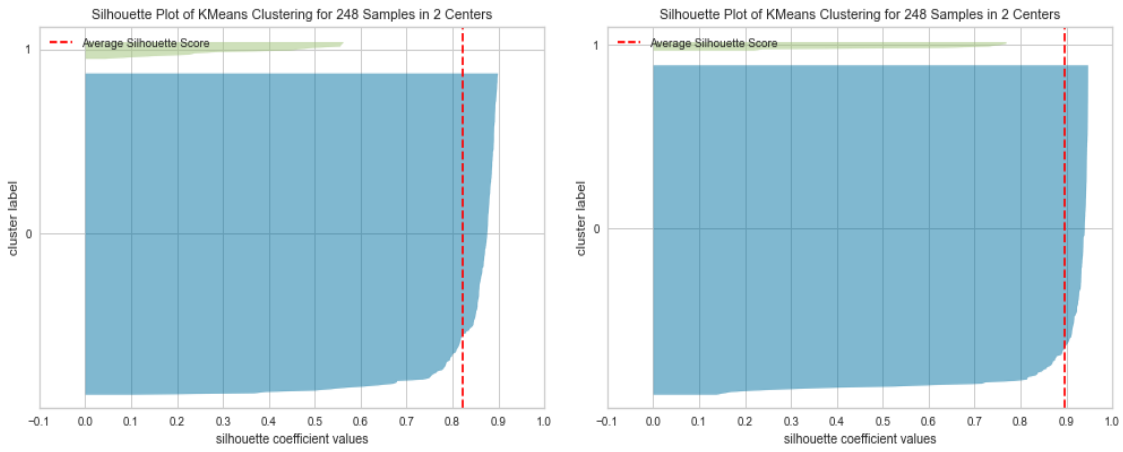
Table 7.7: Few best performing k values on 248Entries rice

No. of clusters(k)	SI
2	0.8208
3	0.6967
4	0.4411

Table 7.8: Results on 248Entries rice

Technique used	No. of features selected	SI
Solorio et al.	8	0.8208
NCHFS-ACO	**1**	**0.8957**
NCHFS-ACO	2	0.8954
NCHFS-ACO	8	0.8319

Table 7.8 shows the results of the NCHFS-ACO on the SNP-seek rice dataset for various n_{max} values by taking $k = 2$. When one feature is selected, the NCHFS-ACO gives better SI than the Solorio et al. [1] approach. The NCHFS-ACO also performs better than Solorio et al. [1] in the case of 8 features by selecting another set of 8 features. It can be observed from the Silhouette Visualizer presented in Figure 7.3(b) for the NCHFS-ACO approach, that all clusters are having better average SI value than the Silhouette Visualizer shown in Figure 7.3(a) obtained from Solorio et al. [1] approach.



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 7.3: Results on 248Entries rice

Results on W05 protein dataset:

Table 7.9 shows the SI values corresponding to a few best-performing k values by K-means clustering. It can be seen that $k = 2$ is giving the highest SI among various k values. So further analysis is performed by taking $k = 2$.

Table 7.9: Few best performing k on W05 Protein dataset

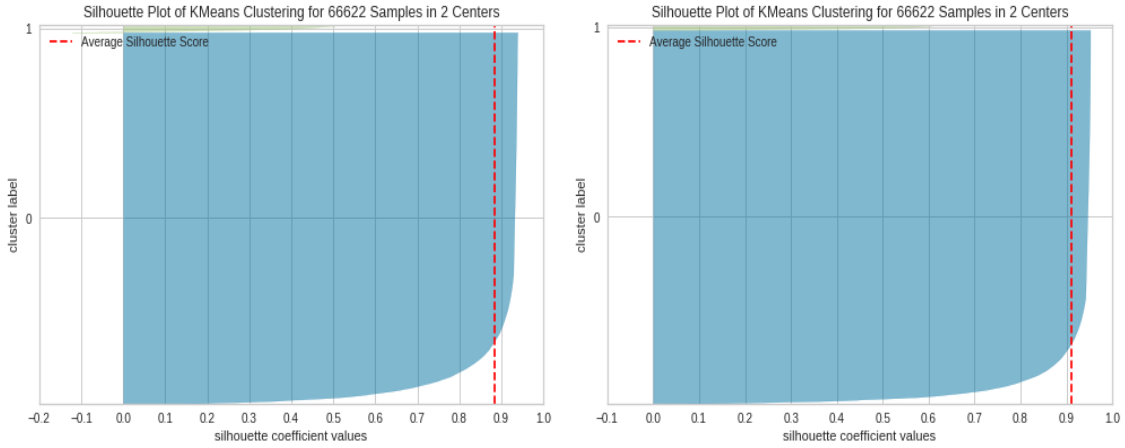
No. of clusters(k)	SI
2	0.8847
3	0.7364
4	0.7230

Table 7.10: Results on W05 Protein dataset

Technique used	No. of features selected	SI
Solorio et al.	29	0.8847
NCHFS-ACO	**1**	**0.9109**
NCHFS-ACO	10	0.8956
NCHFS-ACO	29	0.8848

Table 7.10 shows the results of the NCHFS-ACO on the W05 Protein dataset for various n_{max} values by taking $k = 2$. When one feature is selected, the NCHFS-ACO

gives a better SI than the Solorio et al. [1] approach. The NCHFS-ACO also performs slightly better than Solorio et al. [1] in the case of 29 features by selecting another set of 29 features. The Silhouette Visualizer obtained from Solorio et al. [1] and NCHFS-ACO approach is presented in Figure 7.4(a) and Figure 7.4(b) to observe the clustering results. It can be observed from the Silhouette Visualizer obtained from NCHFS-ACO that all clusters are having better average SI values than the Silhouette Visualizer obtained from Solorio et al. [1] approach.



(a) Solorio et al. [1] approach (b) Proposed NCHFS-ACO

Figure 7.4: Results on W05 Protein dataset

Comparative performance analysis on SNP datasets:

The comparison between Solorio et.al. [1] and the NCHFS-ACO in terms of the SI is presented in Figure 7.5. From the figure, it can be seen that the NCHFS-ACO provides significantly higher SI values across all Real-life SNP datasets. We performed another comparison in terms of SI, by taking the same number of features as computed by Solorio et al. [1] approach and the corresponding results are highlighted in saffron color. Figure 7.5 also indicates that with the same number of features, the NCHFS-ACO provides superior SI values for MAGIC-rice and 248Entries datasets but it achieves the same SI value in the case of the SNP-seek rice dataset.

Comparative performance analysis on protein dataset:

The comparison between Solorio et.al. [1] and the NCHFS-ACO in terms of the SI is

shown in Figure 7.6. From the figure, it can be seen that the NCHFS-ACO provides significantly higher SI values than the Solorio et al. [1] approach. We performed another comparison by taking the same number of features as computed by Solorio et al. [1] approach and the corresponding results are highlighted in saffron color. Figure 7.6 also indicates that with the same number of features, the NCHFS-ACO provides a slightly higher SI value.

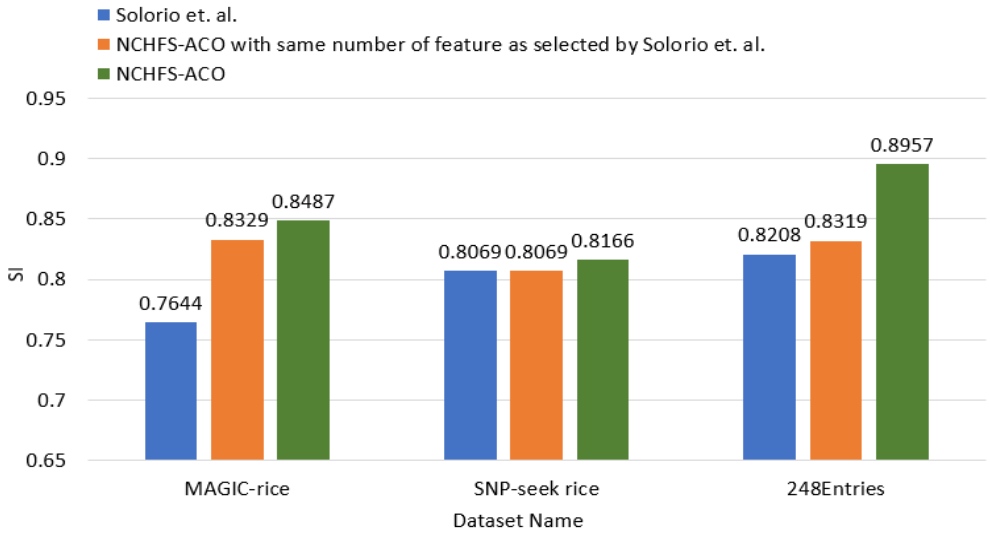


Figure 7.5: Comparison between Solorio et.al. [1] and NCHFS-ACO in SI on SNP datasets

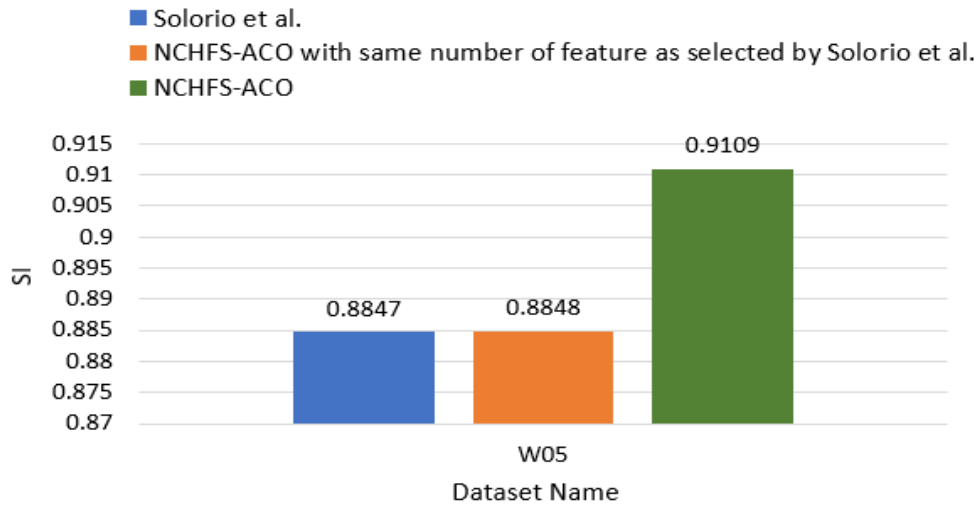


Figure 7.6: Comparison between Solorio et.al. [1] and NCHFS-ACO in SI on protein dataset

7.5 Experimental Analysis of Proposed Multi-Objective Based Incremental Clustering on SNP Datasets

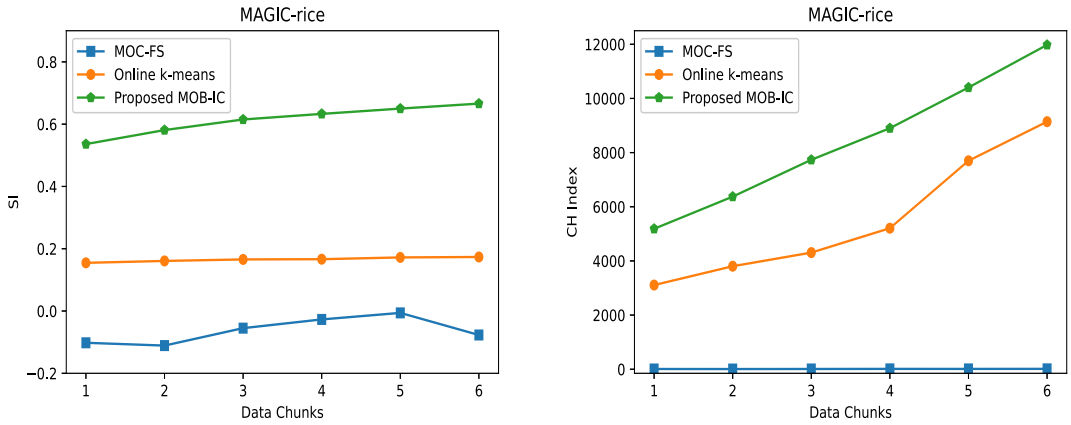
We conducted experiments using the SNP datasets listed in Table 7.2 to assess the effectiveness of the proposed MOB-IC method (detailed in Chapter 6). To evaluate the performance, we computed results in terms of SI and the Calinski-Harabasz Index (CH Index). These metrics are discussed in detail in Section 2.5. In addition, we compared the results of the proposed MOB-IC with the MOC-FS [54] and online K-means [55]. The results on the SNP datasets are discussed subsequently.

Results on MAGIC-rice:

The experimental findings for the MAGIC-rice dataset is presented in Table 7.11. It can be noticed that all SI values for the suggested MOB-IC are positive and higher than the MOC-FS [54] and online K-means [55], indicating that all data points are more closely related with their own cluster than with the cluster that is closest to them. In the case of CH Index, the proposed approach gives significantly higher values than MOC-FS [54] and online K-means [55], due to this the proposed approach performs superior than MOC-FS [54] and online K-means [55]. Figure 7.7(a) and Figure 7.7(b) compare the SI and CH Index of the proposed approach, MOC-FS, and online K-means.

Table 7.11: Results on MAGIC-rice dataset

Data Chunks (DC)	DC Size	SI			CH Index		
		MOC-FS	Online K-means	Proposed MOB-IC	MOC-FS	Online K-means	Proposed MOB-IC
1	8466	-0.102	0.1547	**0.536**	9.3	3108	**5186**
2	1693	-0.111	0.1608	**0.581**	7.1	3804	**6374**
3	1693	-0.055	0.1657	**0.615**	9.7	4307	**7734**
4	1693	-0.027	0.1664	**0.633**	12.7	5208	**8901**
5	1693	-0.006	0.1721	**0.650**	11.9	7698	**10404**
6	1694	-0.077	0.1735	**0.666**	13.9	9142	**11978**



(a) SI (b) CH Index

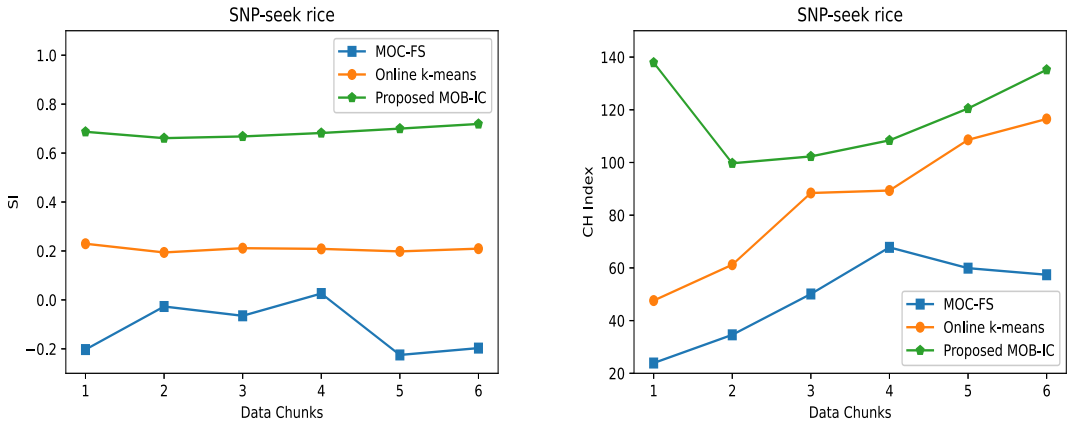
Figure 7.7: Comparison on MAGIC-rice

Results on SNP-seek rice:

The experimental findings for the SNP-seek rice dataset are presented in Table 7.12. It can be observed that the proposed MOB-IC has all positive SI values, which shows that all data points are tightly associated with their own cluster in comparison to the nearest cluster. The proposed approach works better than the MOC-FS [54] and online K-means [55] for all data chunks in both the SI and CH Index. In Figure 7.8(a) and Figure 7.8(b), the SI and CH Index of the proposed MOB-IC, MOC-FS [54], and online K-means [55] are compared.

Table 7.12: Results on SNP-seek rice dataset

Data Chunks (DC)	DC Size	SI			CH Index		
		MOC-FS	Online K-means	Proposed MOB-IC	MOC-FS	Online K-means	Proposed MOB-IC
1	126	-0.203	0.2295	**0.687**	23.9	47.60	**137.9**
2	25	-0.027	0.1939	**0.661**	34.6	61.20	**99.7**
3	25	-0.065	0.2113	**0.668**	50.1	88.42	**102.3**
4	25	0.026	0.2085	**0.682**	67.8	89.36	**108.4**
5	25	-0.225	0.1982	**0.700**	59.9	108.6	**120.5**
6	26	-0.197	0.2094	**0.719**	57.4	116.52	**135.2**



(a) SI (b) CH Index

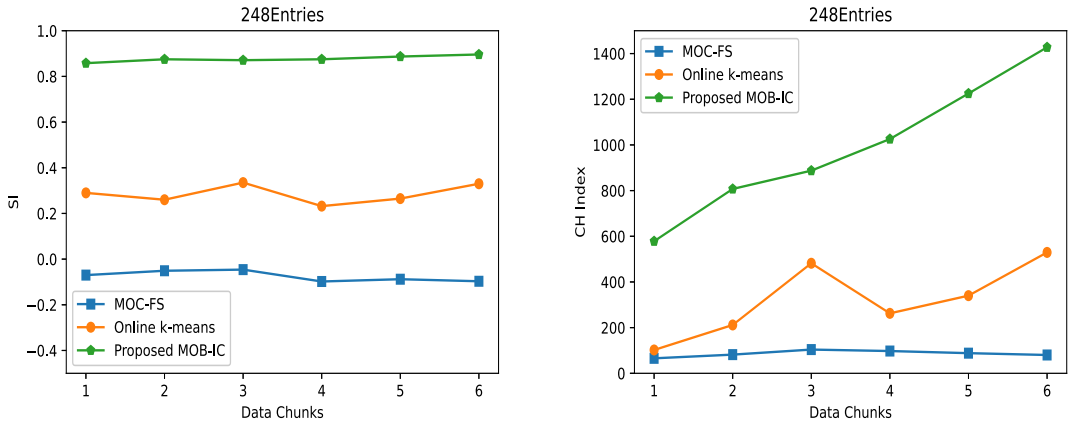
Figure 7.8: Comparison on SNP-seek rice

Results on 248Entries rice:

The experimental results for the 248Entries dataset are shown in Table 7.13. It can be noted that all SI values for the proposed method are positive, indicating that all data points are firmly related with their own cluster in comparison to the next cluster. In case of both SI and CH Index, the performance of the suggested MOB-IC is also superior to that of the MOC-FS [54] and online K-means [55] for all data chunks. Figure 7.9(a) and Figure 7.9(b) provide a comparison of the proposed technique, MOC-FS, and online K-means in terms of SI and CH Index, respectively.

Table 7.13: Results on 248Entries rice dataset

Data Chunks (DC)	DC Size	SI			CH Index		
		MOC-FS	Online K-means	Proposed MOB-IC	MOC-FS	Online K-means	Proposed MOB-IC
1	124	-0.070	0.290	**0.858**	65.6	102.12	**578.3**
2	25	-0.051	0.260	**0.875**	81.8	211.63	**807.2**
3	25	-0.046	0.335	**0.871**	103.9	481.83	**887.4**
4	25	-0.098	0.232	**0.875**	97.5	262.53	**1025.8**
5	25	-0.088	0.265	**0.887**	88.2	340.035	**1225.1**
6	24	-0.097	0.330	**0.896**	80.2	529.33	**1427.5**



(a) SI (b) CH Index

Figure 7.9: Comparison on 248Entries rice

7.6 Summary

In this chapter, we examined the performance of our proposed feature selection and multi-objective based clustering on real-life plant Single Nucleotide Polymorphisms (SNP) and protein sequences. We applied a 12-dimensional feature extraction approach to preprocess the SNP data and a 60-dimensional approach for the protein data. Subsequently, we utilized the processed data as input for these methods. Our proposed feature selection yielded significant improvements in SI across all real-life SNP datasets. Notably, it also outperformed other state-of-the-art method, even on large protein datasets. Furthermore, our incremental clustering method demonstrated strong performance in terms of SI and CH Index across all SNP datasets. Particularly, it consistently produced appropriate clustering with increased and positive SI values, surpassing the cluster quality of the MOC-FS method. While we previously evaluated the efficacy of our algorithms on benchmark datasets, this chapter focuses on their application to real-life plant SNP and protein datasets. The results revealed that our proposed approaches performed exceptionally well in selecting appropriate features and clustering incremental data compared to other state-of-the-art methods. Therefore, we can conclude that our proposed approaches are general purpose and can be effectively applied to any data.

Chapter 8

Conclusions and Future Work

This thesis focuses on the development of novel feature selection, scalable alignment-free feature extraction, and incremental clustering algorithms to address various plant genomics issues, such as the high dimensionality of genome data, the time-consuming nature of alignment-based approaches, and the handling of real-time dynamic data, among others. In particular, we created a novel clustering-based hybrid feature selection strategy, two alignment-free scalable feature extraction approaches using Apache Spark, and an incremental clustering based on multiple objectives. To deal with the high dimensionality, initially, we devised a clustering-based hybrid feature selection strategy based on Ant colony Optimization (NCHFS-ACO). The NCHFS-ACO employs K-means clustering to assign the fitness of features in terms of Silhouette Index (SI) along with the Laplacian score during the feature selection process, as well as to assess the performance of the proposed feature selection algorithm. The NCHFS-ACO is tested on a variety of benchmark datasets and compared against the state-of-the-art technique. The results show that our proposed method outperforms the existing method in terms of the SI and the Jaccard Index (JI). Then, we designed two alignment-free, Apache Spark based scalable feature extraction approaches for the genome dataset to overcome the time-consuming nature of alignment-based approaches. First, we proposed a 14-dimensional feature extraction approach for obtaining features from genome sequences. This method extracts the most important features based on the arrangement of nucleotides. These extracted features are being

clustered using K-means and Fuzzy c-means clustering algorithms, and the results are evaluated in terms of SI and Davies-Bouldin Index (DBI) for the unlabeled datasets and in terms of JI and Rand Index (RI) for the labeled datasets. However, this method does not extract features based on nucleotide classification using chemical properties. So, to further improve the performance, we proposed another alignment-free, scalable feature extraction method that extracts the significantly important features based on the classification of nucleotides using their chemical properties in terms of entropy and the length of the sequence. Further, the clustering of these extracted features is performed using K-means clustering, and the results are evaluated in terms of SI and DBI. Also, the developed scalable feature extraction techniques are generalized approaches that can be applied to any SNP (nucleotide form) and DNA sequences. After that, to handle the real-time dynamic data, we developed a novel incremental clustering method based on multiple objectives. The proposed incremental clustering simultaneously optimized the three objective functions named inter-cluster distance, intra-cluster distance, and cluster density using the objective weighting concept for the dynamic data analysis and overcome the deficiency of disregarding extra crucial cluster attributes including shape, size, and dimensionality. In addition, the proposed method beats the existing state-of-the-art methods in terms of external evaluation measures when tested on five labeled benchmark datasets.

Further, we tested the proposed feature selection and incremental clustering on the unlabeled real-life SNP and protein datasets. We extracted features from these sequences, and those feature vectors were used as inputs for feature selection and incremental clustering algorithms. The results show that the proposed feature selection and incremental clustering approaches performed well in comparison to other state-of-the-art approaches. The proposed feature selection and incremental clustering approaches are general-purpose and can be applied to any problem. A brief summary of our research achievements are as follows:

8.1 Summary of Research Achievements

The research objectives have been successfully fulfilled and summarized as follows:

(i) **A Novel Clustering-Based Hybrid Feature Selection Approach Using Ant Colony Optimization:**

We proposed a novel clustering-based hybrid feature selection method named "NCHFS-ACO" to tackle the high dimensionality issue in datasets. This approach draws inspiration from the social behavior of ants to select the most appropriate features while eliminating redundant and irrelevant ones that impede model construction. Unlike conventional methods that select features sequentially, NCHFS-ACO adopts a random feature selection order to avoid becoming trapped in local optima. Moreover, it employs a hybrid approach to evaluate feature qualities using K-means clustering in terms of SI and Laplacian score. Additionally, NCHFS-ACO mimics the behavior of Temnothorax Albipennis ant species, employing a tandem run strategy to identify the most promising features from a leader subset, ensuring their retention throughout computation and yielding a global optimal solution. Since the proposed approach is clustering-based, it is applicable to both labeled and unlabeled data. Consequently, when evaluated on ten high-dimensional benchmark datasets, our proposed method outperforms an existing method in terms of SI and JI.

(ii) **A Novel Apache Spark Based 14-Dimensional Scalable Feature Extraction Approach for the Clustering of Genomics Data:**

To address the time-consuming nature of alignment-based approaches, we introduced an alignment-free Apache Spark-based scalable 14-dimensional feature extraction technique (14d-SFET) for converting genomics data into numeric feature vectors. Our proposed 14d-SFET method captures five distinct types of features: sequence length, nucleotide frequencies, modified total distance, modified distribution, and entropy, culminating in a 14-dimensional feature vector. Notably, our approach employs a novel power method that rectifies the issue of

extracting identical features for dissimilar sequences. Moreover, we utilized the extracted feature vectors as input for K-means and Fuzzy c-means clustering algorithms to cluster vast genomic datasets. To evaluate our approach, we conducted experiments using seven unlabeled real-life plant genomics datasets of the soybean crop collected from Indian Council of Agricultural Research-Indian Institute of Soybean Research (ICAR-IISR), Indore, India and two labeled benchmark genome sequence datasets collected from UCI machine learning repository [48]. The experimental results demonstrate the superior performance of our proposed 14d-SFET method in terms of both internal and external evaluation measures when compared to existing methods.

(iii) **A Novel 13-Dimensional Alignment-Free Scalable Feature Extraction Method for Genomic Data Clustering:**

The preceding 14d-SFET approach solely extracted features based on the arrangement of nucleotides. However, this method failed to incorporate features derived from the chemical properties of nucleotides, which could have offered additional insights into genome functionality. Thus, we proposed another alignment-free, scalable 13-dimensional feature extraction approach (13dim-SFA) utilizing Apache Spark. The proposed 13dim-SFA method discerned 13 crucial features based on both the length and the chemical properties of nucleotides. Initially, to capture features based on chemical properties, 13dim-SFA categorized genome sequences into three groups: pyrimidine or purine, keto or amino, and strong hydrogen bond or weak hydrogen bond groups. Subsequently, it extracted position and frequency based features in terms of entropy values from these categorized sequences. The resulting 13-dimensional feature vector served as input for K-means clustering to facilitate genome sequence clustering, and the clustering results are measured in terms of SI and DBI. Experimental findings indicate that the proposed 13dim-SFA outperforms other state-of-the-art approaches, demonstrating its efficacy in genome sequence clustering.

(iv) **A Novel Multi-Objective Based Incremental Clustering Method for**

Dynamic Data Analysis

The continuous generation of extensive data from various sources emphasizes the necessity for real-time analytical techniques. Hence, to address real-time dynamic data, we introduced a novel multi-objective based incremental clustering (MOB-IC) approach. The Proposed MOB-IC efficiently clusters incremental data points by simultaneously optimizing three major clustering objectives: intra-cluster distance, inter-cluster distance, and cluster density. This approach utilizes the concept of objective weighting, assigning approximate equal weight to each objective to ensure an unbiased solution. An additional advantage is its real-time cluster generation and updating, eliminating the need for predetermined clustering parameters like the number of clusters. To evaluate the proposed approach's dynamic performance on fixed-size datasets, we divided the datasets into six chunks of approximate sizes 50%, 10%, 10%, 10%, 10%, 10%. These chunks were provided incrementally as input, and results were measured at the end of each chunk in terms of RI and Normalized Mutual Information (NMI). The proposed method is evaluated on five benchmark datasets, and the experimental results indicate that it surpasses other state-of-the-art approaches. In terms of time complexity, it also exhibits better performance compared to other state-of-the-art methods.

(v) **Investigation of Massive Real-Life Plant SNP and Protein Datasets on Developed Feature Selection and Multi-objective Based Incremental Clustering**

The proposed NCHFS-ACO and MOB-IC approaches have been tested with the labeled benchmark datasets. However, to evaluate the performance of these methods on the unlabeled real-life plant SNP and protein datasets of rice and soybean crops, we applied them to these datasets. We used a 12-dimensional feature extraction method to preprocess the SNP datasets, which creates a fixed-length 12-dimensional feature vector for each SNP sequence. For the protein sequence, we used a 60-dimensional feature extraction method, which creates a

60-dimensional feature vector for each protein sequence. After preprocessing, we examined the performance of NCHFS-ACO on the SNP and protein datasets and measured the results in terms of SI. The experimental finding demonstrates that the NCHFS-ACO performed well in comparison to other existing method proposed by Solorio et al. [1]. Then we investigated the performance of MOB-IC on the SNP datasets by supplying the processed SNP data in an incremental manner and assessing the results in terms of SI and Calinski–Harabasz Index (CH Index). The experimental results reveal that the proposed MOB-IC performed well in comparison to other state-of-the-art methods.

8.2 Future Research Directions

Although there has been notable advancement in feature selection, scalable feature extraction, and incremental clustering for analyzing plant genome data, there are still other intriguing future avenues to investigate, as follows:

(i) **Novel feature Selection approaches for unlabeled genome data:**

The developed hybrid feature selection approach based on ant colony optimization incorporates one filter measure and one wrapper measure. In forthcoming research, we could explore the inclusion of additional filter measures or the utilization of multiple filter measures simultaneously [166], aiming to explore the complex feature interaction to enhance the robustness and effectiveness of the feature selection process.

(ii) **Novel GPU based alignment-free feature extraction approaches:**

The Developed feature extraction approaches extract features based on the arrangement of nucleotides or based on the chemical properties of nucleotides. However, this work can be extended by incorporating some feature extraction criteria such as the most repeated pattern (MRP) count and the palindrome count, as well as some features based on the amino acid composition, i.e., amino acid count, their atomic composition, and their molecular weight, can be used to

enhance the performance of the proposed feature extraction method. In addition, the graphics processing unit (GPU)-based cuda programming [167] could be used to make the processed approach faster for the supercomputing infrastructure and able to handle petabytes of data.

(iii) **Multi-objective based incremental clustering using Apache Spark:**

The developed multi-objective based incremental clustering runs on a standalone machine. In future, this approach can be parallelized in distributed systems by using Apache Spark framework to reduce the time required to calculate individual objective functions when processing data with bigger input chunk sizes.

Bibliography

[1] Saúl Solorio-Fernández, J Ariel Carrasco-Ochoa, and José Fco Martínez-Trinidad. A new hybrid filter–wrapper feature selection method for clustering based on ranking. *Neurocomputing*, 214:866–880, 2016.

[2] Beatriz Remeseiro and Veronica Bolon-Canedo. A review of feature selection methods in medical applications. *Computers in biology and medicine*, 112:103375, 2019.

[3] Yvan Saeys, Inaki Inza, and Pedro Larranaga. A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19):2507–2517, 2007.

[4] Girija Attigeri, MM Manohara Pai, and Radhika M Pai. Feature selection using submodular approach for financial big data. *Journal of Information Processing Systems*, 15(6):1306–1325, 2019.

[5] Aradhana Behura. The cluster analysis and feature selection: Perspective of machine learning and image processing. *Data Analytics in Bioinformatics: A Machine Learning Perspective*, pages 249–280, 2021.

[6] Huan Liu and Lei Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on knowledge and data engineering*, 17(4):491–502, 2005.

[7] Jiliang Tang and Huan Liu. An unsupervised feature selection framework for social media data. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):2914–2927, 2014.

[8] Shiyu Zhou and Jionghua Jin. Automatic feature selection for unsupervised clustering of cycle-based signals in manufacturing processes. *Iie Transactions*, 37(6):569–584, 2005.

[9] Kumar N Neeraj and V Maurya. A review on machine learning (feature selection, classification and clustering) approaches of big data mining in different area of research. *Journal of Critical Reviews*, 7(19):2610–2626, 2020.

[10] Hugo Campos-de Quiroz. Plant genomics: an overview. *Biological research*, 35(3-4):385–399, 2002.

[11] Nozomu Yachie, Kazuhide Sekiyama, Junichi Sugahara, Yoshiaki Ohashi, and Masaru Tomita. Alignment-based approach for durable data storage into living organisms. *Biotechnology progress*, 23(2):501–505, 2007.

[12] Dan Wei, Qingshan Jiang, Yanjie Wei, and Shengrui Wang. A novel hierarchical clustering algorithm for gene sequences. *BMC bioinformatics*, 13(1):1–15, 2012.

[13] Jie Ren, Xin Bai, Yang Young Lu, Kujin Tang, Ying Wang, Gesine Reinert, and Fengzhu Sun. Alignment-free sequence analysis and applications. *Annual Review of Biomedical Data Science*, 1:93–114, 2018.

[14] Guangrong Li, Xiaohua Hu, Xiajiong Shen, Xin Chen, and Zhoujun Li. A novel unsupervised feature selection method for bioinformatics data sets through feature clustering. In *2008 IEEE international conference on granular computing*, pages 41–47. IEEE, 2008.

[15] Zryan Najat Rashid, Subhi RM Zebari, Karzan Hussein Sharif, and Karwan Jacksi. Distributed cloud computing and distributed parallel computing: A review. In *2018 International Conference on Advanced Science and Engineering (ICOASE)*, pages 167–172. IEEE, 2018.

[16] Lior Rokach and Oded Maimon. Clustering methods. *Data mining and knowledge discovery handbook*, pages 321–352, 2005.

[17] Manoranjan Dash and Huan Liu. Feature selection for clustering. In *Pacific-Asia Conference on knowledge discovery and data mining*, pages 110–121. Springer, 2000.

[18] Eduardo R Hruschka, Thiago F Covoes, and Nelson FF Ebecken. Feature selection for clustering problems: a hybrid algorithm that iterates between k-means and a bayesian filter. In *Fifth International Conference on Hybrid Intelligent Systems (HIS'05)*, pages 6–pp. IEEE, 2005.

[19] Yun Li, Bao-Liang Lu, and Zhong-Fu Wu. A hybrid method of unsupervised feature selection based on ranking. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 2, pages 687–690. IEEE, 2006.

[20] Ke Chen, Feng-Yu Zhou, and Xian-Feng Yuan. Hybrid particle swarm optimization with spiral-shaped mechanism for feature selection. *Expert Systems with Applications*, 128:140–156, 2019.

[21] Sina Tabakhi, Parham Moradi, and Fardin Akhlaghian. An unsupervised feature selection algorithm based on ant colony optimization. *Engineering Applications of Artificial Intelligence*, 32:112–123, 2014.

[22] J Dhalia Sweetlin, H Khanna Nehemiah, and Arputharaj Kannan. Feature selection using ant colony optimization with tandem-run recruitment to diagnose bronchitis from ct scan images. *Computer methods and programs in biomedicine*, 145:115–125, 2017.

[23] R Joseph Manoj, Anto Praveena, and K Vijayakumar. An aco–ann based feature selection algorithm for big data. *Cluster Computing*, 22(2):3953–3960, 2019.

[24] Wenping Ma, Xiaobo Zhou, Hao Zhu, Longwei Li, and Licheng Jiao. A two-stage hybrid ant colony optimization for high-dimensional feature selection. *Pattern Recognition*, 116:107933, 2021.

[25] Marco Dorigo and Luca Maria Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on evolutionary computation*, 1(1):53–66, 1997.

[26] Anton A Komar et al. Single nucleotide polymorphisms. *Methods in Molecular Biology*, 578, 2009.

[27] DF Feng, MS Johnson, and RF Doolittle. Aligning amino acid sequences: comparison of commonly used methods. *Journal of molecular evolution*, 21:112–125, 1985.

[28] Lilian TC França, Emanuel Carrilho, and Tarso BL Kist. A review of dna sequencing techniques. *Quarterly reviews of biophysics*, 35(2):169–200, 2002.

[29] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

[30] Mohiuddin Ahmed, Raihan Seraj, and Syed Mohammed Shamsul Islam. The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, 9(8):1295, 2020.

[31] James C Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & geosciences*, 10(2-3):191–203, 1984.

[32] Behrooz Hosseini and Kourosh Kiani. A big data driven distributed density based hesitant fuzzy clustering using apache spark with application to gene expression microarray. *Engineering Applications of Artificial Intelligence*, 79:100–113, 2019.

[33] Behrooz Hosseini and Kourosh Kiani. A robust distributed big data clustering-based on adaptive density partitioning using apache spark. *Symmetry*, 10(8):342, 2018.

[34] Daxin Jiang, Chun Tang, and Aidong Zhang. Cluster analysis for gene expression data: a survey. *IEEE Transactions on Knowledge & Data Engineering*, (11):1370–1386, 2004.

[35] Mahmoud Ghandi, Dongwon Lee, Morteza Mohammad-Noori, and Michael A Beer. Enhanced regulatory sequence prediction using gapped k-mer features. *PLoS computational biology*, 10(7):e1003711, 2014.

[36] M Muthulakshmi et al. A novel feature extraction from genome sequences for taxonomic classification of living organisms. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, 12(2):1436–1451, 2021.

[37] Kakoli Banerjee and Vikram Bali. Design and development of bioinformatics feature based dna sequence data compression algorithm. *EAI Endorsed Transactions on Pervasive Health and Technology*, 5(20), 2019.

[38] Claire Nédellec. Machine learning for information extraction in genomics—state of the art and perspectives. In *Text Mining and its Applications*, pages 99–118. Springer, 2004.

[39] Razvan Bunescu, Ruifang Ge, Rohit J Kate, Edward M Marcotte, Raymond J Mooney, Arun K Ramani, and Yuk Wah Wong. Comparative experiments on learning information extractors for proteins and their interactions. *Artificial intelligence in medicine*, 33(2):139–155, 2005.

[40] Ge Wang, Pengbo Pu, and Tingyan Shen. An efficient gene bigdata analysis using machine learning algorithms. *Multimedia Tools and Applications*, 79(15):9847–9870, 2020.

[41] Goo Jun, Mary Kate Wing, Gonçalo R Abecasis, and Hyun Min Kang. An efficient and scalable analysis framework for variant extraction and refinement from population-scale dna sequence data. *Genome research*, 25(6):918–925, 2015.

[42] Mehrdad J Gangeh, Hadi Zarkoob, and Ali Ghodsi. Fast and scalable feature selection for gene expression data using hilbert-schmidt independence cri-

terion. *IEEE/ACM transactions on computational biology and bioinformatics*, 14(1):167–181, 2017.

[43] Yewang Chen, Xiaoliang Hu, Wentao Fan, Lianlian Shen, Zheng Zhang, Xin Liu, Jixiang Du, Haibo Li, Yi Chen, and Hailin Li. Fast density peak clustering for large scale data based on knn. *Knowledge-Based Systems*, 187:104824, 2020.

[44] T Reichhardt, DA Benson, I Karsch-Mizrachi, DJ Lipman, J Ostell, BA Rapp, DL Wheeler, A Bairoch, R Apweiler, RD Fleischmann, et al. What is bioinformatics? an introduction and overview. *Yearbook of medical informatics*, 10(01):83–100, 2001.

[45] Fazli Can. Incremental clustering for dynamic information processing. *ACM Transactions on Information Systems (TOIS)*, 11(2):143–164, 1993.

[46] Absalom E Ezugwu, Abiodun M Ikotun, Olaide O Oyelade, Laith Abualigah, Jeffery O Agushaka, Christopher I Eke, and Andronicus A Akinyelu. A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. *Engineering Applications of Artificial Intelligence*, 110:104743, 2022.

[47] Jeyachandran Sivakamavalli, Kiyun Park, and Ihn-Sil Kwak. Genome databases, types and applications: An overview. 2020.

[48] Markelle Kelly, Rachel Longjohn, and Kolby Nottingham. The uci machine learning repository. *https://archive.ics.uci.edu*, 2023.

[49] Anne V Brown, Shawn I Conners, Wei Huang, Andrew P Wilkey, David Grant, Nathan T Weeks, Steven B Cannon, Michelle A Graham, and Rex T Nelson. A new decade and new data at soybase, the usda-ars soybean genetics and genomics database. *Nucleic Acids Research*, 49(D1):D1496–D1501, 2021.

[50] Preeti Jha, Aruna Tiwari, Neha Bharill, Milind Ratnaparkhe, Mukkamalla Mounika, and Neha Nagendra. Apache spark based kernelized fuzzy clustering

framework for single nucleotide polymorphism sequence analysis. *Computational Biology and Chemistry*, 92:107454, 2021.

[51] Yoshihiro Kawahara, Melissa de la Bastide, John P Hamilton, Hiroyuki Kanamori, W Richard McCombie, Shu Ouyang, David C Schwartz, Tsuyoshi Tanaka, Jianzhong Wu, Shiguo Zhou, et al. Improvement of the oryza sativa nipponbare reference genome using next generation sequence and optical map data. *Rice*, 6:1–10, 2013.

[52] Siyu Han, Yanchun Liang, Qin Ma, Yangyi Xu, Yu Zhang, Wei Du, Cankun Wang, and Ying Li. Lncfinder: an integrated platform for long non-coding rna identification utilizing sequence intrinsic composition, structural information and physicochemical property. *Briefings in bioinformatics*, 20(6):2009–2027, 2019.

[53] Preeti Jha, Aruna Tiwari, Neha Bharill, Milind Ratnaparkhe, Om Prakash Patel, Nilagiri Harshith, Mukkamalla Mounika, and Neha Nagendra. Apache spark-based scalable feature extraction approaches for protein sequence and their clustering performance analysis. *International Journal of Data Science and Analytics*, 15(4):359–378, 2023.

[54] Sivadi Balakrishna. Multi objective-based incremental clustering by fast search technique for dynamically creating and updating clusters in large data. *Cluster Computing*, 25(2):1441–1457, 2022.

[55] Amber Abernathy and M Emre Celebi. The incremental online k-means clustering algorithm and its application to color quantization. *Expert Systems with Applications*, 207:117927, 2022.

[56] Lei Yu and Huan Liu. Efficient feature selection via analysis of relevance and redundancy. *The Journal of Machine Learning Research*, 5:1205–1224, 2004.

[57] Saúl Solorio-Fernández, J Ariel Carrasco-Ochoa, and José Fco Martínez-Trinidad. A review of unsupervised feature selection methods. *Artificial Intelligence Review*, 53(2):907–948, 2020.

[58] Utkarsh Mahadeo Khaire and R Dhanalakshmi. Stability of feature selection algorithm: A review. *Journal of King Saud University-Computer and Information Sciences*, 34(4):1060–1073, 2022.

[59] Archana Shivdas Sumant and Dipak Patil. Stability investigation of ensemble feature selection for high dimensional data analytics. In *International Conference on Image Processing and Capsule Networks*, pages 801–815. Springer, 2022.

[60] Monami Banerjee and Nikhil R Pal. Feature selection with svd entropy: Some modification and extension. *Information Sciences*, 264:118–134, 2014.

[61] Saúl Solorio-Fernández, José Fco Martínez-Trinidad, and J Ariel Carrasco-Ochoa. A new unsupervised spectral feature selection method for mixed data: a filter approach. *Pattern Recognition*, 72:314–326, 2017.

[62] Hamidreza Rashidy Kanan, Karim Faez, and Mehdi Hosseinzadeh. Face recognition system using ant colony optimization-based selected features. In *2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications*, pages 57–62. IEEE, 2007.

[63] Mehran Javani, Karim Faez, and Davoud Aghlmandi. Clustering and feature selection via pso algorithm. In *2011 international symposium on artificial intelligence and signal processing (AISP)*, pages 71–76. IEEE, 2011.

[64] KP Swetha and V Susheela Devi. Simultaneous feature selection and clustering using particle swarm optimization. In *International Conference on Neural Information Processing*, pages 509–515. Springer, 2012.

[65] Jay Prakash and Pramod Kumar Singh. Particle swarm optimization with k-means for simultaneous feature selection and data clustering. In *2015 Second*

International Conference on Soft Computing and Machine Intelligence (ISCMI), pages 74–78. IEEE, 2015.

[66] Jay Prakash and Pramod Kumar Singh. Gravitational search algorithm and k-means for simultaneous feature selection and data clustering: a multi-objective approach. *Soft Computing*, 23(6):2083–2100, 2019.

[67] Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[68] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[69] Deng Cai, Chiyuan Zhang, and Xiaofei He. Unsupervised feature selection for multi-cluster data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 333–342, 2010.

[70] Zechao Li, Yi Yang, Jing Liu, Xiaofang Zhou, and Hanqing Lu. Unsupervised feature selection using nonnegative spectral analysis. In *Proceedings of the AAAI conference on artificial intelligence*, volume 26, pages 1026–1032, 2012.

[71] Ferdinando S Samaria and Andy C Harter. Parameterisation of a stochastic model for human face identification. In *Proceedings of 1994 IEEE workshop on applications of computer vision*, pages 138–142. IEEE, 1994.

[72] Michael J Lyons, Julien Budynek, and Shigeru Akamatsu. Automatic classification of single facial images. *IEEE transactions on pattern analysis and machine intelligence*, 21(12):1357–1362, 1999.

[73] Minnan Luo, Feiping Nie, Xiaojun Chang, Yi Yang, Alexander G Hauptmann, and Qinghua Zheng. Adaptive unsupervised feature selection with structure regularization. *IEEE transactions on neural networks and learning systems*, 29(4):944–956, 2017.

[74] Ahmed Al-Ani. Feature subset selection using ant colony optimization. *International journal of computational intelligence*, 2005.

[75] Rahul Karthik Sivagaminathan and Sreeram Ramakrishnan. A hybrid approach for feature subset selection using neural networks and ant colony optimization. *Expert systems with applications*, 33(1):49–60, 2007.

[76] Bolun Chen, Ling Chen, and Yixin Chen. Efficient ant colony optimization for image feature selection. *Signal processing*, 93(6):1566–1576, 2013.

[77] Xiabi Liu, Ling Ma, Li Song, Yanfeng Zhao, Xinming Zhao, and Chunwu Zhou. Recognizing common ct imaging signs of lung diseases through a new feature selection method based on fisher criterion and genetic optimization. *IEEE journal of biomedical and health informatics*, 19(2):635–647, 2014.

[78] Sanghamitra Bandyopadhyay. An efficient technique for superfamily classification of amino acid sequences: feature extraction, fuzzy clustering and prototype selection. *Fuzzy Sets and Systems*, 152(1):5–16, 2005.

[79] Terje Kristensen and Fabien Guillaume. Classification of dna sequences by a mlp and svm network. In *Proceedings of the International Conference on Bioinformatics & Computational Biology (BIOCOMP)*, page 1, 2013.

[80] Junpeng Bao, Ruiyu Yuan, and Zhe Bao. An improved alignment-free model for dna sequence similarity metric. *BMC bioinformatics*, 15(1):1–15, 2014.

[81] Long Shi and Hailan Huang. Dna sequences analysis based on classifications of nucleotide bases. *Affective Computing and Intelligent Interaction*, pages 379–384, 2012.

[82] Ngoc G Nguyen, Vu Anh Tran, Dau Phan, Favorisen R Lumbanraja, Mohammad Reza Faisal, Bahriddin Abapihi, Mamoru Kubo, and Kenji Satou. Dna sequence classification by convolutional neural network. *Journal Biomedical Science and Engineering*, 9(5):280–286, 2016.

[83] Geofrey G Towell, Jude W Shavlik, Michiel O Noordewier, et al. Refinement of approximate domain theories by knowledge-based neural networks. In *Proceedings of the eighth National conference on Artificial intelligence*, volume 2, pages 861–866. Boston, MA, 1990.

[84] Jinyan Li and Limsoon Wong. Using rules to analyse bio-medical data: a comparison between c4. 5 and pcl. In *International Conference on Web-Age Information Management*, pages 254–265. Springer, 2003.

[85] MASANORI Higashihara, JOVAN DAVID Rebolledo-Mendez, YOICHI Yamada, and KENJI Satou. Application of a feature selection method to nucleosome data: accuracy improvement and comparison with other methods. *WSEAS Transactions on Biology and Biomedicine*, 5(5):95–104, 2008.

[86] Raid Alzubi, Naeem Ramzan, Hadeel Alzoubi, and Abbes Amira. A hybrid feature selection method for complex diseases snps. *IEEE Access*, 6:1292–1301, 2017.

[87] Marwah A Helaly, Sherine Rady, and Mostafa M Aref. Deep learning for taxonomic classification of biological bacterial sequences. In *Machine Learning and Big Data Analytics Paradigms: Analysis, Applications and Challenges*, pages 393–413. Springer, 2021.

[88] Jasbir Dhaliwal and John Wagner. A novel feature extraction method based on highly expressed snps for tissue-specific gene prediction. *Journal of Big Data*, 8(1):1–13, 2021.

[89] Junwei Xiao, Jianfeng Lu, and Xiangyu Li. Davies bouldin index based hierarchical initialization k-means. *Intelligent Data Analysis*, 21(6):1327–1338, 2017.

[90] Robson P Bonidia, Douglas S Domingues, Danilo S Sanches, and André C P L F de Carvalho. MathFeature: feature extraction package for DNA, RNA and protein sequences based on mathematical descriptors. *Briefings in Bioinformatics*, 23(1):bbab434, 11 2021.

[91] Jun Zhang, Rod Chiodini, Ahmed Badr, and Genfa Zhang. The impact of next-generation sequencing on genomics. *Journal of genetics and genomics*, 38(3):95–109, 2011.

[92] Shibu Yooseph, Weizhong Li, and Granger Sutton. Gene identification and protein classification in microbial metagenomic sequence data via incremental clustering. *BMC bioinformatics*, 9(1):1–13, 2008.

[93] Peng Zhou, Yi-Dong Shen, Liang Du, Fan Ye, and Xuejun Li. Incremental multi-view spectral clustering. *Knowledge-Based Systems*, 174:73–86, 2019.

[94] Ling Wang, Peipei Xu, and Qian Ma. Incremental fuzzy clustering of time series. *Fuzzy Sets and Systems*, 421:62–76, 2021.

[95] Sirisup Laohakiat and Vera Sa-Ing. An incremental density-based clustering framework using fuzzy local clustering. *Information Sciences*, 547:404–426, 2021.

[96] Kalyanmoy Deb, Karthik Sindhya, and Jussi Hakanen. Multi-objective optimization. In *Decision Sciences*, pages 161–200. CRC Press, 2016.

[97] Ye Tian, Shangshang Yang, and Xingyi Zhang. An evolutionary multiobjective optimization based fuzzy method for overlapping community detection. *IEEE Transactions on Fuzzy Systems*, 28(11):2841–2855, 2019.

[98] Zahra Zareizadeh, Mohammad Sadegh Helfroush, Akbar Rahideh, and Kamran Kazemi. A robust gene clustering algorithm based on clonal selection in multiobjective optimization framework. *Expert Systems with Applications*, 113:301–314, 2018.

[99] Weimin Li, Xiaokang Zhou, Chao Yang, Yuting Fan, Zhao Wang, and Yanxia Liu. Multi-objective optimization algorithm based on characteristics fusion of dynamic social networks for community discovery. *Information Fusion*, 79:110–123, 2022.

[100] Ye Tian, Yuandong Feng, Xingyi Zhang, and Changyin Sun. A fast clustering based evolutionary algorithm for super-large-scale sparse multi-objective optimization. *IEEE/CAA Journal of Automatica Sinica*, 10(4):1048–1063, 2022.

[101] Lilian Astrid Bejarano, Helbert Eduardo Espitia, and Carlos Enrique Montenegro. Clustering analysis for the pareto optimal front in multi-objective optimization. *Computation*, 10(3):37, 2022.

[102] Shihua Liu, Bingzhong Zhou, Decai Huang, Liangzhong Shen, et al. Clustering mixed data by fast search and find of density peaks. *Mathematical Problems in Engineering*, 2017, 2017.

[103] Linlin Zhuo, Kenli Li, Bo Liao, Hao Li, Xiaohui Wei, and Keqin Li. Hcfs: a density peak based clustering algorithm employing a hierarchical strategy. *IEEE Access*, 7:74612–74624, 2019.

[104] Fanyu Bu, Zhikui Chen, Peng Li, Tong Tang, and Ying Zhang. A high-order cfs algorithm for clustering big data. *Mobile Information Systems*, 2016:1–8, 2016.

[105] Zachary D Stephens, Skylar Y Lee, Faraz Faghri, Roy H Campbell, Chengxiang Zhai, Miles J Efron, Ravishankar Iyer, Michael C Schatz, Saurabh Sinha, and Gene E Robinson. Big data: astronomical or genomical? *PLoS biology*, 13(7):e1002195, 2015.

[106] Monerah Al-Mekhlal and Amir Ali Khwaja. A synthesis of big data definition and characteristics. In *2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*, pages 314–322. IEEE, 2019.

[107] Alessio Botta, Walter De Donato, Valerio Persico, and Antonio Pescapé. Integration of cloud computing and internet of things: a survey. *Future generation computer systems*, 56:684–700, 2016.

[108] Neil Gershenfeld, Raffi Krikorian, and Danny Cohen. The internet of things. *Scientific American*, 291(4):76–81, 2004.

[109] Amanda Lazar, Christian Koehler, Theresa Jean Tanenbaum, and David H Nguyen. Why we use and abandon smart devices. In *Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing*, pages 635–646, 2015.

[110] Vernon Turner, John F Gantz, David Reinsel, and Stephen Minton. The digital universe of opportunities: Rich data and the increasing value of the internet of things. *IDC Analyze the Future*, 16:13–19, 2014.

[111] John Gantz and David Reinsel. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView: IDC Analyze the future*, 2007(2012):1–16, 2012.

[112] David Reinsel-John Gantz-John Rydning, John Reinsel, and John Gantz. The digitization of the world from edge to core. *Framingham: International Data Corporation*, 16:1–28, 2018.

[113] Barton E Slatko, Andrew F Gardner, and Frederick M Ausubel. Overview of next-generation sequencing technologies. *Current protocols in molecular biology*, 122(1):e59, 2018.

[114] Philippe Collas. The current state of chromatin immunoprecipitation. *Molecular biotechnology*, 45:87–100, 2010.

[115] Lingyun Song and Gregory E Crawford. Dnase-seq: a high-resolution technique for mapping active gene regulatory elements across the genome from mammalian cells. *Cold Spring Harbor Protocols*, 2010(2):pdb–prot5384, 2010.

[116] Juliana S Bernardes, Jorge H Fernandez, and Ana Tereza R Vasconcelos. Structural descriptor database: a new tool for sequence-based functional site prediction. *BMC bioinformatics*, 9(1):1–12, 2008.

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

[118] Peter Mika. Flink: Semantic web technology for the extraction and analysis of social networks. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(2):211–223, 2005.

[119] Yingyi Bu, Bill Howe, Magdalena Balazinska, and Michael D Ernst. Haloop: efficient iterative data processing on large clusters. *Proceedings of the VLDB Endowment*, 3(1-2):285–296, 2010.

[120] Jielong Xu, Zhenhua Chen, Jian Tang, and Sen Su. T-storm: traffic-aware online scheduling in storm. In *Proc. of IEEE 34th International Conference on Distributed Computing Systems*, pages 535–544. IEEE, Madrid, Spain, June 30-July 3, 2014.

[121] Thomas A Runkler and Helmut Krcmar. Stream processing on demand for lambda architectures. In *Proc. of 12th European Workshop on Computer Performance Engineering, Computer Performance Engineering*, volume 9272 of *Lecture Notes in Computer Science*, pages 243–257. 2015.

[122] Matei Zaharia, Mosharaf Chowdhury, Tathagata Das, Ankur Dave, Justin Ma, Murphy McCauley, Michael J Franklin, Scott Shenker, and Ion Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Proc. of 9th USENIX Conference on Networked Systems Design and Implementation*, pages 2–2. USENIX Association, San Jose, CA, April, 2012.

[123] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, and Ion Stoica. Spark: cluster computing with working sets. *HotCloud*, 10:10–10, 2010.

[124] Apache Spark. Apache spark. *Retrieved January*, 17:2018, 2018.

[125] Firat Tekiner and John A Keane. Big data framework. In *2013 IEEE international conference on systems, man, and cybernetics*, pages 1494–1499. IEEE, 2013.

[126] Azlinah Mohamed, Maryam Khanian Najafabadi, Yap Bee Wah, Ezzatul Akmal Kamaru Zaman, and Ruhaila Maskat. The state of the art and taxonomy of big data analytics: view from new big data framework. *Artificial Intelligence Review*, 53(2):989–1037, 2020.

[127] Yinan Xu, Hui Liu, and Zhihao Long. A distributed computing framework for wind speed big data forecasting on apache spark. *Sustainable Energy Technologies and Assessments*, 37:100582, 2020.

[128] Panos Louridas and Christof Ebert. Embedded analytics and statistics for big data. *IEEE software*, 30(6):33–39, 2013.

[129] Dhruba Borthakur et al. Hdfs architecture guide. *Hadoop Apache Project*, 53(1-13):2, 2008.

[130] Salman Salloum, Ruslan Dautov, Xiaojun Chen, Patrick Xiaogang Peng, and Joshua Zhexue Huang. Big data analytics on apache spark. *International Journal of Data Science and Analytics*, 1(3):145–164, 2016.

[131] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

[132] Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[133] Raimundo Real and Juan M Vargas. The probabilistic basis of jaccard's index of similarity. *Systematic biology*, 45(3):380–385, 1996.

[134] Tadeusz Caliński and Jerzy Harabasz. A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27, 1974.

[135] Verena Zuber, A Pedro Duarte Silva, and Korbinian Strimmer. A novel algorithm for simultaneous snp selection in high-dimensional genome-wide association studies. *BMC bioinformatics*, 13(1):1–8, 2012.

[136] Jiming Jiang, Cong Li, Debashis Paul, Can Yang, Hongyu Zhao, et al. On high-dimensional misspecified mixed model analysis in genome-wide association study. *The Annals of Statistics*, 44(5):2127–2160, 2016.

[137] Mingshun Yuan, Zijiang Yang, Guangzao Huang, and Guoli Ji. Feature selection by maximizing correlation information for integrated high-dimensional protein data. *Pattern Recognition Letters*, 92:17–24, 2017.

[138] PETERY Chou and Gerald D Fasman. Amino acid sequence. *Adv Enzymol Relat Areas Mol Biol*, 47:45, 2009.

[139] Eric E Schadt, John Lamb, Xia Yang, Jun Zhu, Steve Edwards, Debraj GuhaThakurta, Solveig K Sieberts, Stephanie Monks, Marc Reitman, Chunsheng Zhang, et al. An integrative genomics approach to infer causal associations between gene expression and disease. *Nature genetics*, 37(7):710–717, 2005.

[140] David T Jones, WR Taylort, and Janet M Thornton. A new approach to protein fold recognition. *Nature*, 358(6381):86–89, 1992.

[141] Zachary D Stephens, Skylar Y Lee, Faraz Faghri, Roy H Campbell, Chengxiang Zhai, Miles J Efron, Ravishankar Iyer, Michael C Schatz, Saurabh Sinha, and Gene E Robinson. Big data: astronomical or genomical? *PLoS biology*, 13(7):e1002195, 2015.

[142] Eric J Sedivy, Faqiang Wu, and Yoshie Hanzawa. Soybean domestication: the origin, genetic architecture and molecular bases. *New Phytologist*, 214(2):539–553, 2017.

[143] Philip Traldi Wysmierski and Natal Antonio Vello. The genetic base of brazilian soybean cultivars: evolution over time and breeding implications. *Genetics and molecular Biology*, 36:547–555, 2013.

[144] Yanting Shen, Huilong Du, Yucheng Liu, Lingbin Ni, Zheng Wang, Chengzhi Liang, and Zhixi Tian. Update soybean zhonghuang 13 genome to a golden reference. *Sci. China Life Sci*, 62:1257–1260, 2019.

[145] Jeong-Dong Lee, J Grover Shannon, Tri D Vuong, and Henry T Nguyen. Inheritance of salt tolerance in wild soybean (glycine soja sieb. and zucc.) accession pi483463. *Journal of Heredity*, 100(6):798–801, 2009.

[146] Min Xie, Claire Yik-Lok Chung, Man-Wah Li, Fuk-Ling Wong, Xin Wang, Ailin Liu, Zhili Wang, Alden King-Yung Leung, Tin-Hang Wong, Suk-Wah Tong, et al. A reference-grade wild soybean genome. *Nature communications*, 10(1):1–12, 2019.

[147] Sobin Kim and Ashish Misra. Snp genotyping: technologies and biomedical applications. *Annu. Rev. Biomed. Eng.*, 9:289–320, 2007.

[148] Locedie Mansueto, Roven Rommel Fuentes, Frances Nikki Borja, Jeffery Detras, Juan Miguel Abriol-Santos, Dmytro Chebotarov, Millicent Sanciangco, Kevin Palis, Dario Copetti, Alexandre Poliakov, et al. Rice snp-seek database update: new snps, indels, and queries. *Nucleic acids research*, 45(D1):D1075–D1081, 2017.

[149] Nonoy Bandillo, Chitra Raghavan, Pauline Andrea Muyco, Ma Anna Lynn Sevilla, Irish T Lobina, Christine Jade Dilla-Ermita, Chih-Wei Tung, Susan McCouch, Michael Thomson, Ramil Mauleon, et al. Multi-parent advanced generation inter-cross (magic) populations in rice: progress and potential for genetics research and breeding. *Rice*, 6(1):1–15, 2013.

[150] Christine Jade Dilla-Ermita, Erwin Tandayu, Venice Margarette Juanillas, Jeffrey Detras, Dennis Nicuh Lozada, Maria Stefanie Dwiyanti, Casiana Vera Cruz, Edwige Gaby Nkouaya Mbanjo, Edna Ardales, Maria Genaleen Diaz, et al. Genome-wide association analysis tracks bacterial leaf blight resistance loci in rice diverse germplasm. *Rice*, 10(1):1–17, 2017.

[151] Preeti Jha, Aruna Tiwari, Neha Bharill, Milind Ratnaparkhe, Om Prakash Patel, Nilagiri Harshith, Mukkamalla Mounika, and Neha Nagendra. Apache spark-based scalable feature extraction approaches for protein sequence and their clus-

tering performance analysis. *International Journal of Data Science and Analytics*, 15(4):359–378, 2023.

[152] Simon Fong, Yan Zhuang, Rui Tang, Xin-She Yang, Suash Deb, et al. Selecting optimal feature set in high-dimensional data by swarm search. *Journal of Applied Mathematics*, 2013, 2013.

[153] Nigel R Franks and Tom Richardson. Teaching in tandem-running ants. *Nature*, 439(7073):153–153, 2006.

[154] Xiaofei He, Deng Cai, and Partha Niyogi. Laplacian score for feature selection. *Advances in neural information processing systems*, 18, 2005.

[155] Ren Qi, Anjun Ma, Qin Ma, and Quan Zou. Clustering and classification methods for single-cell rna-sequencing data. *Briefings in bioinformatics*, 21(4):1196–1208, 2020.

[156] Rajesh Dwivedi, Aruna Tiwari, Neha Bharill, and Milind Ratnaparkhe. A hybrid feature selection approach for data clustering based on ant colony optimization. In Mohammad Tanveer, Sonali Agarwal, Seiichi Ozawa, Asif Ekbal, and Adam Jatowt, editors, *Neural Information Processing*, pages 659–670, Cham, 2023. Springer International Publishing.

[157] Didier Debroas, Jean-François Humbert, François Enault, Gisèle Bronner, Michael Faubladier, and Emmanuel Cornillot. Metagenomic approach studying the taxonomic and functional diversity of the bacterial community in a mesotrophic lake (lac du bourget–france). *Environmental microbiology*, 11(9):2412–2424, 2009.

[158] Loris Nanni, Sheryl Brahnam, and Alessandra Lumini. High performance set of pseaac and sequence based descriptors for protein classification. *Journal of Theoretical Biology*, 266(1):1–10, 2010.

[159] Annick Lesne. Shannon entropy: a rigorous notion at the crossroads between probability, information theory, dynamical systems and statistical physics. *Mathematical Structures in Computer Science*, 24(3):e240311, 2014.

[160] Rajesh Dwivedi, Aruna Tiwari, Neha Bharill, and Milind Ratnaparkhe. A novel clustering-based hybrid feature selection approach using ant colony optimization. *Arabian Journal for Science and Engineering*, 48:10727–10744, 2023.

[161] Rahul Kumar, Rajesh Dwivedi, and Ebenezer Jangam. Hybrid fuzzy c-means using bat optimization and maxi-min distance classifier. In *Advances in Computing and Data Sciences: Third International Conference, ICACDS 2019, Ghaziabad, India, April 12–13, 2019, Revised Selected Papers, Part II 3*, pages 68–79. Springer, 2019.

[162] Kristina P Sinaga and Miin-Shen Yang. Unsupervised k-means clustering algorithm. *IEEE access*, 8:80716–80727, 2020.

[163] Nicholas M Luscombe, Dov Greenbaum, and Mark Gerstein. What is bioinformatics? an introduction and overview. *Yearbook of medical informatics*, 10(01):83–100, 2001.

[164] Petra Nowotny, Jennifer M Kwon, and Alison M Goate. Snp analysis to dissect human traits. *Current Opinion in Neurobiology*, 11(5):637–641, 2001.

[165] Preeti Jha, Aruna Tiwari, Neha Bharill, Milind Ratnaparkhe, Om Prakash Patel, Nilagiri Harshith, Mukkamalla Mounika, and Neha Nagendra. Apache spark-based scalable feature extraction approaches for protein sequence and their clustering performance analysis. *International Journal of Data Science and Analytics*, 15(4):359–378, 2023.

[166] Ruwang Jiao, Bing Xue, and Mengjie Zhang. Learning to preselection: A filter-based performance predictor for multiobjective feature selection in classification. *IEEE Transactions on Evolutionary Computation*, 2024.

[167] Bertil Schmidt and Andreas Hildebrandt. From gpus to ai and quantum: three waves of acceleration in bioinformatics. *Drug Discovery Today*, page 103990, 2024.