A Scalable method for extracting features using a complex network from SNP sequences and a novel Scalable Max of Min Algorithm for clustering

MS (Research) Thesis

By

Achint Kumar Kansal

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

NOVEMBER 2024

A Scalable method for extracting features using a complex network from SNP sequences and a novel Scalable Max of Min Algorithm for clustering

A THESIS

submitted to the

INDIAN INSTITUTE OF TECHNOLOGY INDORE

in fulfillment of the requirements for

the award of the degree

of

MS (Research)

By

Achint Kumar Kansal

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

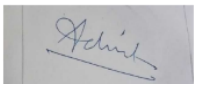
NOVEMBER 2024

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled "A Scalable method for extracting features using a complex network from SNP sequences and a novel Scalable Max of Min Algorithm for clustering" in the fulfilment of the requirements for the award of the degree of MASTER OF SCIENCE (RESEARCH) and submitted in the DISCIPLINE 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 August 2022 to May 2024 under the supervision of Prof. Aruna Tiwari, Professor, Indian Institute of Technology Indore, 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.



22-11-2024

Signature of the Student with Date

(Achint Kumar Kansal)

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

Signature of Thesis Supervisor

with Date **22/11/2024**

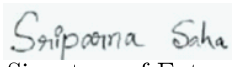
(Prof. Aruna Tiwari)

Achint Kumar Kansal has successfully given his MS (Research) Oral Examination held on 22-11-2024

22/11/2024

Signature of Chairperson, OEB

Date:

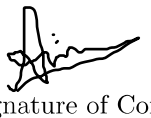


Signature of External Examiner

Date: 22/11/2024

Signature of Thesis Supervisor

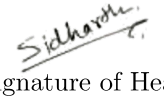
Date: **22/11/2024**



22/11/24

Signature of Convener, DPGC

Date:



Signature of Head of Discipline

Date: 22/11/24

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my heartfelt gratitude to a number of persons who, in one or another other way contributed by making this time learnable, enjoyable, and bearable. First, I would like to thank my supervisor **Prof. Aruna Tiwari**, who was a constant source of inspiration during my work. Without her constant guidance and research directions, this research work would not have been possible. Her continuous support and encouragement have motivated me to remain streamlined in my research work.

I am thankful to the PSPC members, **Dr. Ranveer Singh** and **Prof. Abhishek Srivastava** for taking some valuable time to evaluate my progress throughout my journey in IIT Indore. Their good comments and suggestions helped me to improve my work at various stages.

I extend my sincere thanks to **Dr. Milind Ratnaparkhe** from **ICAR-IISR Indore** for providing me with the necessary infrastructure and resources for my research. I would also like to thank my seniors, **Dr. Neha Bharill**, **Dr. Om Prakash Dr. Preeti Jha**, **Mr. Rajesh Dwivedi**, **Mr. Abhishek Tripathi**, **Ms. Ananya Roy** and **Ms. Sujata Gudge** for guiding me throughout my research.

I would like to appreciate the fine company of my dearest friends **Nisha, Shibani Mam, Bharat Sir, Urvesh, Sai Reddy,** and **Ashok Dada**, who have supported me and taken care of me throughout my stay at IIT Indore. I am also grateful to the institute staff for their unfailing support and assistance.

I would like to express my heartfelt respect to my grandmother **Sudha Gupta (Ama)**, my father **Mr. Tarun Kumar Gupta** and my sister **Vanshika Kansal** for the love, care and support they have provided to me throughout my life. Also, I would like to dedicate this thesis to my mother **Lt. Ritu Gupta**, whose memory and love guide me through every step of this journey.

Finally, I am thankful to all who directly or indirectly contributed, helped and supported me.

Achint Kumar Kansal

To my family and friends

Abstract

Feature extraction is pivotal in bioinformatics as it converts variable-length genome sequences into fixed-length mathematical feature vectors, which serve as input for clustering algorithms to cluster similar sequences. One of the types of genome sequences is the Single Nucleotide Polymorphism (SNP), which categorises individuals into risk categories for diseases and predicts treatment outcomes more reliably.

Extracting features with current state-of-the-art approaches from SNP sequences poses many challenges, including extracting similar features for distinct sequences and lacking context-based features. These approaches also take enormous time to compute features for a huge amount of SNP sequences. Therefore, a scalable approach to extract features is proposed based on a complex network, which converts the real-life SNP sequences (collected from ICAR-Indian Institute of Soybean Research Indore) into the complex network and extracts the proposed relevant features. The proposed approach distributes the sequences on various cores using Apache Spark Big Data framework. Hence, the time utilised to extract those features has reduced drastically. The evaluation of the proposed scalable feature extraction approach is done by applying K-means and Fuzzy c-means (FCM) algorithms to assess the performance of the proposed feature vector set compared with the other state-of-the-art approaches for feature extraction in terms of the Silhouette index and the Calinski-Harabasz index. Promising results for real-life SNP datasets have been obtained when our proposed feature vectors are compared with other state-of-the-art approaches.

Additionally, as most SNP datasets are unlabeled, determining the optimal number of clusters presents another significant challenge. A novel scalable algorithm called the S-MaxMin algorithm is proposed based on the distance metric to find the optimal number of clusters. The proposed S-MaxMin algorithm is being tested on different datasets, including various labelled benchmark datasets, giving the same number of clusters as the actual number of classes. Also, the S-MaxMin algorithm is tested on various real-life plant genome SNP datasets (unlabelled), which yielded approximately the same number of clusters as the clusters with a high Silhouette index score.

Furthermore, these two proposed approaches, i.e. a scalable approach to extract features based on a complex network and the S-MaxMin algorithm, are integrated to develop a Scalable Integrated Framework that first preprocesses raw sequences and then identifies known crops similar to the unknown plant species that can assist plant breeders (agricultural scientists) in enhancing seed quality. This framework helps in identifying unique special traits in unknown plant species such as rust resistance, drought resistance, etc., which can then be used to perform genetic engineering to improve those traits.

Keywords: Feature Extraction, S-MaxMin, Apache Spark, Fuzzy c-means clustering, K-means clustering, Single Nucleotide Polymorphism.

List of Publications

In Refereed Journals

A.K. Kansal, A. Tiwari, M. Ratnaparkhe, R. Dwivedi and P. Jha. *A Scalable method for extracting features using a complex network from SNP sequences and clustering using the Scalable Max of Min Algorithm*, Soft Computing, Springer.
(Accepted)

Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

ML Machine Learning

FCM Fuzzy c-means

SI Silhouette Index

CH Calinski-Harabasz index

WGS Whole Genome Sequence

SNP Single Nucleotide Polymorphism

DNA Deoxyribonucleic acid

RNA Ribonucleic acid

HGP Human Genome Project

S-MaxMin Scalable Max of Min algorithm

BWA-mem Burrows Wheeler Alignment Maximal Exact Matching

BWT Burrows Wheeler Transform

BC(v) Vertex Betweenness Centrality

MBC(v) Mean Vertex Betweenness Centrality

BC(e) Edge Betweenness Centrality

MBC(e) Mean Edge Betweenness Centrality

Chapter 1

Introduction

Feature extraction is the transformation of raw data into meaningful numerical feature vectors. It involves identifying and selecting relevant characteristics from the raw data, reducing its dimensionality while retaining crucial information. In genomics, where vast amounts of biological data are generated from diverse sources, feature extraction is paramount. A key challenge today in genomics involves matching similar genomic sequences, including DNA/RNA, Protein, and SNP sequences. To match the similar sequences, it is necessary to employ clustering techniques. However, clustering also faces the challenge of presetting the number of clusters, as these genome datasets are unlabelled. Also, plant breeders (agriculture scientists) face the challenge of identifying known crops similar to unknown crops as no end-to-end tool has been developed, and each process has to be done with separate tools.

This thesis mainly focuses on feature extraction techniques for handling large amounts of genome sequences, approaches to find the optimal number of clusters in an unlabelled dataset and developing an integrated framework that first preprocesses raw sequences and then identifies crops similar to the unknown plant species that can assist agricultural scientists in enhancing seed quality. This framework helps in identifying unique special traits in unknown plant species such as rust resistance, drought resistance, etc., which can then be used to perform genetic engineering to improve those traits.

1.1 Background

Traditionally, alignment-based approaches are used to match each genome sequence with other genome sequences based on alignment techniques such as BLAST. However, as the length of each sequence and the number of sequences increase, these techniques become computationally intensive. These methods also face challenges when dealing with sequences of varying lengths, which can result in reduced clustering accuracy.

Therefore, Machine Learning (ML) methods are used that utilize alignment-free approaches to address these issues. The initial step to group or cluster similar genome sequences involves converting the genome sequence into a mathematical feature vector using a feature extraction technique. There are various types of real-life plant genome sequences, including- DNA/RNA, Protein and SNP sequences. Out of these sequences, SNP sequences help to categorise individuals into risk categories for diseases and predict treatment outcomes more reliably [1], [2]. These sequences comprise four different characters or nucleotides (A, T, G, C). The challenge in the feature extraction process, especially for SNP sequences, involves identifying an effective method to derive measurements that ensure similar genome sequences cluster together while distinct genome sequences fall into separate clusters. Extensive research has been conducted in this area to develop robust methods. Liu et al. [3] proposed a 12-dimensional approach, focusing solely on global representation. This method accounts for four nucleotides (A, T, G, C) by considering their counts (4 features), the sum of their indexes (4 features), and their distribution (4 features). Another approach was introduced by Kaisers et al. [4], known as the 2mer approach, which combines local and global representation by counting interactions between two nucleotides. This method generates a feature vector of size $16(4 * 4)$, but it suffers from the same issue of producing similar feature vectors for different sequences and lacks context-based features. Another method, also proposed by Kaisers et al. [4] known as the 3mer approach, aims to enhance the likelihood of identifying similar sequences by having the count of 3mers in the sequence. However, this approach exponentially increases the length of the feature vector to $64(4 * 4 * 4)$, leading to redundancy in the feature vector set. The above-mentioned

approaches for extracting features can extract similar features for distinct sequences, and those approaches also do not capture context-based information, so there is a need to extract features so that it can capture context-based information and different feature vector sets for different sequences. Additionally, these approaches are not scalable, i.e. these approaches for a very large dataset will take massive time to execute. Therefore, there is a need to innovate new algorithms to handle such huge, growing data.

Apart from extracting features of SNP sequences, the number of clusters is needed as input to perform clustering analysis. Many popular clustering algorithms such as K-means [5], K-medoids [6], FCM [7], etc., require specifying the cluster number beforehand. Therefore, the number of clusters is needed as input to perform clustering analysis. Various state-of-the-art techniques exist to find the optimal number of clusters. The traditional Elbow method [8], [9] and Silhouette method [10] have been commonly used to determine the optimal cluster number but are with limitations. The Silhouette method uses silhouette coefficients but may not always provide the best cluster number. The Elbow method's effectiveness relies on identifying clear elbow points in line charts, making it subjective and less reliable when dealing with smooth curves. The quantitative discriminant approach [11] is a recent method to identify the number of clusters, which is the automation of finding the elbow point using the elbow method without manual intervention. This approach helps overcome the Elbow method's ambiguity when the curve is smooth. The quantitative discriminant approach calculates the Euclidean distance between adjacent points in the dataset. This approach helps identify the optimal cluster number based on the minimum angle. However, the problem with these approaches to identify the optimal number of clusters is that the underlying algorithm used is the clustering algorithm to find the optimal number of clusters. Hence, in these cases, one must initially select a predetermined number of clusters to apply clustering and this initial choice should be evaluated for every cluster number to determine the optimal number of clusters using appropriate methods. Therefore, calculating the optimal number of clusters takes enormous time, and Big Data solutions are needed.

1.2 Motivation

Feature extraction is the initial step in clustering of the genome sequences to group similar sequences to identify the traits of the unknown SNP sequence. The problem with the current feature extraction techniques involves extracting similar features for distinct sequences; these approaches do not capture context-based information. Additionally, these approaches are not scalable, i.e., these approaches for a very large dataset will take an enormous amount of time to execute. The clustering algorithm requires an input of the number of clusters. As discussed in Section 1.1, various techniques are available to determine the optimal number of clusters. Still, the problem with the current approaches to find the optimal number of clusters is that the underlying algorithm used is the clustering algorithm to find the number of clusters. Hence, calculating the number of clusters takes enormous time.

Motivated by the success of Big Data frameworks, this thesis investigates a scalable feature extraction approach to handle large SNP sequences (Big data) based on the complex network [12] by distributing the sequences on various cores using Apache Spark Big Data processing framework [13] in which each worker node will take one sequence at a time to extract the features. The number of worker nodes equals the number of cores in the system. Furthermore, to investigate the optimal number of clusters, the S-MaxMin algorithm is proposed, which is a scalable algorithm to find the optimal number of clusters in the dataset using a distance-based, linear-time approach. This algorithm is made scalable using Apache Spark Big Data framework to reduce the time it takes for a large dataset to find the number of clusters.

1.3 Objectives

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

(i) To develop a novel scalable method to extract features from real-life plant genome SNP sequences based on a complex network theory that extracts a 21-dimensional mathematical feature vector and evaluates its performance by applying K-means

and FCM algorithms.

(ii) To develop a novel scalable algorithm for finding the optimal number of clusters in an unlabelled dataset and evaluate it on the benchmark datasets (whose number of classes are known) and the real-life plant SNP datasets (unlabeled) datasets.

(iii) To design a Scalable Integrated Framework for Genome Assembly and Special Trait Identification using the above two approaches.

1.4 Thesis Contributions

The significant contribution of the work done in this field is to design and develop feature extraction techniques for huge genome data and a method to find the optimal number of clusters for unlabelled datasets. These contributions are divided into two broad categories. Firstly, a scalable feature extraction technique is designed based on the complex network. Secondly, the design of a scalable algorithm to find the optimal number of clusters using a distance-based method, which is a linear-time approach.

A brief overview of our research contributions is provided below, and more details are available in the later chapters.

Contribution I: The current state-of-the-art feature extraction approaches can extract similar features for distinct sequences and do not capture context-based information. These approaches are not scalable, i.e. these approaches for a very large dataset will take enormous time to execute. Hence, a scalable algorithm for feature extraction is proposed to handle large SNP sequences (Big data) based on the complex network [12] by distributing the sequences on various cores using Apache Spark Big Data processing framework [13] in which each worker node will take one sequence at a time to extract the features. The number of worker nodes equals the number of cores in the system. The evaluation of the proposed feature vector set using clustering algorithms, K-means, and FCM shows that the proposed feature vectors perform well compared to other state-of-the-art approaches.

Contribution II: The problem with the current state-of-the-art approaches for find-

ing the optimal number of clusters is that the underlying algorithm in these approaches is the clustering algorithm to find the number of clusters. Hence, calculating the number of clusters takes enormous time. So, this thesis proposes a scalable algorithm to find the number of clusters in the dataset using a distance-based, linear-time approach. This algorithm is made scalable to reduce the time it takes for a large dataset to find the optimal number of clusters. Our approach is being tested on various benchmark and unlabelled real-life plant datasets. It gives the correct number of clusters for these datasets.

Contribution III: The plant breeders (Agriculture Scientists) use traditional methods to match the unknown sequences with many known sequences for identifying special traits, which assists agricultural scientists in enhancing seed quality. This requires extensive computation power and time as alignment-based approaches are used. Also integrated end-to-end solutions are currently unavailable in the market. Hence, different tools need to be used for each operation to find similar sequences of unknown species. Therefore, using the alignment-free method, a scalable integrated framework interface is developed that first preprocesses the raw sequence short reads and then identifies the crops similar to unknown plant species.

1.5 Organization of the Thesis

This thesis is organized into six chapters. A summary of each chapter is provided below:

Chapter 2 (Literature Survey)

This chapter discusses the current state-of-the-art feature extraction approaches used for genome SNP sequences. We also discuss the complex network theory for genome sequences, followed by different clustering algorithms. Then, we discussed the literature on finding the optimal number of clusters of the unlabelled dataset. We also discuss the big data framework adapted for our proposed feature extraction approach and proposed algorithm to find the optimal number of clusters. Finally, we discussed the performance measures and real-life plant genome SNP dataset used here.

Chapter 3 (A Scalable method for extracting features using a complex network from SNP sequences)

In this chapter, we discuss the proposed scalable algorithm to extract the features based on the complex network from real-life plant genome SNP sequences. The chapter finally reports the experimental evaluation that compares our proposed feature-extracted vectors with other state-of-the-art feature vectors using clustering algorithms. Also, to validate the superiority of our proposed method over other approaches, the non-parametric statistical measure, the Friedman test, is employed, demonstrating that our approach for feature extraction is better than other approaches.

Chapter 4 (A Scalable algorithm for finding the optimal number of clusters)

This chapter discusses a proposed scalable algorithm for finding the optimal number of clusters in an unlabelled dataset. The chapter finally reports the experimental evaluation of the proposed approach. This approach is being tested on various benchmark and unlabelled real-life plant datasets. It gives the correct number of clusters for these datasets. Also, sensitivity analysis is performed for the *threshold* hyperparameter.

Chapter 5 (Scalable Integrated Framework Interface for Genome Assembly and Special Trait Identification for Real-Life Crop data)

In this chapter, we discuss the design of an integrated framework interface for plant breeders to identify crops similar to unknown plant sequences using the proposed approaches.

Chapter 6 (Conclusion and Future work)

In this chapter, we conclude the work in our thesis and discuss the future directions of our research.

Chapter 2

Literature Survey

The chapter discusses the foundational concepts required to proceed to the proposed scalable feature extraction technique for real-life plant genome sequences and the proposed algorithm to find the optimal number of clusters. Section 2.1 refers to the various state-of-the-art approaches for extracting features from the genome SNP sequences. Furthermore, the features for the proposed complex network have been included in Section 2.2. Section 2.3 discusses various state-of-the-art clustering algorithms, followed by Section 2.4, which discusses the state-of-the-art methods for finding the optimal number of clusters. Additionally, Big Data frameworks discussed in Section 2.5 have been utilized to make our approaches scalable, followed by Section 2.6, which include various performance measures for assessing the proposed feature vectors. Finally, Section 2.7 discusses the Real-life plant genome SNP dataset.

2.1 Feature Extraction Techniques for SNP sequences

In this section, a detailed description of the existing state-of-the-art feature extraction techniques applied to SNP sequences is provided, serving as the basis for the subsequent discussion of the proposed feature extraction methods used for comparative analysis with the proposed approach.

2.1.1 12Dim Approach

In this, the authors have proposed the 12-dimensional feature vector for DNA sequences [3] and SNP sequences [14] based on the contents of four nucleotides (Adenine (A), Thymine (T), Guanine (G), and Cytosine (C)). The feature vectors comprise their distances from the origin and distribution along the sequences [3].

1. The first four features of the 12Dim approach consist of counting A, T, G, and C nucleotides. It will be represented as n_A, n_T, n_G, n_C.

2. The next four features of the 12Dim approach are the total distance of each nucleotide (A, T, G, C) to the first nucleotide, which is calculated using Eq. (2.1). From this feature, the authors have calculated the total distance of each nucleotide with respect to the first nucleotide.

$$T_i = \sum_{j=1}^{n_i} t_j, \tag{2.1}$$

   where, $i \in \{A, T, G, C\}$, t_j is distance from the first nucleotide i to the jth nucleotide i, n_i is total number of i^{th} nucleotide.

3. The last four features are related to the distribution of each nucleotide in the DNA sequence. The variance of distance for each nucleotide is the best parameter for the distribution and is calculated using Eq. (2.2).

$$D_i = \frac{\sum_{j=1}^{n_i}(t_j - \mu_i)}{n_i}, \tag{2.2}$$

   where $i \in \{A, T, G, C\}$, t_j is the distance from the nucleotide i to the j^{th} nucleotide i in the DNA sequence, $\mu_i = \frac{T_i}{n_i}$, where T_i is the total distance of a particular nucleotide i as discussed in Eq. (2.1). Thus, a feature vector obtained as
   $< n_A, n_T, n_G, n_C, T_A, T_G, T_T, T_C, D_A, D_T, D_G, D_C >$.

A limitation of the 12Dim Approach is that sequences that are not similar may

have identical total distance and distribution features for each nucleotide [15]. Hence, to tackle this problem, this thesis proposes a feature extraction approach.

2.1.2 2mer Approach

There is another approach to find the mathematical feature of the SNP sequences called a 2mer Approach. In this approach, the authors [4] have extracted the count of the 2-mers feature. 2-mers are substrings of length '2' contained within a biological sequence. The possible number of 2-mers is 16. For each 2-mer present in the SNP sequence, the authors counted the number of 2-mers, which represents a feature vector as: $< n_{AA}, n_{AT}, n_{AG}, n_{AC}, n_{TA}, n_{TT}, n_{TG}, n_{TC}, n_{GA}, n_{GT}, n_{GG}, n_{GG}, n_{CA}, n_{CT}, n_{CG}, n_{CC} >$

The drawback of the 2mer Approach is its inability to generate context-based features and its deficiency in biological interpretation. [16]. Hence, further enhancements are needed.

2.1.3 3mer Approach

One more commonly used approach to extract features from the SNP sequences is the 3mer approach. This approach is a word frequency-based alignment-free approach where the length of a word is three [4]. The rationale behind this method is similar sequences share similar words or 3mer, and mathematical operations with the word's frequency give a good relative measure of sequence dissimilarity. This approach is widely used in genome sequences [17]. There are 64 possible 3mers (64-dimensional feature vectors), and the count for each 3mer represents the feature vector, which represents the feature vectors as $< n_{AAA}, n_{AAT}, n_{AAG}, n_{AAC}, ...n_{CCA}, n_{CCT}, n_{CCG}, n_{CCC} >$.

One limitation of the 3mer Approach is its reliance on a high-dimensional feature space, which might introduce complexity and computational challenges. Additionally, this approach can not effectively capture context-based information, limiting its ability to represent and analyze data accurately.

These approaches can extract similar features for distinct sequences and do not capture context-based information. Additionally, these approaches are not scalable, i.e. these approaches for a very large dataset will take enormous time to execute. Thus, a scalable algorithm for feature extraction is proposed to handle huge SNP sequences (Big data) based on the complex network.

The proposed approach for feature extraction based on complex networks incorporates features derived from graph theory, which will be discussed in the subsequent section.

2.2 Mapping of features to Complex Network

Complex networks are widely used in mathematical modelling and have been an extremely active field in recent years [12]. The field of complex networks has its roots in graph theory and has applications in various domains, including genomics. In genome sequences, each node in a complex network represents a distinct k-mer, with 4^k nodes possible. The edge connecting the nodes represents the associativity between the two nodes, and each edge weight represents the frequency with which two k-mers occur consecutively in the genome sequence. This complex network is applied to our proposed work. Hence, this section discusses the features of the proposed approach based on the complex network theory [18], [19].

1. **Vertex Betweenness centrality** [20] is a measure used in network analysis to assess the importance of nodes within a complex network. It quantifies the extent to which a node lies on the shortest paths between other pairs of nodes in the network. The vertex betweenness centrality $BC(v)$ of a node $v \in V$ is calculated using the following Eq. (2.3).

$$BC(v) = \sum_{s,t \in V, s \neq v \neq t} \frac{\sigma(s,t|v)}{\sigma(s,t)}, \tag{2.3}$$

   where, $\sigma(s,t|v)$ is the number of shortest path from s to t through node v, and $\sigma(s,t)$ is the total number of shortest path from s to t. The mean of betweenness

centrality($MBC(v)$) for all nodes is calculated using the following Eq. (2.4).

$$MBC(v) = \frac{\sum_{\forall v \in V} BC(v)}{|V|}. \tag{2.4}$$

2. **Edge betweenness centrality** [21] is a measure that determines the importance of individual edges within a network. It quantifies how often an edge lies on the shortest paths between pairs of vertices in the network. To calculate edge betweenness centrality $BC(e)$ for an edge $e \in E$ is calculated using the following Eq. (2.5).

$$BC(e) = \sum_{s,t \in V, s \neq t} \frac{\sigma(s,t|e)}{\sigma(s,t)}, \tag{2.5}$$

where, $\sigma(s,t|e)$ is the number of shortest path from s to t through edge e, and $\sigma(s,t)$ is the total number of shortest path from s to t. The mean of edge betweenness centrality($MBC(e)$) for all edges in E is calculated using the following Eq. (2.6).

$$MBC(e) = \frac{\sum_{\forall e \in E} BC(e)}{|E|}. \tag{2.6}$$

3. **Assortativity** is also one of the most important features required as it tells about the correlation between two nodes. The assortativity coefficient is used, which is also the Pearson correlation coefficient of degree between pairs of linked nodes [22]. The r-value, which is the Assortavity coefficient, lies between -1 and 1 and is calculated using the following Eq. (2.7).

$$r = \frac{\sum_{jk} jk(e_{jk} - q_j * q_k)}{\sigma_p^2}, \tag{2.7}$$

where, and e_{jk} is the joint probability distribution of the remaining degrees of the two nodes with degrees j and k at either end of a randomly chosen edge, q_k is the distribution of the remaining degree of a node with degree k, and it is calculated using the following Eq. (2.8).

$$q_k = \frac{(k+1) * p_{k+1}}{\sum_j (j * p_j)}, \tag{2.8}$$

where p_k is the probability that a randomly chosen vertex on the graph will have degree k.

4. **Density of a graph** [23] is a measure that quantifies the compactness of the graph. It represents the ratio of the actual number of edges present in the graph to the maximum possible number of edges for that graph depicted in Eq. (2.9). Its value lies between 0 (the minimal density) and 1 (the maximal density).

$$D = \frac{2 * |E|}{|V| * (|V| - 1)}. \tag{2.9}$$

5. **Average degree connectivity** [20] denoted as $deg_{avg}$is a measure that tells the average connectivity among the neighbours of a node of the edges in a graph, and it is calculated using the following Eq. (2.10).

$$deg_{avg} = \frac{2 * |E|}{|V|}. \tag{2.10}$$

6. **Number of edges** is the total number of edges in a complex network after threshold application denoted as $|E|$.

7. **Network motifs of size 3** [20] are small, weakly connected induced subgraphs that appear more frequently in a real network than could be statistically expected. To find the motifs in a network, python $igraph\_motifs\_randesu()$ function is used, in which the input is given of the graph (adjacency list) and the size of motifs, i.e. 3 in this case. These Network motifs are represented as XYZ, where XYZ represents three nodes of network motifs.

After extracting features from the genome sequences, clustering needs to be applied. Hence, two state-of-the-art clustering algorithms are applied for comparative analysis, which will be discussed in the subsequent section.

2.3 Clustering Algorithms

This section will discuss two state-of-the-art clustering algorithms, K-means (Hard clustering) and FCM (Soft Clustering).

2.3.1 K-means Algorithm

The K-means algorithm is one of the state-of-the-art unsupervised algorithms used for clustering of datapoint [24]. In the K-means algorithm, the initial 'K' centroids are taken, where 'K' is the hyperparameter chosen by the user. Then, datapoints x_i in dataset X are assigned to each centroid, and for each cluster j, the mean of datapoints is found as

$$c_j = \frac{1}{|S_j|} \sum_{x_i \in S_j} x_i \tag{2.11}$$

where S_j is the set of datapoints assigned to cluster j, and c_j is the updated cluster. Assign the datapoints x_i to the new cluster. The above equation is repeated until the convergence condition is reached, i.e. the centroid position is not significantly changed.

The main objective of K-means is to minimize the objective function

$$J = \sum_{j=1}^{K} \sum_{x_i \in S_j} ||x_i - c_j||^2 \tag{2.12}$$

This equation represents to minimize the total within-cluster variance.

2.3.2 FCM Algorithm

FCM, also known as soft clustering, is also a state-of-the-art methodology for clustering, introduced by [7].In FCM, each datapoint x_i is assigned to each cluster with some membership values, and the objective is to minimize the following function.

$$J_{FCM} = \sum_{i=1}^{n} \sum_{j=1}^{K} (u_{ij})^m ||x_i - c_j||^2 \tag{2.13}$$

where K is the hyper-parameter chosen by the user based on some prior information for the dataset, u_{ij} is the membership value of datapoint x_i associated with cluster j, $m > 1$ is the fuzzification parameter, c_j is the cluster centroid, and n is the number of datapoints.

The problem with these clustering algorithms is that they need an input of the number of clusters, which is user-specific. There are various state-of-the-art techniques to determine the optimal number of clusters discussed in the subsequent subsection.

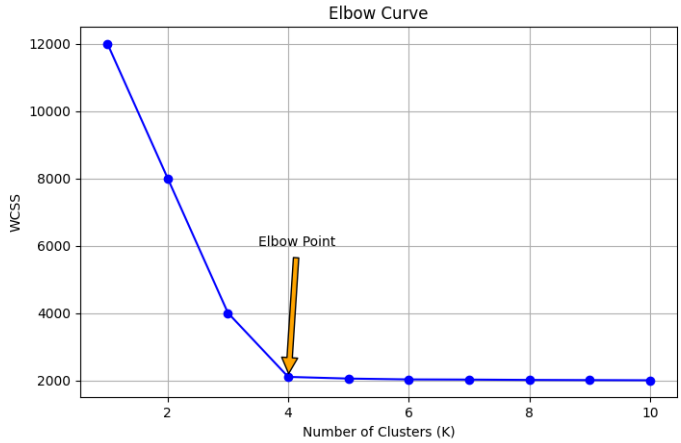
2.4 Methodologies to find the optimal number of clusters

This section describes some methodologies used to determine the optimal number of clusters, including the elbow, the silhouette, and the quantitative discriminant method, which are discussed below.

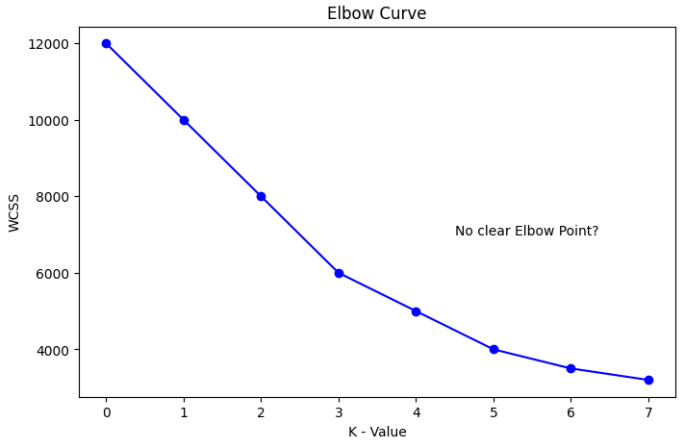
2.4.1 Elbow method

The Elbow method [25], which is one of the earliest techniques used to determine the optimal number of clusters in a dataset, involves iteratively increasing the number of clusters from an initial value of K (usually 2) until a plateau is reached where the rate of decrease in cost slows down significantly. The cost here refers to the within-cluster sum of square (WCSS), i.e. the sum of the square distance between points in a cluster and the cluster centroid. This starting point in the plateau is often referred to as the "elbow point", as depicted in Fig. 2.1a. The optimal number of clusters is then determined based on this elbow point.

However, this method has the disadvantage of identifying the elbow point manually, which becomes ambiguous when the plotted curve is smooth, as illustrated in Fig. 2.1b. The elbow point may vary depending on the analyst's judgment.



(a) Visible Elbow Point



(b) Not Visible Elbow Point

Figure 2.1: Visualisation for different datasets to find the optimal number of clusters (a) Clear view to find elbow point (b) No Clear Elbow Point

2.4.2 Silhouette method

Another method to determine the optimal number of clusters is the Silhouette method [10]. It is another well-known method with decent performance to estimate the potential optimal cluster number. It uses the average distance between one data point and others in the same cluster and the average distance among different clusters to score the clustering result.

This method's scoring metric is named the Silhouette Index coefficient (SI), and SI is defined using Eqn. 2.14.

$$SI = \frac{b - a}{\max(a, b)}, \tag{2.14}$$

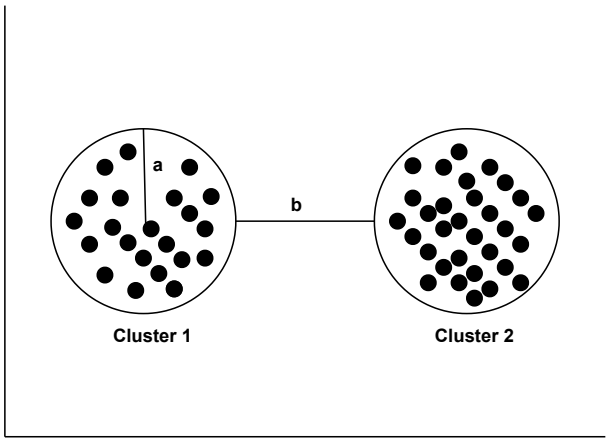


Figure 2.2: a: Intra-Cluster distance, b: Inter-Cluster distance

where 'a' represents the mean intra-cluster distance and 'b' denotes the mean inter-cluster distance. Fig. 2.2 visualises the inter-cluster distance 'b' and intra-cluster distance 'a'.

The interval of the SI values is $-1 \leq SI \leq 1$. A value of S closer to 1 indicates that a sample is better clustered, and if it is closer to -1, the sample should be categorized into another cluster. This method is preferable for estimating the potential optimal cluster number. Meanwhile, the silhouette index can evaluate the best number of clusters in most cases for many distinct scenarios.

2.4.3 Quantitative Discriminant Method

To automate the finding of the elbow without manual intervention, the quantitative discriminant method [11] is used. This algorithm computes the Euclidean distance between three adjacent points and represents them as 'i', 'j', and 'k'. The angle formed by every three adjacent two-dimensional data point pairs α_j is computed using the Eqn. 2.15.

$$\alpha_j = arccos \frac{E_{ij}^2 + E_{jk}^2 + E_{ik}^2}{2E_{ij}^2 E_{jk}^2}, \tag{2.15}$$

where,

$$E_{ij} = \sqrt{(n_i - n_j)^2 + (k_i - k_j)^2}. \tag{2.16}$$

n_i and k_i represent the first dimension and the second dimension of the two-dimensional data point, respectively. The minimal angle is found, and the index of the optimal cluster number (K_{opt}) is determined. The quantitative discriminant method exploits the interaction angle of the adjacent elbow point as a criterion to determine the discriminant elbow point.

The problem with these approaches is that they will assume some cluster numbers and run the clustering algorithms to find the optimal number of clusters. Also, the problem with these approaches is that the underlying algorithm they use is the clustering algorithm to determine the optimal number of clusters. This thesis proposes a linear time approach to find the number of clusters in the dataset using a distance-based approach.

The drawback with the approaches for feature extraction discussed in Section 2.1 and finding the optimal number of clusters discussed in 2.4 is that the algorithms take enormous time to compute features and find the optimal number of clusters for a huge dataset. So, our proposed approach applies the Apache Spark-based Big Data framework, which is discussed in detail in the subsequent subsection.

2.5 Apache Spark Based Big Data Framework

The era of genome sequencing commenced significantly with initiatives like the Human Genome Project (HGP) [26], which aimed to sequence the entire human genome. Launched in 1990 and completed in 2003, the HGP marked a pivotal moment in biotechnology, providing invaluable insights into human genetics. However, the project also underscored the massive challenge of managing, analyzing, and storing extensive genetic data. The initial sequencing of the human genome took 13 years and cost approximately $3 billion, highlighting the need for more efficient computational solutions to handle such large-scale data. The advent of next-generation sequencing (NGS) [27] technologies dramatically changed the landscape of genomic research by increasing the speed and reducing the cost of sequencing. Technologies such as Illumina's sequencing platforms [28] can generate terabytes of data in a single run, far surpassing the data

output from earlier methods. This surge in data production necessitated a corresponding development in data handling capacities; thus, the field of bioinformatics had to evolve rapidly.

Before the growth of big data technologies, biotechnologists relied on localized and often insufficient computational resources, leading to data processing and analysis bottlenecks [29]. The challenge is not just the volume of data but also its complexity. Genomic data consists of long sequences of nucleotides, and deciphering the functional implications of these sequences requires sophisticated computational algorithms and substantial processing power [30].

Big data frameworks such as Apache Hadoop [31], Apache Spark [32], Apache Hive [33], and others have been developed to manage and process these vast and diverse data troves. In the context of genomic data analysis, the utilization of Apache Spark becomes particularly advantageous due to its ability to handle the complexities and scale of genomic datasets efficiently. Genomic data analysis involves processing vast amounts of genetic sequences, which requires robust computational resources and optimized algorithms for accurate analysis. Apache Spark's distributed computing model allows for parallel processing of genomic data, significantly reducing processing times and enabling researchers to derive insights from large-scale datasets in a timely manner.

Fig. 2.3 shows the overview of the Apache Spark cluster. Apache Spark cluster consists of one master node and several worker nodes or executor nodes. The master node is a driver, which is used for task scheduling. Spark sequentially initiates a scheduling process with jobs, steps, and tasks. The step is a subset of tasks partitioned from collective jobs, which is used to match the map and reduce phase. An executor is created for each program for each worker node. The executor runs the tasks and caches the data in memory or disk [34].

This thesis uses the Apache Spark cluster to compute feature vectors for SNP sequences using a complex network. The approach for feature extraction is made scalable using Apache Spark Big Data framework to handle vast amounts of genomic sequences so that each CPU core will take one sequence at a time and extract features. The

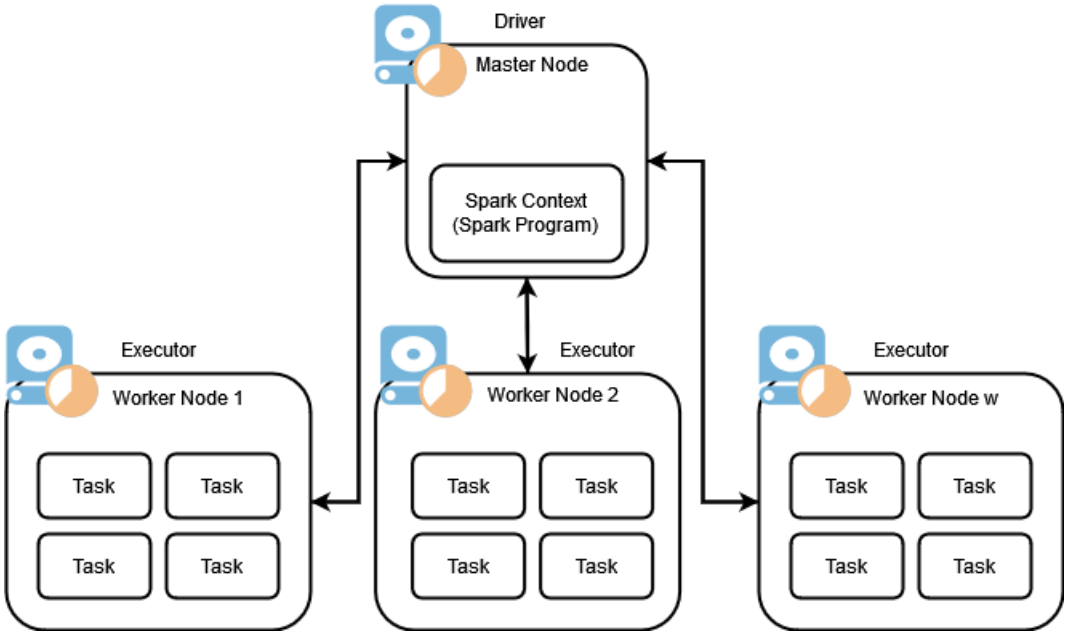


Figure 2.3: Apache Spark Architecture

proposed S-MaxMin algorithm is also integrated with the Apache Spark framework to find the optimal number of clusters.

For the clustering analysis for the huge SNP sequence datasets, various evaluation metrics are used to evaluate the performance of the feature extraction approaches. Evaluation metrics such as the Silhouette Index [35] and Calinski-Harabasz Index [36] are used to assess the performance of the feature extraction approach using several clustering techniques. The details of these measures are presented in the next subsection.

2.6 Performance Measures

To evaluate the performance of the proposed feature extraction approach on unlabeled SNP datasets using clustering analysis, two internal evaluation measures are used, the Silhouette Index and the Calinski-Harabasz Index, which is discussed subsequently in this subsection.

1. **Silhouette Index (SI)**: SI [35] quantifies how well-separated the clusters are compared to how closely the data points within each cluster are grouped to-

gether. It is calculated using Eq. (2.17).

$$SI(i) = \frac{b_i - a_i}{\max(a_i, b_i)}, \tag{2.17}$$

where a_i is average intra-cluster distance of datapoint i, and b_i is average inter-cluster distance of datapoint i with the nearest cluster. The SI ranges from -1 to 1. To calculate the SI for the entire dataset, average the SI of all data points, which is depicted in Eq. (2.18).

$$SI = \frac{1}{n} \sum_{i=1}^{n} SI(i). \tag{2.18}$$

2. **Calinski-Harabasz Index (CH Index)**: CH index [36] for N data points and K clusters for dataset $X \in \{x_1, x_2, ..., x_N\}$ is computed using Eq. (2.19).

$$CH = \frac{\sum_{k=1}^{K} n_k ||c_k - c||^2}{\sum_{k=1}^{K} \sum_{i=1}^{n_k} ||x_i - c_k||^2} * \frac{N - K}{K - 1}, \tag{2.19}$$

where, n_k is the number of datapoints in cluster k, c_k is the centroid of the k^{th} cluster, and c is the global centroid.

The real-life plant genome SNP dataset used to analyse the proposed approach for extracting features based on the complex network is discussed in the next section.

2.7 Real life Plant Genome SNP dataset

This section will discuss the SNP datasets used in the analysis. SNP datasets are employed to validate our proposed scalable feature extraction technique, and an algorithm is developed to determine the optimal number of clusters.

1. **Modified Wm82 a2 SNP50K dataset with JS-335** [37]: The comparison is performed with the complete genome sequence of JS-335 to William82 Assembly 4 Genomic sequence (100x coverage). After modification, the SNP dataset for JS-335 is 0.9GB in size and contains 24,632 data samples and 13,903 SNPs.

2. **Wm82 a1 SNP50K dataset** [38]: The whole dataset has a size of 1.7GB, with 20,087 data samples and 42,509 SNPs.

3. **MAGIC-rice** [39]: This dataset consists of 16,932 SNP sequences, which are separated into 12 separate files for each of the 12 chromosomes (1-12). For feature extraction and clustering, all the chromosomes are combined to create the MAGIC-rice SNP dataset.

4. **248Entries Rice** [40]: The 248Entries Rice dataset consist 248 data samples. The size of the dataset is 30.8 MB.

The dataset details are shown in Table 2.1.

Parameters	**Datasets**			
	Modified Wm82 a2 SNP50K	**Wm82 a1 SNP50K**	**MAGIC-rice**	**248Entries Rice**
Number of samples	24,632	20,087	16,932	248
SNP Length	13,903	42,509	53,374	40,840
Size	0.9GB	1.7GB	1.05GB	30.8MB

Table 2.1: Dataset Description

Chapter 3

A Scalable method for extracting features using a complex network from SNP sequences

This chapter proposes a scalable feature extraction approach based on the complex network for a huge SNP sequence dataset. A scalable 21-dimensional feature extraction method is proposed, which extracts features using a complex network model [12]. The first section of the chapter discusses the conversion of SNP sequences into complex networks, as outlined in Section 3.1. The second section addresses the extraction of features from the complex network using the proposed scalable feature extraction technique, detailed in Section 3.2. This is followed by two more sections for evaluating the proposed approaches to extract features mentioned in Section 3.3 and Section 3.4.

3.1 Mapping SNP sequences to Complex Network

To convert the SNP sequences into a complex network, firstly, the extraction of consecutive substrings of size k from the SNP sequences is performed, commonly referred to as kmers, and then proceed to create nodes. Each node in a complex network of SNP sequences represents a distinct kmers, with 4^k nodes possible. The value of k for kmers is set to three, as shown in Fig. 3.1 because k less than three would have a smaller number of nodes (1mer or 2mer), which leads to higher weights for

each edge in the complex network. Therefore, applying the threshold on the complex network would not alter its overall structure. Therefore, an identical collection of seven feature vector sets can be inferred. Also, when the k increases, more nodes are formed, resulting in lower edge weights and a larger likelihood of graph disconnection. Therefore, it is determined via experimentation that the ideal value for the threshold is 3, and the obtained results support this observation. Therefore, the total number of nodes in the network will equal 64, which is derived from the calculation $(4 * 4 * 4)$.

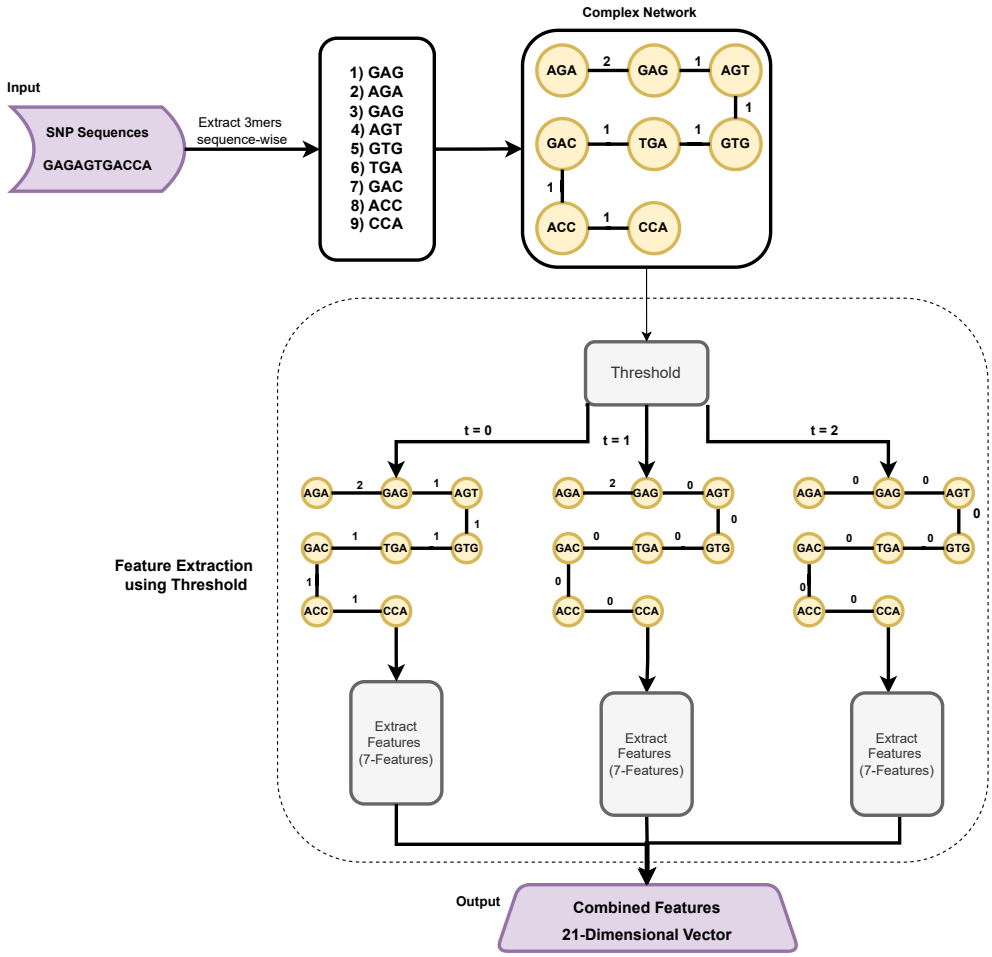


Figure 3.1: Proposed Complex Network for SNP sequence

An edge is established whenever two 3mers occur consecutively, with the weight of the edge set to one. Subsequently, when two 3mers occur consecutively again, the weight of the edge is incremented by 1. The edge connecting the nodes represents

the associativity between the two nodes, and each edge weight represents the frequency with which two 3mers occur consecutively in the SNP sequence. This graph is collectively known as a complex network.

The features from complex networks from different resolutions will be extracted using thresholds. The approach will take an enormous amount of time to compute features for huge amounts of SNP sequences. Therefore, a scalable approach to extract features using thresholds for complex networks is proposed, which will be discussed in detail in the subsequent section.

3.2 Proposed Scalable Feature Extraction of SNP sequences using Complex Network

The proposed feature extraction process based on the complex network is shown in Fig. 3.1. The feature extraction is conducted at various resolutions of a complex network using the threshold. In each iteration from 0 to (threshold-1), when the threshold is set to 0, the algorithm considers the entire complex network and subsequently extracts features. For threshold values of 1 and above, the algorithm removes edges with weights equal to or less than the threshold value and then extracts features.

The threshold for this proposed approach using a complex network is set to three because increasing the threshold would result in a larger number of features, many of which would be zero. Conversely, setting the threshold below three would not adequately capture the various aspects of the complex network. The same is verified by performing extensive experiments.

In this proposed feature extraction technique, seven independent features are extracted, as discussed in Chapter 2, Section 2.2, extracting features at different resolutions for threshold equals three, resulting in a 21-dimensional feature vector. The features include centrality measures (Eqn. 2.4, Eqn. 2.6), assortativity (Eqn. 2.7), density (Eqn. 2.9), average degree (Eqn. 2.10), total edges, and network motifs. Other features, as described in [12], are not included, such as the average shortest

path length, maximum degree and minimum degree. The reason for not including the average shortest path length is the possibility of low edge weights in the graph, which can lead to infinite path lengths when attempting to find the shortest path between unreachable nodes. Also, maximum and minimum degrees have not been considered because the average degree, which inherently accounts for the highest and lowest values, has already been included in the feature vector.

Thus, the proposed feature vector is represented as follows:

$< MBC(v)_{th=0}, MBC(e)_{th=0}, r_{th=0}, D_{th=0}, deg_{avg_{th=0}}, |E|_{th=0}, XYZ_{th=0}, MBC(v)_{th=1}, MBC(e)_{th=1}, r_{th=1}, D_{th=1}, deg_{avg_{th=1}}, |E|_{th=1}, XYZ_{th=1}, MBC(v)_{th=2}, MBC(e)_{th=2}, r_{th=2}, D_{th=2}, deg_{avg_{th=2}}, |E|_{th=2}, XYZ_{th=2} >$

After extracting these seven independent features at threshold = 3, 21-dimensional feature vectors are obtained, as shown in Fig. 3.1. The algorithm for extracting these features is described in Algorithm 1. This algorithm describes converting the SNP sequence into the complex network in lines $1-4$. Line $5, 6$ extracts the seven features for each threshold value $(0, 1, 2)$.

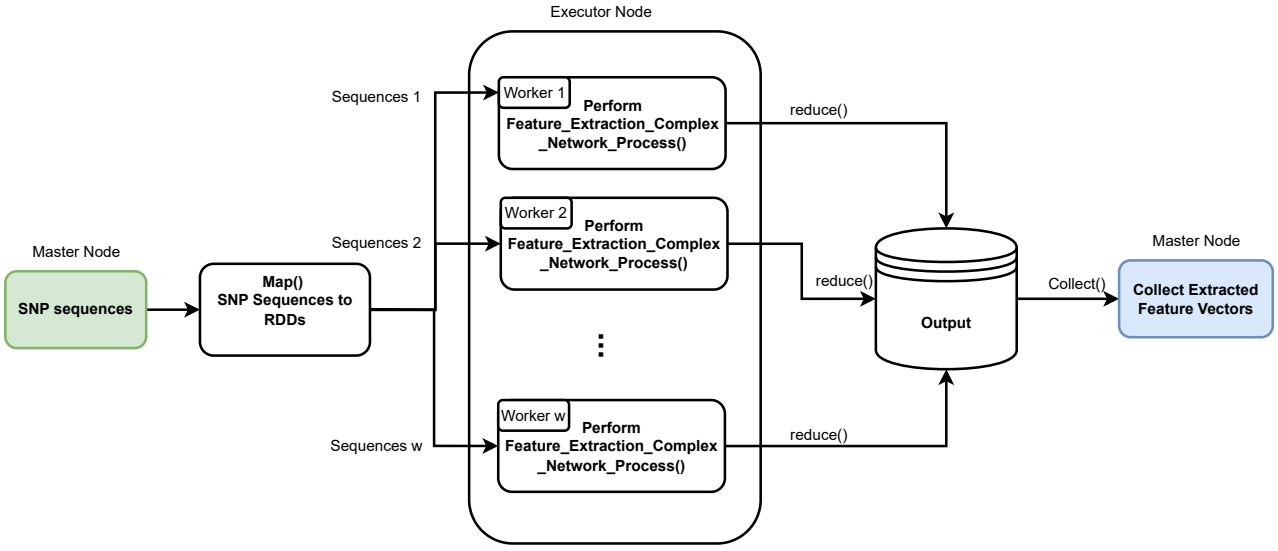


Figure 3.2: Apache Spark Cluster for Feature Extraction

However, extracting these mathematical feature vectors from complex networks takes enormous time. Hence, a scalable algorithm that integrates with the Apache Spark framework is proposed to extract the proposed feature vector set from a large

Algorithm 1: *Feature_Extraction_Complex_Network_Process*()

Data: SNP sequences A, T, G, C

Result: Extracted Feature Vectors

1: Initialise an empty graph G using an adjacency list for complex network
2: Read the sequence from input and store it in string *seq*
3: Generate k-mers of size k from the sequence, by breaking down the *seq* into k length substrings, and store it in *kmer*
4: **for** $i \leftarrow 0$ *to* $kmer.size() - 1$ **do**
    Add kmer[i] and kmer[i+1] to the Graph G
    if $kmer[i]$ *and* $kmer[i+1]$ *is present in Graph* G **then**
        Increase edge weight size by one
    end
    else
        Add an edge between $kmer[i]$ and $kmer[i+1]$
    end
    end
5: Initialise a threshold value th (Hyper-parameter)
    `/* Apply threshold scheme */`
6: **for** $i \leftarrow 0$ *to* $th - 1$ **do**
    for *each edge* e *in* G *if edge_weight* $\leq$ *threshold* **do**
        $edge\_weight[e] \leftarrow 0$
    end
    if $num\_of\_edges \leq 1$ **then**
        break
    end
    Extract features from the graph as mentioned and store it in a file
    end

SNP dataset, as illustrated in Fig. 3.2. Large amounts of SNP sequences are initially partitioned into w chunks utilizing Resilient Distributed Datasets (RDDs). Subsequently, each chunk is assigned to a distinct worker node, parallelizing the extraction process of features as shown in the Algorithm 2. In this algorithm, line 1 reads the SNP sequences from the file and then distributes each sequence to various cores to perform distributed computing on executor nodes using the Map function in Apache Spark. This extraction is performed in parallel on multiple cores, and then the feature vectors are collected using the Collect function in Apache Spark. The flow of Algorithm 2 in the Apache Spark Cluster is shown in Fig. 3.2.

A comprehensive discussion of the proposed scalable feature extraction technique

Algorithm 2: Scalable Feature Extraction using complex network

Data: SNP sequences (A, T, G, C)
Result: Extracted Feature Vectors
1: **Read** the SNP sequence from the file
2: **Map** each sequence to RDDs
3: **Perform** $Feature\_Extraction\_Complex\_Network\_Process()$ algorithm for each sequence stored in RDDs.
4: **Collect** the feature vectors and store them in a file.

in terms of time complexity is discussed in the subsequent section.

3.3 Complexity Analysis

To calculate $MBC(v)$ in complex networks for vertex and edge, first, betweenness centrality at each node must be calculated according to Eq. 2.5. Calculating $BC(v)$ in one node for undirected graphs takes $O(V + E)$ time. To calculate the $BC(v)$ for each node, time complexity will be $O(V * (V + E))$; hence, summing all $BC(v)$ and averaging it will result in $MBC(v)$ according to Eq. 2.6.

To calculate assortativity, the degrees of each node need to be calculated, which takes $O(E)$ time, and assortativity needs to be calculated using Eq. 2.7. The sum of products of degrees of connected nodes, the sum of degrees, and the sum of the square of degrees need to be calculated, which takes $O(E)$ time in total, and then applying the Pearson correlation formula, which takes $O(1)$ time. Hence, calculating assortativity requires $O(E)$ time. To calculate the density of the graph, average degree connectivity and number of edges from the complex network, it takes $O(E)$ time to compute. Calculating network motifs of size 3 includes enumerating triplets, checking for connectivity and counting motif occurrences, which takes time complexity of $O(V^3)$.

Overall, the time required to calculate the feature vector for any type of graph (sparse or dense) requires a time complexity of $O(V(V + E) + E + V^3) \approx O(V^3)$ and if there are n genome sequences, then the time required will be $O(nV^3)$. The approach to extract the features is made scalable according to Algorithm 2 to reduce

time complexity. The time complexity to find the extracted feature for n genome sequences is reduced to $O(nV^3/w)$, where w is the number of worker nodes.

Further in Section 3.4, the experimental evaluation of the proposed feature extraction technique is presented.

3.4 Experimental Evaluation

In this thesis, four unlabelled plant genome SNP datasets are utilized. To evaluate the performance of the proposed approach, two state-of-the-art clustering techniques have been employed: K-means [24] and FCM [7] to assess the effectiveness of the proposed approach independent of the learning models. This section describes the parameters used for the clustering techniques. Subsequently, we discuss the experimental results on various SNP datasets and the runtime analysis at each core, followed by statistical analysis for feature extraction techniques. Our investigation is conducted in parallel environments with multiple cores using the Apache Spark environment. The Precision 5820 tower is utilized, with features including an Intel Xeon W-2102 @2.90GHz chipset, a 4-core processor, and 64GiB of RAM.

3.4.1 Parameter Specification

The parameter specification is described in Table 3.1. In the FCM algorithm, a fuzzification parameter of 1.75 is used, and the experiment is conducted for values of c ranging from 2 to 10. During our experiment, K-means and FCM algorithms are executed for 10 epochs. Furthermore, no scaler has been employed to standardize the properties of the data points.

Algorithms	**Parameter**	**Values**
FCM	Fuzzification Parameter (m)	1.75
	Epochs	10
	Number of clusters (c)	2-10
K-means	Epochs	10
	Number of clusters (K)	2-10

Table 3.1: Parameter Specification

In the next subsection, the comparative analysis of the proposed feature extraction technique will be discussed with other state-of-the-art feature extraction approaches.

3.4.2 Evaluation of Feature Extraction Approaches

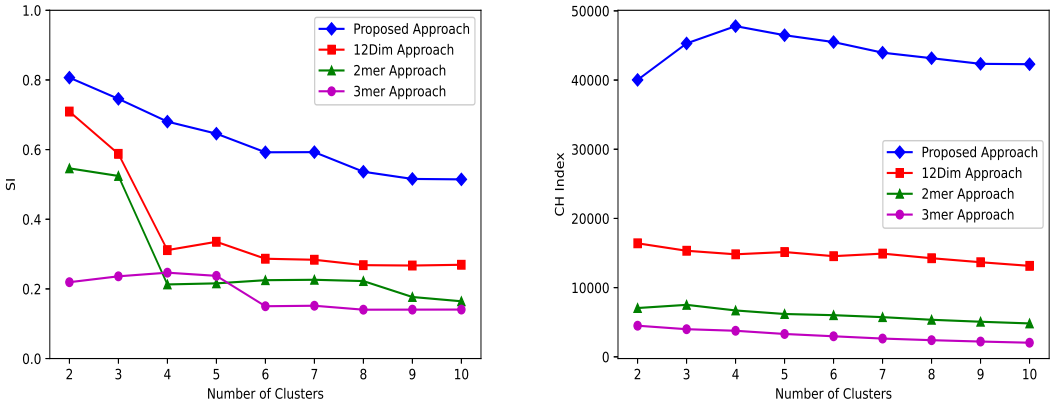
This section will discuss feature extraction results based on the complex network for real-life genome SNP datasets, evaluated on different clustering algorithms.

1. **Modified Wm82 a2 SNP50K**

   The clustering analysis has been conducted for clusters ranging from 2 to 10 for both the proposed and existing approaches on the Modified Wm82 a2 SNP50K dataset. From Table 3.2 and Fig. 3.3 shown for K-means, the proposed complex network approach exhibits the best evaluation results in terms of SI and CH Index, while the 3mer Approach demonstrates the worst-case performance.

Table 3.2: Results on Modified Wm82 a2 using K-Means

Modified Wm82 a2 SNP50K	Clusters	Proposed Approach	12Dim Approach [14]	2mer Approach [4]	3mer Approach [4]
SI	2	**0.80663**	0.70920	0.54613	0.21924
	3	**0.74572**	0.58773	0.52433	0.23613
	4	**0.68030**	0.31141	0.21282	0.24658
	5	**0.64573**	0.33525	0.21589	0.23747
	6	**0.59228**	0.28658	0.22500	0.15022
	7	**0.59270**	0.28366	0.22619	0.15164
	8	**0.53643**	0.26817	0.22247	0.14030
	9	**0.51576**	0.26709	0.17674	0.14047
	10	**0.51452**	0.26926	0.16445	0.14067
CH Index	2	**40022.25911**	16406.27757	7051.38113	4496.69638
	3	**45295.31967**	15328.06741	7511.71364	3991.55615
	4	**47794.29026**	14815.90659	6700.76759	3765.07412
	5	**46480.73833**	15144.20967	6189.16412	3301.93535
	6	**45491.7203**9	14547.38830	6016.94377	2968.71516
	7	**43958.36382**	14915.84123	5733.84789	2638.59477
	8	**43159.86706**	14251.58124	5353.65978	2404.43479
	9	**42343.89566**	13671.72616	5069.50606	2207.44730
	10	**42291.02284**	13144.87532	4825.91154	2037.83201



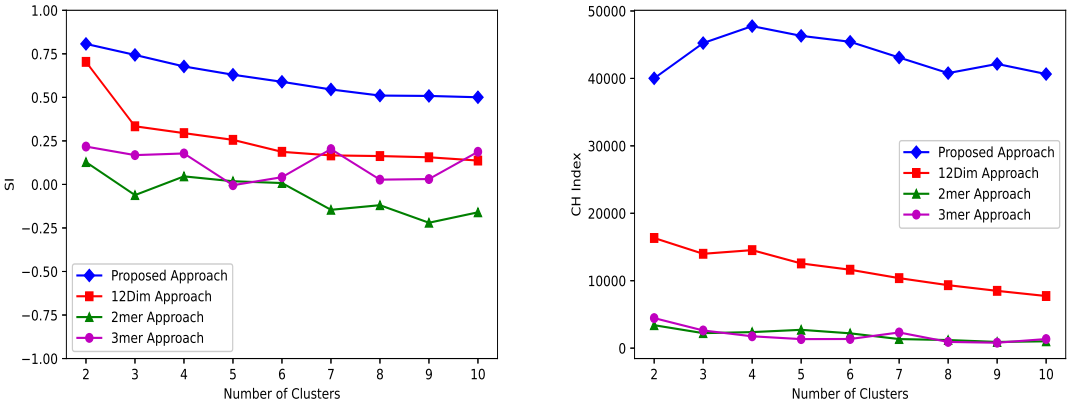
(a) SI (b) CH Index

Figure 3.3: Results on Modified Wm82 a2 using K-Means

In Table 3.3 and Fig. 3.4 have shown below the performance metrics for Modified Wm82 a2 SNP50K using FCM clustering, the worst performance is given by the 2mer Approach, and the best results for our proposed complex network approach are observed in terms of SI and CH Index.

Table 3.3: Results on Modified Wm82 a2 using FCM

Modified Wm82 a2 SNP50K	Clusters	Proposed Approach	12Dim approach [14]	2mer Approach [4]	3mer Approach [4]
SI	2	**0.80697**	0.70377	0.12823	0.21738
	3	**0.74372**	0.33387	-0.06135	0.16829
	4	**0.67750**	0.29464	0.04619	0.17779
	5	**0.62999**	0.25593	0.01827	-0.00448
	6	**0.58931**	0.18716	0.00795	0.04152
	7	**0.54542**	0.16667	-0.14578	0.20287
	8	**0.51030**	0.16284	-0.11929	0.02741
	9	**0.50812**	0.15586	-0.22005	0.03082
	10	**0.50055**	0.13711	-0.16016	0.18742
CH Index	2	**40015.63628**	16336.49812	3403.75069	4456.38513
	3	**45240.69982**	13992.14851	2224.91629	2622.64298
	4	**47753.77058**	14535.95513	2378.74481	1758.19667
	5	**46311.73112**	12563.46722	2717.87768	1338.82998
	6	**45432.17984**	11628.37240	2196.39417	1357.71508
	7	**43099.81612**	10382.58927	1346.37465	2324.64139
	8	**40786.47328**	9337.757580	1201.08359	923.646570
	9	**42145.56527**	8496.130710	921.499979	804.833701
	10	**40645.73674**	7728.933380	1013.18683	1348.29502



(a) SI (b) CH Index

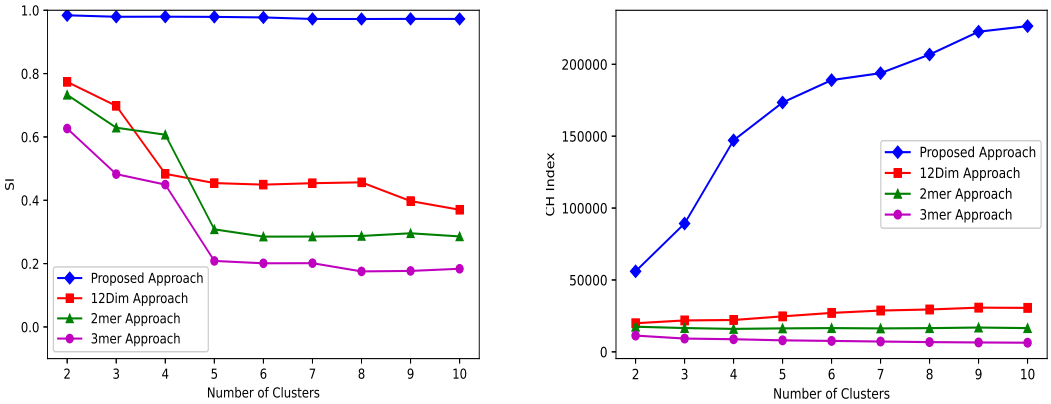
Figure 3.4: Results on Modified Wm82 a2 using FCM

2. **Wm82 a1 SNP50K**

Using the K-means algorithm on the Wm82 a1 SNP50K dataset, the experimentation is performed on cluster sizes ranging from 2 to 10 for both the proposed and existing approaches, as depicted in Table 3.4 and Fig. 3.5. Our proposed approach achieved the highest SI and CH Index values compared to all existing approaches. Conversely, the 3mer Approach exhibited the poorest performance among the evaluated approaches in the context of K-means algorithm evaluation.

Table 3.4: Results on Wm82 a1 using K-means

Wm82 a1 SNP50K	Clusters	Proposed Approach	12Dim approach [14]	2mer Approach [4]	3mer Approach [4]
SI	2	**0.98407**	0.77372	0.73301	0.62650
	3	**0.97942**	0.69807	0.62920	0.48258
	4	**0.97969**	0.48347	0.60695	0.44948
	5	**0.97908**	0.45427	0.30820	0.20844
	6	**0.97733**	0.44936	0.28520	0.20095
	7	**0.97249**	0.45388	0.28544	0.20135
	8	**0.97237**	0.45660	0.28730	0.17535
	9	**0.97276**	0.39753	0.29573	0.17688
	10	**0.97261**	0.37002	0.28584	0.18368
CH Index	2	**56014.048120**	19894.24250	17482.12644	11307.7316
	3	**89244.878420**	21848.31730	16549.21204	9244.12873
	4	**147112.98850**	22156.18382	15965.47840	8774.85659
	5	**173423.92680**	24678.73758	16324.04092	8002.45238
	6	**189002.63640**	27076.03005	16521.23506	7604.19289
	7	**193834.37010**	28731.81992	16304.51976	7177.15371
	8	**206775.97140**	29458.16256	16495.57486	6777.31670
	9	**222669.13200**	30743.75744	16908.44574	6528.59008
	10	**226542.25830**	30568.53089	16509.76687	6348.53524



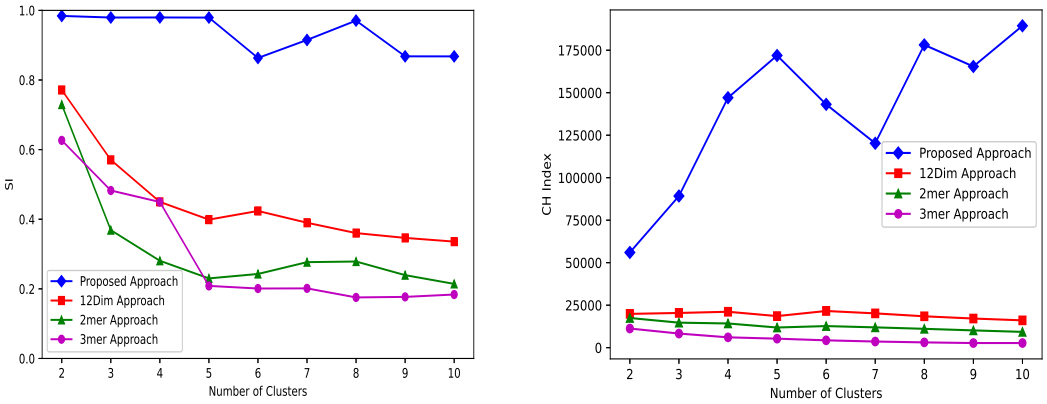
(a) SI (b) CH Index

Figure 3.5: Results on Wm82 a1 using K-Means

Also, using FCM, the proposed approach demonstrated superior SI and CH Index values compared to all existing approaches in evaluating performance as illustrated in Table 3.5 and Fig. 3.6. Conversely, the 3mer Approach exhibited the least favourable performance among the evaluated approaches.

Table 3.5: Results on Wm82 a1 using FCM

Wm82 a1 SNP50K	Clusters	Proposed Approach	12Dim approach [14]	2mer Approach [4]	3mer Approach [4]
SI	2	**0.98407**	0.77133	0.72942	0.61653
	3	**0.97945**	0.57046	0.36872	0.18769
	4	**0.97966**	0.44980	0.28066	0.05892
	5	**0.97930**	0.39888	0.23001	0.03057
	6	**0.86328**	0.42375	0.24274	0.00768
	7	**0.91505**	0.38990	0.27669	-0.01607
	8	**0.97064**	0.36018	0.27852	-0.02851
	9	**0.86796**	0.34636	0.23938	-0.05831
	10	**0.86780**	0.33568	0.21425	-0.02099
CH Index	2	**56014.048120**	19880.52861	17463.48720	11271.3346
	3	**89216.938470**	20428.92002	14719.25590	8313.00504
	4	**147064.03080**	21133.85603	14245.21231	6089.98935
	5	**171889.31170**	18585.83388	11866.58599	5303.69164
	6	**143139.42670**	21585.24809	12708.67562	4322.58233
	7	**120259.17900**	20179.39214	11950.93982	3620.72011
	8	**178140.28420**	18464.48670	11097.72922	3114.15083
	9	**165428.55720**	17147.54362	10166.32590	2729.92688
	10	**189355.15080**	16076.75183	9293.912839	2732.22616



(a) SI (b) CH Index

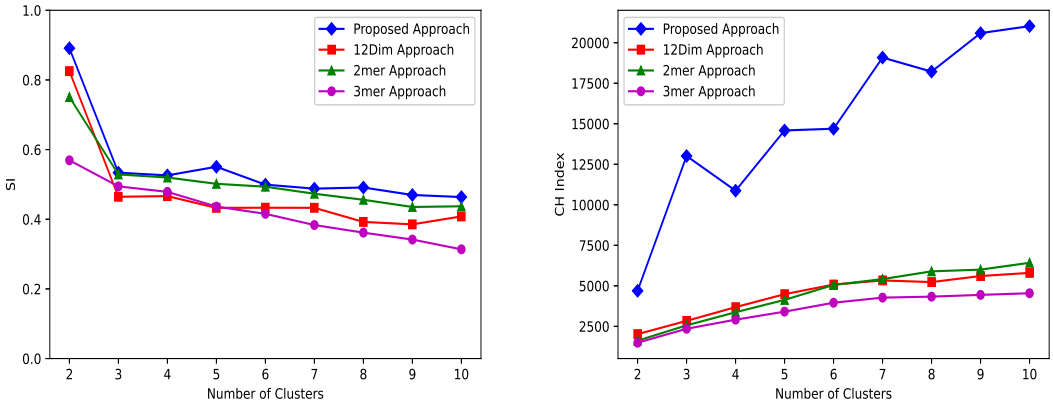
Figure 3.6: Results on Wm82 a1 using FCM

3. **MAGIC-rice**

Our analysis for K-means clustering, ranging the clusters from $2-10$, is shown in Table 3.6 and Fig. 3.7 for the MAGIC-rice dataset. It's evident that our proposed approach outperforms other approaches in both the SI and CH Index and the 3mer Approach is performing the worst.

Table 3.6: Results on MAGIC-rice using K-Means

MAGIC-rice	Clusters	Proposed Approach	12Dim approach [14]	2mer Approach [4]	3mer Approach [4]
SI	2	**0.89111**	0.82538	0.75094	0.56925
	3	**0.53370**	0.46454	0.52873	0.49444
	4	**0.51598**	0.46640	0.51984	0.47858
	5	**0.55066**	0.43270	0.50173	0.43639
	6	**0.48034**	0.43310	0.49350	0.41552
	7	**0.48788**	0.43282	0.47324	0.38343
	8	**0.49119**	0.39236	0.45589	0.36156
	9	**0.46969**	0.38534	0.43526	0.34167
	10	**0.46373**	0.40813	0.43721	0.31361
CH Index	2	**4692.315560**	2020.21120	1617.28028	1489.60467
	3	**13010.92688**	2837.55544	2560.84455	2351.08813
	4	**10870.07150**	3684.18462	3369.77156	2911.51802
	5	**14587.10056**	4484.81783	4132.40201	3405.94725
	6	**14693.50429**	5077.29506	5054.04251	3957.58019
	7	**19084.63866**	5333.32790	5414.30004	4272.32192
	8	**18216.73292**	5230.77522	5894.16196	4332.44539
	9	**20591.61157**	5604.08935	5998.99074	4440.86519
	10	**21024.15202**	5798.96697	6421.67297	4543.07258



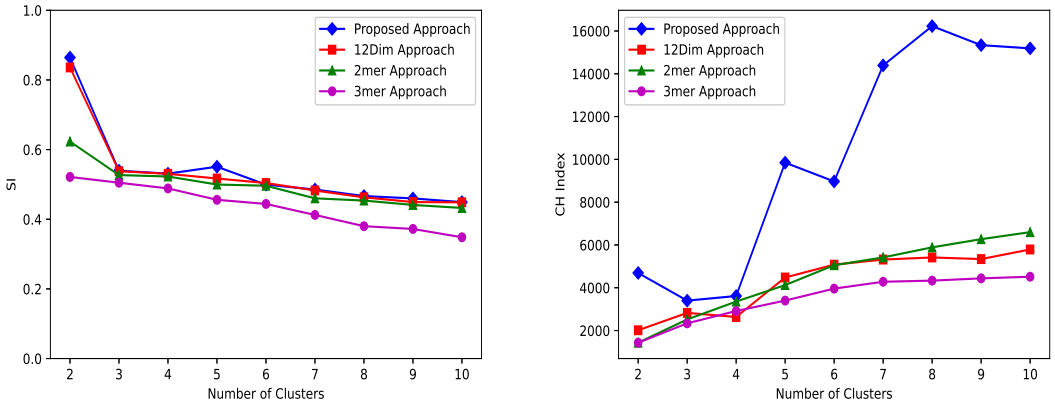
(a) SI (b) CH Index

Figure 3.7: Results on MAGIC-rice using K-Means

Similarly, using FCM clustering analysis for MAGIC-rice, as shown in Table 3.7 and Fig. 3.8, our proposed approach performs better than other approaches. Additionally, the 3mer Approach exhibits the worst performance in terms of SI and CH Index.

Table 3.7: Results on MAGIC-rice using FCM

MAGIC-rice	Clusters	Proposed Approach	12Dim approach [14]	2mer Approach [4]	3mer Approach [4]
SI	2	**0.86429**	0.83586	0.62311	0.52149
	3	**0.53943**	0.53796	0.52679	0.50491
	4	**0.53090**	0.53041	0.52277	0.48855
	5	**0.55087**	0.51702	0.49969	0.45584
	6	**0.49940**	0.50398	0.49621	0.44397
	7	**0.48533**	0.48261	0.46003	0.41241
	8	**0.46656**	0.46294	0.45366	0.38007
	9	**0.45981**	0.44935	0.44110	0.37213
	10	**0.44913**	0.44859	0.43244	0.34823
CH Index	2	**4691.045630**	2012.85510	1431.99432	1430.62501
	3	**3397.596320**	2825.43260	2517.53865	2337.10037
	4	**3613.817130**	2634.11364	3353.65404	2912.37031
	5	**9846.849070**	4478.37392	4123.86775	3400.15004
	6	**8968.912550**	5078.26118	5053.04886	3959.18188
	7	**14392.44882**	5317.18338	5417.06770	4279.52028
	8	**16227.49702**	5415.60043	5887.91594	4332.63008
	9	**15338.37041**	5339.88722	6270.29250	4438.83314
	10	**15188.17753**	5789.04110	6598.19679	4514.95895



SI CH Index

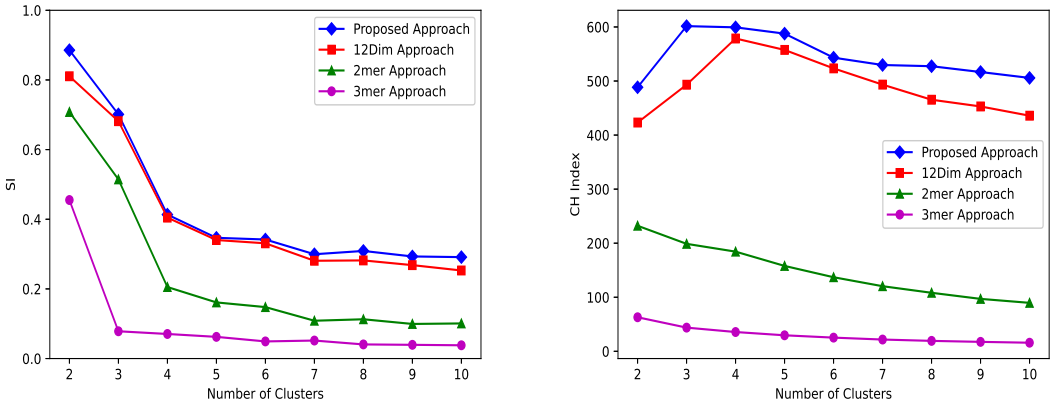
Figure 3.8: Results on MAGIC-rice using FCM

4. **248Entries Rice**

In the 248Entries Rice dataset, while evaluating with clusters ranging from $2-10$, our proposed extracted features using the complex network on the K-means algorithm are giving better results as compared to the other approaches (refer Table 3.8 and Fig. 3.9).

Table 3.8: Results on 248Entries Rice using K-Means

248Entries Rice	Clusters	Proposed Approach	12Dim approach [14]	2mer Approach [4]	3mer Approach [4]
SI	2	**0.88551**	0.81061	0.70799	0.45521
	3	**0.70117**	0.68160	0.51446	0.07833
	4	**0.40497**	0.40468	0.20535	0.07055
	5	**0.34667**	0.34033	0.16118	0.06227
	6	**0.34184**	0.33092	0.14774	0.04917
	7	**0.29950**	0.28079	0.10858	0.05168
	8	**0.30898**	0.28179	0.11278	0.04056
	9	**0.29317**	0.26825	0.09935	0.03939
	10	**0.29123**	0.25288	0.10082	0.03809
CH Index	2	**488.41158**	423.18917	232.25495	62.97499
	3	**601.54927**	493.13256	198.79548	43.80500
	4	**599.30224**	578.66184	184.37341	35.66217
	5	**587.66020**	557.53169	157.94633	29.60674
	6	**543.21454**	523.40501	137.02295	25.30050
	7	**529.62586**	493.33566	120.42573	21.75028
	8	**527.38309**	465.38500	108.24468	19.29177
	9	**516.66197**	452.95120	96.966936	17.46108
	10	**505.72734**	435.83754	89.626142	15.87541



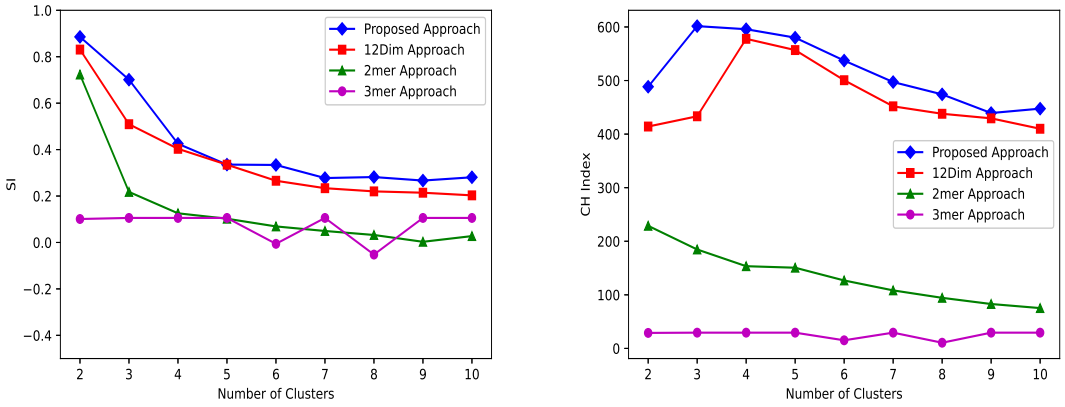
(a) SI (b) CH Index

Figure 3.9: Results on 248Entries Rice using K-Means

Similarly, when our proposed approach is evaluated on FCM, it gives better results than other approaches in terms of both SI and CH Index (refer Table 3.9 and Fig. 4.5).

Table 3.9: Results on 248Entries Rice using FCM

248Entries Rice	Clusters	Proposed Approach	12Dim approach [14]	2mer Approach [4]	3mer Approach [4]
SI	2	**0.88551**	0.83078	0.72327	0.10106
	3	**0.70117**	0.50938	0.21774	0.10566
	4	**0.42515**	0.40335	0.12561	0.10566
	5	**0.33539**	0.33495	0.10221	0.10566
	6	**0.33383**	0.26577	0.06910	-0.00586
	7	**0.27741**	0.23375	0.04947	0.10566
	8	**0.28175**	0.22003	0.03224	-0.05217
	9	**0.26662**	0.21425	0.00286	0.10566
	10	**0.28076**	0.20290	0.02743	0.10566
CH Index	2	**488.41156**	414.14023	228.83399	28.92225
	3	**601.54927**	433.27628	184.66718	29.39383
	4	**595.79717**	577.87258	153.44290	29.39383
	5	**580.05830**	556.84342	150.61909	29.39383
	6	**537.26952**	500.60561	126.82166	15.13109
	7	**497.17070**	451.83690	108.29620	29.39383
	8	**474.15067**	437.95842	94.397050	10.61397
	9	**439.32546**	429.43141	82.865580	29.39383
	10	**447.52045**	409.96726	75.249230	29.39383



(a) SI (b) CH Index

Figure 3.10: Results on 248Entries Rice using FCM

Overall, the values presented in Table 3.2 and Table 3.3 for Modified Wm 82 a2 dataset, Table 3.4 and Table 3.5 for Wm 82 a1 dataset, Table 3.6 and Table 3.7 for MAGIC-rice, and Table 3.8 and Table 3.9 for 248Entries Rice represent the results obtained from the experiment of our proposed feature vector set on K-means and FCM clustering, respectively, with different performance evaluation metrics (SI and CH Index). The comparison is performed with different state-of-the-art approaches (12Dim Approach, 2mer Approach, and 3mer Approach) to varying values of K (ranging from 2 to 10).

Here, the value of SI indicates how well the clusters are well-separated using inter-cluster distance and intra-cluster distance ranging values from [-1 to 1]. A SI value close to 1 means the object is well-matched to its cluster and poorly matched to neighbouring clusters. This suggests a good separation between clusters. The SI value close to 0 indicates that the object is on or close to the decision boundary between two neighbouring clusters. SI value close to -1 suggests that the object may have been assigned to the wrong cluster. It is more similar to neighbouring clusters than to its cluster. Additionally, the CH Index compares the ratio of inter-cluster dispersion to intra-cluster dispersion for different clustering solutions. The CH index values range from 0 to positive infinity.

So, SI values close to +1 are preferred to justify how better the clustering is performed, and a higher CH value suggests a better separation between clusters.

Our proposed approach for extracting features based on complex networks has been made scalable, and the computation time has been significantly reduced by integrating the Apache Spark Big Data framework. The core-wise runtime computation for each dataset is presented in the next subsection.

3.4.3 Runtime Analysis for proposed Feature Extraction Approach

The time required to extract features from real-life plant SNP datasets is calculated to demonstrate scalability, as shown in Table 3.10. There is a significant difference in the time required to extract the features of complex networks on a single core versus multiple cores. Therefore, more executor nodes with multiple cores can be utilized to reduce time.

Table 3.10: Runtime for feature extraction using complex network on different cores using Apache spark

Dataset	**Time (hh:mm:ss)**			
	1 core	**2 core**	**3 core**	**4 core**
Modified Wm82 a2 SNP50K	07:58:08	06:01:14	03:49:28	02:31:59
Wm82 a1 SNP50K	18:54:32	13:01:52	09:48:15	05:39:09
MAGIC-rice	32:03:12	25:24:03	16:36:51	08:54:27
248Entries Rice	00:22:57	00:17:46	00:11:03	00:06:15

To statistically prove that our proposed approach is superior to other state-of-the-art approaches, a non-parametric test, the Friedman test, is performed. To evaluate which approach is performing better, the Nemenyi posthoc test is conducted, which is discussed in detail in the next subsection.

3.4.4 Statistical Analysis of Experiment Results of Feature Extraction Techniques

In this analysis, SI values are initially extracted from Tables 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, and 3.9, specifically focusing on the row associated with a cluster value of 2. This selection is made due to the observation that, for K = 2, higher SI values are obtained in comparison to other cluster values, and is shown in Table 3.11. Subsequently, statistical analysis is conducted on these extracted values utilizing the Friedman test.

The Friedman test is a non-parametric test in which it ranks the algorithm for each dataset separately, and the best-performing algorithm gets the rank of 1, the second best rank of 2, ..., and so on. In the case of ties, average ranks are assigned. This assigning of rank is shown in Table 3.12 for Table 3.11. This test assesses whether the compared methods are different. If they are, the Nemenyi post-hoc test is utilized to identify their differences.

Table 3.11: Different datasets with K-means and FCM values taken from cluster 2 compared with four different methods of feature extraction

Datasets	**Silhouette Index(SI) Scores**			
	Proposed Approach	**12Dim Approach**	**2mer Approach**	**3mer Approach**
Modified Wm82 a2 (K-means)	**0.80663**	0.7092	0.54613	0.21924
Modified Wm82 a2 (FCM)	**0.80697**	0.70377	0.12823	0.21738
Wm82 a1 (K-means)	**0.98407**	0.77372	0.73301	0.6265
Wm82 a1 (FCM)	**0.98407**	0.77133	0.72942	0.61653
MAGIC-rice (K-means)	**0.89111**	0.82538	0.75094	0.56925
MAGIC-rice (FCM)	**0.86429**	0.83586	0.62311	0.52149
248Entries (K-means)	**0.88551**	0.81061	0.70799	0.45521
248Entries (FCM)	**0.88551**	0.83078	0.72327	0.10106

The Friedman test with the corresponding post-hoc test is used for the 4 algorithms across 8 datasets (4 different datasets, each containing 2 different values). Here, equality among all methods is assumed under the null hypothesis. The Friedman statistic is computed for the ranks on SI values from Table 3.12 as

Table 3.12: Rank comparison based on the highest of all the SI on all clusters (Table 3.11) of the proposed approach for feature extraction, 12Dim Approach, 2mer Approach, 3mer Approach

Datasets	Proposed Approach	12Dim Approach	2mer Approach	3mer Approach
Modified Wm82 a2 (K-means)	**1**	2	3	4
Modified Wm82 a2 (FCM)	**1**	2	4	3
Wm82 a1 (K-means)	**1**	2	3	4
Wm82 a1 (FCM)	**1**	2	3	4
MAGIC-rice (K-means)	**1**	2	3	4
MAGIC-rice (FCM)	**1**	2	3	4
248Entries (K-means)	**1**	2	3	4
248Entries (FCM)	**1**	2	3	4
AVERAGE RANK	**1**	2	3.125	3.875

$$\chi^2 = \frac{12N}{k(k+1)} \left(\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4} \right) \tag{3.1}$$

where χ^2 is the Friedman test statistic, N is the total number of observations (in this case $N = 8$), k is the number of methods ($k = 4$), and R_j is the sum of ranks for methods j.

$\chi^2 = \frac{12*8}{4*(4+1)}((1.000)^2 + (2.000)^2$

$+(3.125)^2 + (3.875)^2 - \frac{4\times(4+1)^2}{4} \approx \mathbf{22.95}$

Now, F_F of the F-distribution will be calculated using

$$F_F = \frac{(N-1)\chi^2}{N(k-1) - \chi^2}, \tag{3.2}$$

which equals $\approx$ **3511.35**, with degree of freedom $((k-1), (k-1)(N-1))$ which is **(3,21)**. The critical value of $F(3, 21)$ is 3.028 for the significance level at $\alpha = 0.05$. The null hypothesis is rejected since the value of $F_F = 3511.35 > 3.028$. Further, the Nemenyi posthoc test is performed for pairwise comparison of methods and is verified by computing the critical difference (CD) at $p = 0.10$, which is computed using, $CD = q_\alpha \sqrt{\frac{k(k+1)}{6N}}$ which $CD = 1.17263$.

The difference between the average ranks of 12Dim approach, 2mer Approach, 3mer Approach with the proposed approach are $(2.000 - 1.000)$, $(3.125 - 1.000)$, $(3.875 - 1.000)$, respectively. The majority of the differences are greater than CD(1.17263), so it can be concluded that our proposed approach is significantly better than the other compared approaches for clustering.

3.5 Summary

In this chapter, the scalable alignment-free feature extraction approach based on the complex network for huge SNP sequences was proposed. This was done by distributing the SNP sequences on various cores using the Apache Spark Big Data processing framework. Each worker node will extract the features one sequence at a time. To assess the effectiveness of the proposed feature vector set for SNP sequences, state-of-the-art clustering algorithms, K-means and Fuzzy c-means (FCM), were utilized to group similar sequences. The proposed approach was assessed on four real-life plant genome SNP datasets viz. Modified Wm82 a2 SNP50K, Wm82 a1 SNP50K, MAGIC-rice and 248Entries Rice. To assess the performance, comparison criteria such as inter-cluster and intra-cluster distances were utilized, using the Silhouette Index (SI) and Calinski-Harabasz (CH) Index for each dataset. Our experimental analysis demonstrates that our proposed approach surpasses in terms of SI and CH Index from the existing state-of-the-art feature extraction techniques for each dataset.

Also, to statistically verify that our proposed approach for feature extraction was better than the other approaches, the Friedman test was used, followed by the Nemenyi post-test. It was found that the majority of the algorithm's rank differences from the average rank of our proposed approach are greater than the critical difference. Hence, our proposed approach for feature extraction performs better statistically.

Chapter 4

A Scalable Algorithm for finding the optimal number of clusters

In the previous chapter, mathematical feature vectors based on a complex network are found for real-life genome SNP sequences. A clustering analysis then needs to be performed to identify similar sequence clusters. However, clustering analysis requires the number of clusters as input. Therefore, this chapter proposes an algorithm called the Max of Min, which uses a distance-based, linear-time approach to find the optimal number of clusters. Section 4.1 describes the Max of Min algorithm concept. Then, the parallelization of the Max of Min algorithm is discussed in Section 4.2, i.e., the S-MaxMin Algorithm, to quickly determine the optimal number of clusters for a large input vector. Further in Section 4.3, the complexity of the proposed algorithm is discussed, followed by experimental analysis in Section 4.4. Finally, to decide the hyperparameter value of the proposed algorithm, sensitivity analysis is discussed in Section 4.5.

4.1 Design of Proposed Max of Min Algorithm

The Max of Min algorithm is designed to find the optimal number of clusters using the distance metric on the partitioned dataset. The principle of the Max of Min algorithm is as follows: Let the dataset X with n data points be defined as follows: $X = \{x_1, x_2, ..., x_n\}$. The centroids for each cluster formed are defined as $\{c_1, c_2, ..., c_k\}$,

where k is the number of clusters that can be formed using this algorithm, and c_i is the chosen data point to be cluster centre. This c_i is never updated as this is only used to find the number of clusters. In theory, each cluster should contain data points very near each other. This algorithm computed the distances between every data point x_i and cluster centre c_j using the Euclidean distance represented as $||x_i - c_j||$. Let d_{max} represent the maximum distance between cluster centres (inter-cluster distance). From each cluster, find the farthest point x_{f_i}, and compute the distance from cluster centre x_{c_i} and x_{f_i}. Let $v = ||x_{c_i} - x_{f_i}||$.

A threshold logic will be used to determine whether to include the next cluster centre, which is:

$$\frac{v}{d_{max}} >= threshold \tag{4.1}$$

where the value of the threshold will range from 0 to 1. $\frac{v}{d_{max}}$, this value will always be in the range from 0 to 1, as v indicates intra-cluster distance, d_{max} represents inter-cluster distance, and intra-cluster distance should always be less than or equal to the inter-cluster distance.

The proposed algorithm is described in Algorithm 3. Line 1 of this algorithm selects a random data point, assumed to be a cluster centre, and identifies a data point farthest from the chosen cluster centre. Subsequently, each data point is assigned to its corresponding cluster based on the minimum distance in line 2. Starting from line 3, the algorithm iteratively determines the farthest data point within each cluster, denoted as x_{f_i}, having a distance v farthest compared to all cluster centres. In line 4, the algorithm computes d_{max}, representing the maximum inter-cluster distance. Based on Eq. (4.1), the algorithm assesses whether x_{f_i} may be a new potential cluster centre to form a new cluster.

The acceleration of this algorithm is possible using parallel computation, particularly in the phase where the cluster to which each datapoint belongs is determined. This aspect can be parallelized, and a detailed discussion is provided in the next section.

Algorithm 3: Max of Min Algorithm

Data: $X \in \{x_1, x_2, x_3, ...x_n\}$
Result: k : number of clusters
`/* Initialise(To find two cluster centres` c_1 `and` c_2`) */`

1: Choose a data point randomly from X call it x_{c1}. Suppose $x_{c2} \in X$ is the farthest point wrt x_{c1}. x_{c1} and x_{c2} are the centre of cluster c_1 and c_2, respectively ;
2: **for** $i \leftarrow 1$ *to* n **do**
    if $||x_i - x_{c1}||$ *is minimum* **then**
        Assign x_i to c_1 ;
    end
    else if $||x_i - x_{c2}||$ *is minimum* **then**
        Assign x_i to c_2 ;
    end
    end
    `/* Generate more clusters if possible */`
3: Let $c_1, c_2, ..., c_k$ be clusters, and let x_{fi} be the farthest point in cluster c_i with respect to cluster centre x_{ci}
4: Let d_{max} is the largest distance between the cluster centres $c = \{x_{c1}, x_{c2}, ...x_{ck}\}$
5: $v = max(||x_{f1} - x_{c1}||, ||x_{f2} - x_{c2}||, ...||x_{fk} - x_{ck}||)$
6: Let $||x_{fi} - x_{ci}||$ be maximum, i.e., $v = ||x_{fi} - x_{ci}||$;
7: **if** $\frac{v}{d_{max}} \geq threshold$ **then**
    Include x_{fi} as c_{k+1} cluster;
    for $i \leftarrow 1$ *to* n **do**
        $u = min(||x_i - x_{c1}||, ||x_i - x_{c2}||, ..., ||x_i - x_{ck}||)$;
        if $||x_i - x_{c1}||$ *is minimum* **then**
            $x_i \leftarrow c_1$;
        end
    end
    end
8: **else**
    Output k
    end

4.2 Design of Proposed Scalable Max of Min Algorithm using Apache Spark: S-MaxMin Algorithm

The bottleneck in the Max of Min algorithm occurs during each iteration when the distance between each data point and all cluster centres is calculated as shown in

Algorithm 3 in lines 3-4. However, this process may be optimized by parallelizing it, allowing for simultaneously determining the minimal distance value for each data point. The map and reduce-by-key function will be employed in Apache Spark's framework. Initially, the process involves creating Resilient Distributed Datasets (RDDs) for the data points. Subsequently, these RDDs are subjected to mapping operations to determine the minimum distance, as shown in Fig. 4.1. Finally, the maximum distance is obtained using the reduce-by-key function on the computed minimum distances. This method of integrating Apache Spark considerably reduces the execution time of the proposed algorithm on large input vectors.

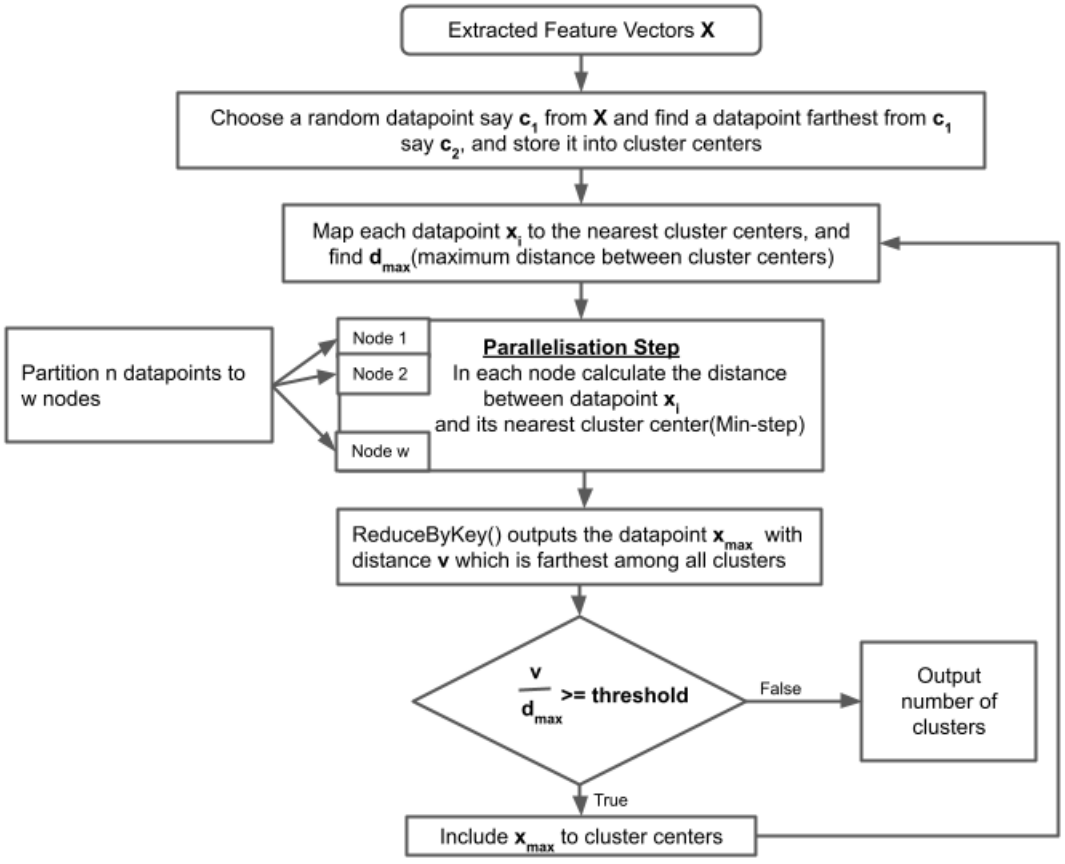


Figure 4.1: Workflow of S-MaxMin Algorithm

A comprehensive discussion of time complexity analysis for the proposed Max of Min algorithm and S-MaxMin algorithm is discussed in the subsequent section.

4.3 Complexity Analysis

The Max of Min algorithm incrementally forms k clusters, where the determination of the value of k relies on the dataset's inherent structure. For each iteration, all the data points need to be assigned to their respective clusters. Therefore, the time complexity of the proposed algorithm Max of Min is $O(n.k)$, with k representing the number of clusters output by the proposed algorithm.

In the S-MaxMin algorithm, each datapoint assignment to the cluster is parallelised by integrating the Apache Spark Big Data framework into the algorithm. Therefore, the time complexity for the S-MaxMin algorithm becomes $O(n.k/w)$, where w is the number of worker nodes.

The evaluation of the results obtained from the proposed Max of Min algorithm on benchmark and real-life plant genome datasets is discussed in detail in the subsequent section.

4.4 Experimental Evaluation on Benchmark and Real life datasets

The S-MaxMin Algorithm is executed on various benchmark and real-life datasets to find the optimal number of clusters. This proposed algorithm has one hyperparameter *threshold*, which is set to be 0.5 for these experiments.

A. **Benchmark Datasets**

The benchmark datasets have executed the algorithm on eight benchmark datasets, including Iris [41], Parkinsons [41], PIDD (downloaded from `https://www.kaggle.com/uciml/pima-indians-diabetes-database`), Wine [41], WDBC [41], Diabetes [41], Vehicle Silhouettes [41], Customer Churn datasets (downloaded from `https://www.kaggle.com/datasets/hassanamin/customer-churn`), as shown in Table 4.1, producing the correct number of clusters when compared to the actual number of classes.

Table 4.1: Results on Benchmark Datasets

Datasets	Number of features	Actual number of classes	S-MaxMin cluster prediction
Iris	4	3	3
Parkinsons	23	2	2
PIDD	9	2	2
Wine	13	3	3
WDBC	30	2	2
Diabetes	8	2	2
Vehicle Silhouettes	18	3	3
Customer Churn	16	2	2

B. **Real life Datasets**

The approach is implemented on real-life datasets, and the resulting number of clusters is presented in Table 4.2. A characteristic indicating the cluster number where a high SI is observed is generated, followed by a comparative analysis of our findings. Upon executing the S-MaxMin algorithm for the Wm82 a1 dataset, it is observed that the resulting number of clusters is 3. However, the maximum SI obtained is 2. This finding is consistent with the information in the Table for the Wm82 a1 dataset, where both the K-means and FCM algorithms exhibit similar SI for clusters 2 and 3. Therefore, when the number of clusters is set to 3, the clustering performance is likewise deemed satisfactory. Furthermore, the algorithm yielded accurate findings for the output for the remaining datasets.

Table 4.2: Results on Real-life Datasets

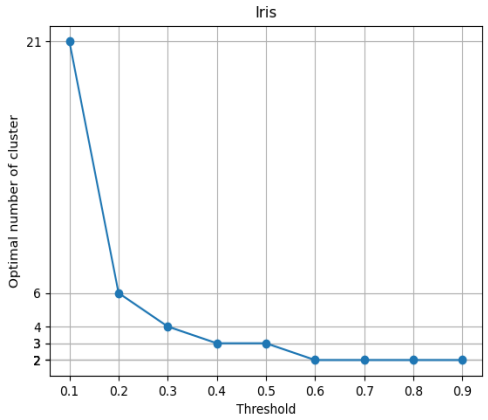
Proposed Approach Feature Set	**S-MaxMin cluster Prediction**	**Cluster number with Max SI**
Modified Wm82 a2	2	2
Wm82 a1	3	2
MagicRice	2	2
248Entries Rice	2	2

In this proposed algorithm, the value of $\frac{v}{d_{max}}$ will range between 0 and 1 because v represents the intra-cluster distance and d_{max} represents inter-cluster distance, and $v \leq d_{max}$. Therefore, its value lies between 0 and 1. In our case, the *threshold* has been set to 0.5 after performing a sensitivity analysis on the *threshold* parameter, which will be discussed in the next section.

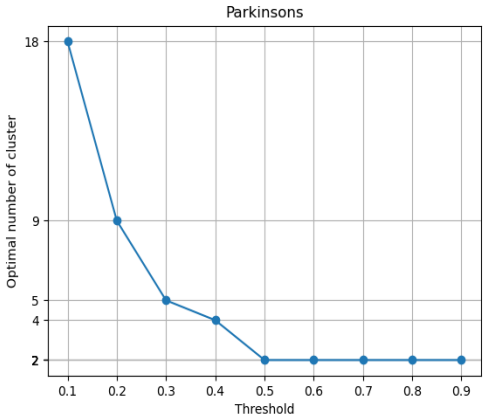
4.5 Parameter Sensitivity and Calibration Analysis for S-MaxMin Algorithm

To identify the *threshold* value for the S-MaxMin algorithm, sensitivity analysis is conducted on different threshold values (ranging $[0-1]$) on eight benchmark datasets, as shown in Fig. 4.2a (Iris), Fig. 4.2b (Parkinsons), Fig. 4.3a (PIDD), Fig. 4.3b (Wine), Fig. 4.4a (WDBC), Fig. 4.4b (Diabetes), Fig. 4.5a (Vehicle Silhouettes) and Fig. 4.5b (Customer Churn). When the *threshold* is set to 0.5 for each dataset, the optimal number of clusters is consistently obtained for the benchmark dataset, where the actual number of classes is known. Hence, the *threshold* value is set to 0.5.

For the unlabelled real-life plant SNP datasets, in the proposed S-MaxMin algorithm, the *threshold* parameter is calibrated to identify the optimal number of clusters. Clustering analysis is performed for each dataset across various ranges of K values $(2-10)$. Specifically, for the Modified Wm82 a2 SNP50K dataset, the results are presented in Tables 3.2 and 3.3; for the Wm82 a1 SNP50K dataset, in Tables 3.4 and 3.5; for the MAGIC-rice dataset, in Tables 3.6 and 3.7; and for the 248Entries Rice dataset, in Tables 3.8 and 3.9. Subsequently, internal validation metrics such as SI and CH Index are calculated for each clustering result. This calibration process determined the optimal number of clusters for the unlabelled real-life plant SNP dataset, prioritizing those with high SI Scores, as shown in Table 4.2. The validation of our proposed S-MaxMin algorithm yields nearly identical results across datasets except for the Wm82 a1 dataset. This observation aligns with the data presented in Table 3.4 and 3.5 for the Wm82 a1 dataset, where both the K-means and FCM algorithms demonstrate similar SI values for clusters 2 and 3. Therefore, when the number of clusters is set to 3, the clustering performance is likewise deemed satisfactory.

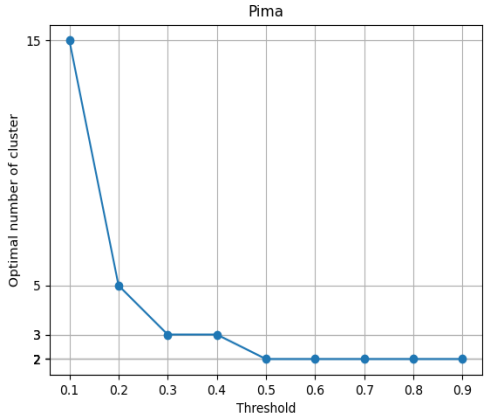


(a) Iris

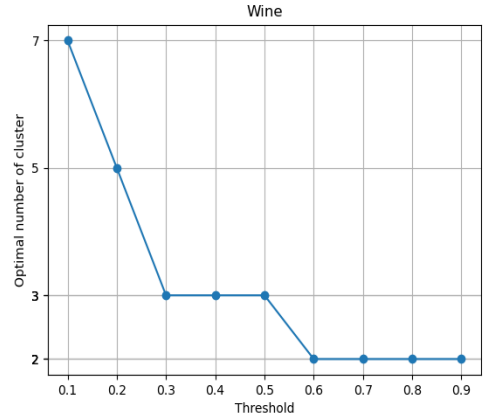


(b) Parkinsons

Figure 4.2: Sensitivity Analysis for threshold of Benchmark datasets (a) Iris (b) Parkinsons

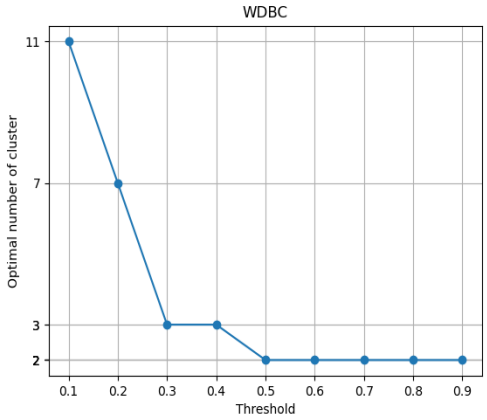


(a) PIDD

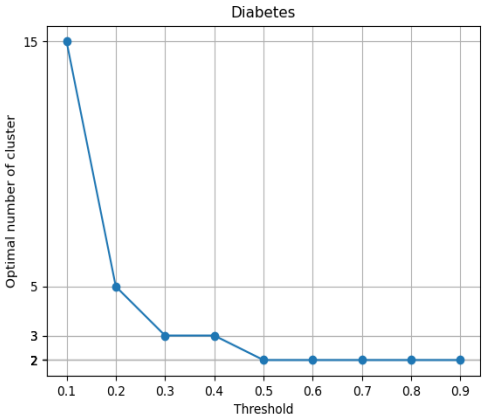


(b) Wine

Figure 4.3: Sensitivity Analysis for threshold of Benchmark datasets (a) PIDD (b) Wine

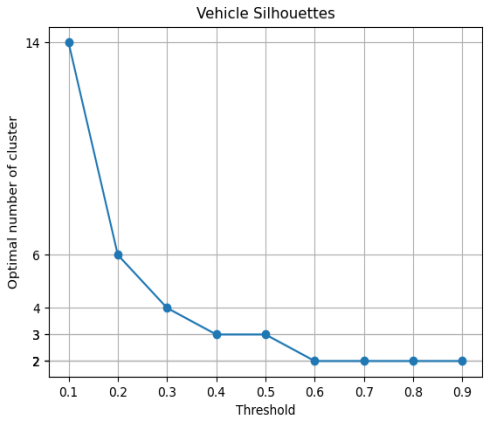


(a) WDBC

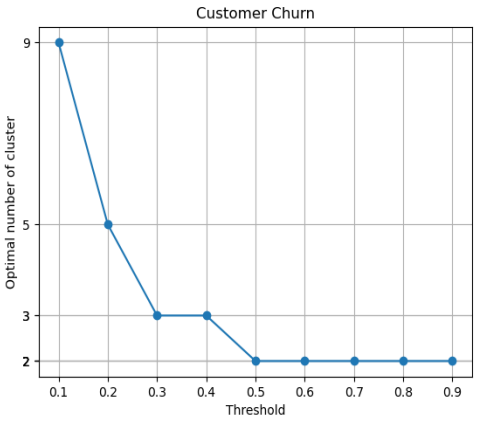


(b) Diabetes

Figure 4.4: Sensitivity Analysis for threshold of Benchmark datasets (a) WDBC (b) Diabetes



(a) Vehicle Silhouettes



(b) Customer Churn

Figure 4.5: Sensitivity Analysis for threshold of Benchmark datasets (a) Vehicle Silhouettes (b) Customer Churn

4.6 Summary

This chapter proposed an algorithm to find the optimal number of clusters called the Max of Min Algorithm, which uses a distance-based, linear-time approach. The bottleneck in the Max of Min algorithm is that there is a serial computation for the

distance between each data point and all cluster centres, which can be done in parallel. So, this part is parallelised, and a scalable version of Max of Min is introduced, called the S-MaxMin algorithm. This algorithm is made scalable using Apache Spark Big Data framework to reduce the time it takes for a large dataset to find the optimal number of clusters.

The S-MaxMin algorithm was applied to various benchmark datasets to validate its effectiveness. Our results correctly identified the number of clusters compared to the ground truth. Additionally, unlabelled real-life plant genome SNP datasets were used to determine the optimal number of clusters, achieving high Silhouette scores when conducting experimental analysis using clustering algorithms, further validating our method. Moreover, to identify the threshold value (hyperparameter), sensitivity analysis was conducted on different threshold values (ranging $[0 - 1]$) on different benchmark datasets and found an optimal value of 0.5.

Chapter 5

Scalable Integrated Framework for Genome Assembly and Special Trait Identification for Real-Life Crop data

The methods developed in Chapter 3 and Chapter 4 are designed for SNP sequences to extract features based on the complex network and identify the optimal number of clusters using the S-MaxMin algorithm, respectively. The plant breeders (Agri-Scientist) face the challenge of identifying crops similar to unknown crops as no end-to-end solution is developed, and each process has to be done with separate tools, making it difficult for plant breeders to set up the environment and then analyse. Thus, in this chapter, an integrated framework is developed that first preprocesses the raw sequences and then identifies crops similar to the unknown plant species.

The flow of this chapter is as follows: firstly, we discuss the conversion of the short reads (raw data) into SNP sequences in Section 5.1, followed by the architecture for scalable integrated framework solution, which will be discussed in Section 5.2. Finally, the experimental results will be discussed for the developed architecture on real-life plant genome sequence JS-335 (collected from ICAR - Indian Institute of Soybean Research Indore in raw short reads form) in Section 5.3.

5.1 Data collection and Preprocessing to extract SNP sequences

Plant genome data are generated by the sequencing machine. The sequencing machine produces raw short reads (refer to Appendix A.1), as illustrated in Figure 5.1. To extract the genome sequences such as DNA/RNA, Protein or SNP sequence from the raw short reads, it is initially necessary to perform genome assembly, which involves arranging the short reads in the correct order. Upon completion of the genome assembly, the resultant output is referred to as the Whole Genome Sequence (WGS). From this sequence, various encoded sequences, including those for DNA/RNA, Protein or SNP sequence, can be derived. In this chapter, the focus is specifically on the SNP sequence. Composed of four distinct nucleotides—adenine (A), thymine (T), guanine (G), and cytosine (C)—these sequences facilitate the categorization of individuals into disease risk categories in plants and enable the more reliable prediction of treatment outcomes [1], [2].

The extraction of the SNP sequences from the raw short reads is shown in Fig. 5.2. Initially, the quality of the short reads is assessed using the FastQC software package (discussed in detail in Appendix A.2). Corrective measures such as adaptor or base trimming are implemented if the quality assessment for short reads indicates poor quality. Subsequently, the short reads are assembled utilizing the BWA-mem (Burrows-Wheeler Alignment Maximal Exact Matching) algorithm (discussed in detail in Appendix A.4), which facilitates the alignment of short reads relative to the higher coverage reference genome sequence. This reference genome assembly process yields a Sequence Alignment Map (SAM) file (discussed the attributes of SAM file in Appendix A.5). Given the substantial size of the SAM file, which complicates further processing, it is converted into a Binary Alignment Map (BAM) file, which is a lossless compression of the SAM file, thereby enabling more efficient handling. Following this, the variant calling function extracts SNPs from the assembled sequence or BAM file. The newly extracted SNP dataset is merged with the pre-existing SNP dataset.

A new Indian variety real-life plant genome sequence is collected from ICAR-Indian

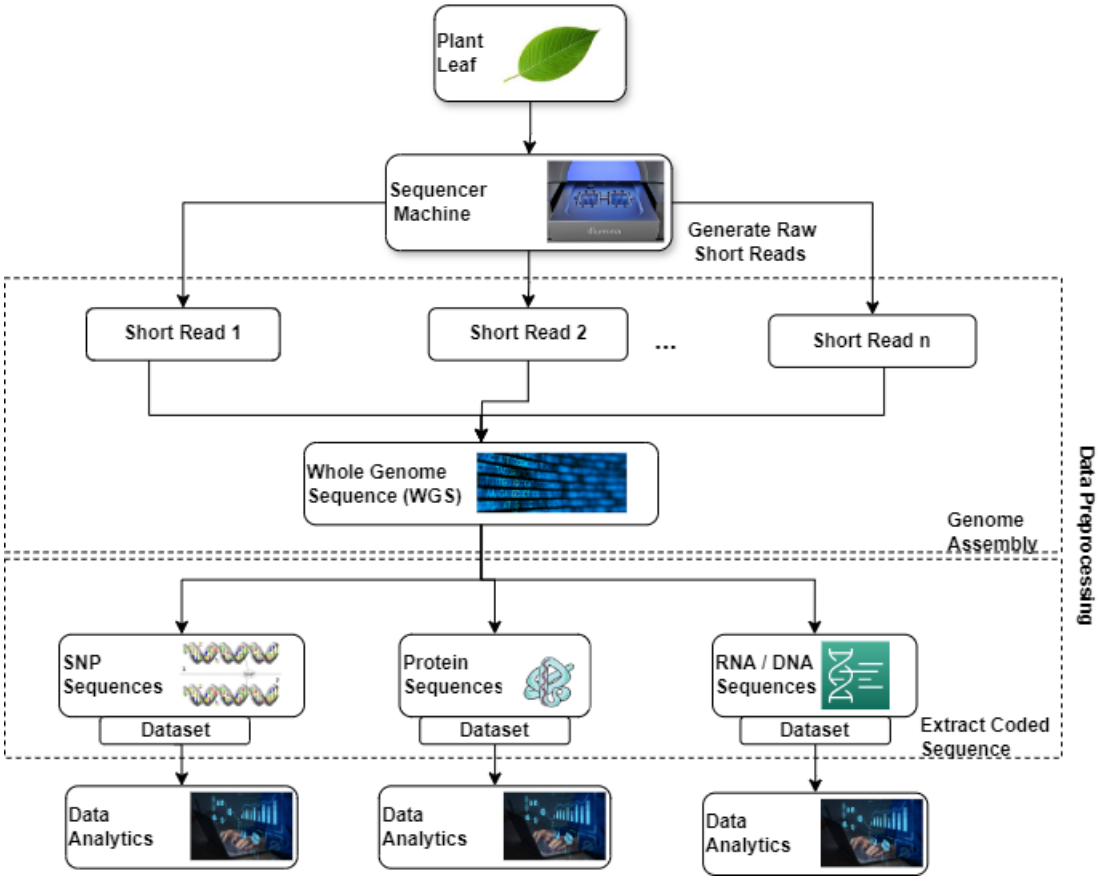


Figure 5.1: Overview

Institute of Soybean Research, Indore, called a JS-335 variant. The genome of JS-335 is assembled using the genome assembly algorithm with the help of the Reference genome William82 Assembly 4 Genomic sequence, achieving 100x coverage as shown in Fig. 5.2. Subsequently, SNP sequences are extracted from the JS-335 variant using a variant calling function (refer Appendix A.6) and merged with the open source SNP50K Wm82 a2 dataset, forming a new dataset designated as the Modified Wm82 a2 SNP50K dataset with JS-335. This dataset is approximately 1.7GB, comprising 48,632 SNP sequences with 13,903 nucleotides per sequence.

When integrating the data preprocessing and data analytics components (including feature extraction and determining the optimal number of clusters), a new scalable integrated framework is developed. This framework will be discussed in detail in the following section.

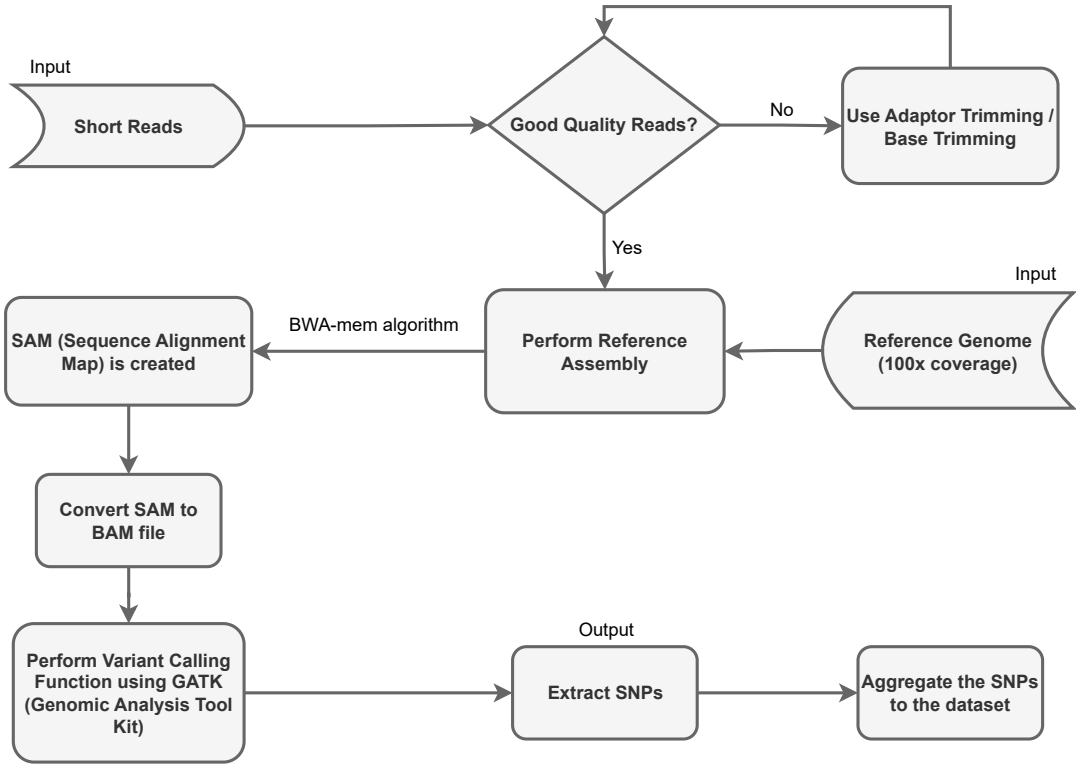


Figure 5.2: Preprocessing (Raw Short Reads → SNPs)

5.2 Architecture of Scalable Integrated Framework

The scalable integrated framework is shown in Fig. 5.3. As discussed in Section 5.1, the raw short-read data is first preprocessed, and the plant genome SNP sequence is extracted. The preprocessing includes the assembly of short reads with the help of the reference genome using the BWA-mem algorithm to obtain WGS, followed by a variant-calling operation on the WGS to extract the SNP sequence.

This new SNP sequence of unknown species of soybean crops is merged with the existing open-source SNP50K Wm82 a2 dataset for soybean crops, which can help identify which existing soybean crops are similar. SNP50K Wm82 a2 dataset contains around 50,000 SNP sequences, which are carefully selected from an initial set of more than 200,000 SNP sequences such that the dataset achieves an even distribution across the genome. The dataset has been used extensively for genotyping the USDA Soybean Germplasm Collection and has facilitated numerous genomic studies and breeding

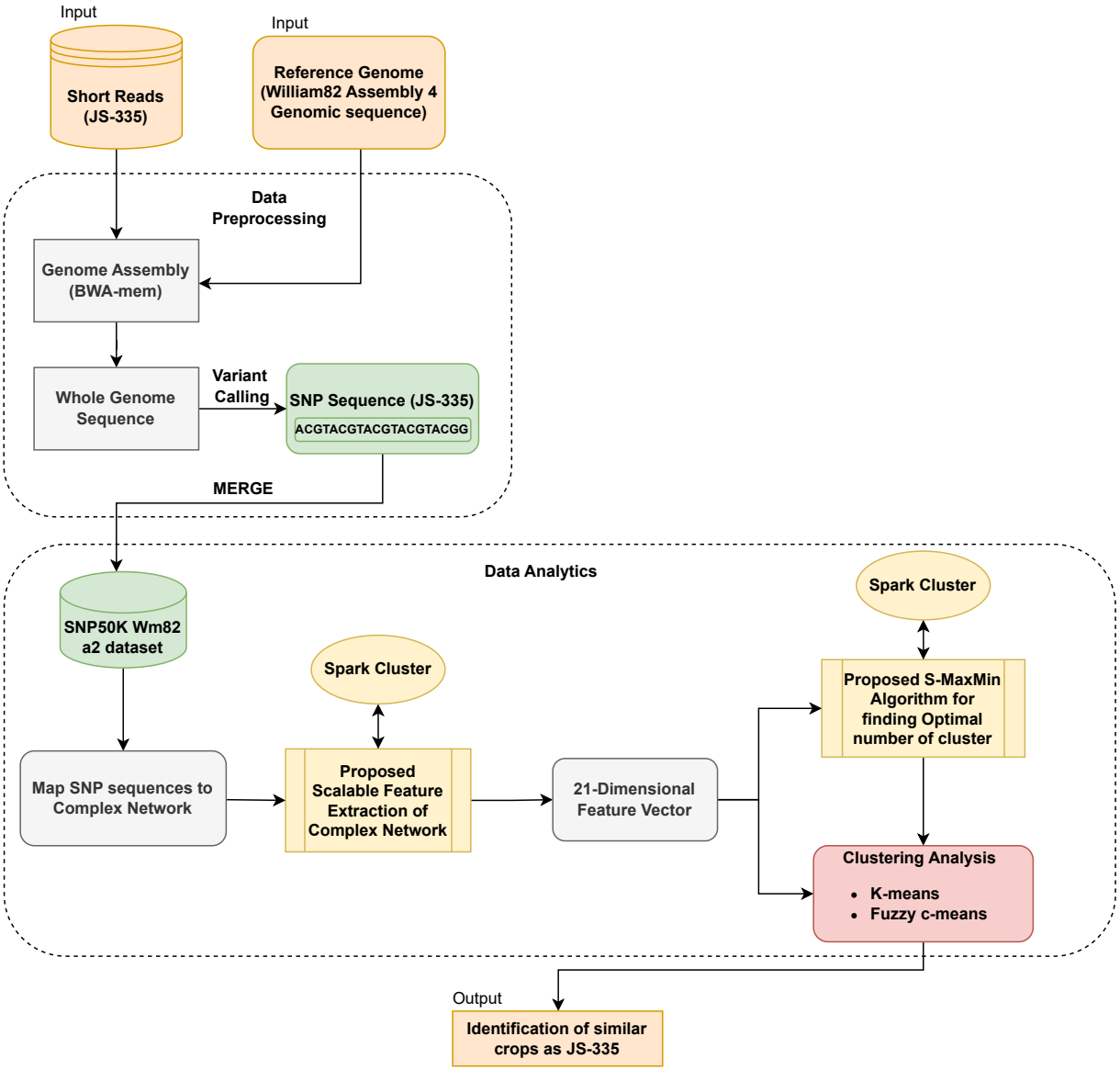


Figure 5.3: Scalable Integrated Framework for Breeders for an Indian variety soybean genome SNP sequence: JS-335

efforts in soybeans [1].

After merging this new sequence, the proposed scalable approach based on the complex network is applied to find the features of all the SNP sequences present in the merged SNP50K Wm82 a2 dataset. A 21-dimensional feature vector of all the sequences is obtained. The advantage of using our proposed approach is that it is

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

scalable, takes much less time to extract the features, and captures context-based information.

The other input that the clustering algorithm requires is the input of the number of clusters. The proposed scalable approach, S-MaxMin, is used to find the optimal number of clusters because the underlying algorithm used is the clustering algorithm to find the optimal number of clusters. Hence, one must initially select a predetermined number of clusters to apply clustering effectively. Subsequently, this initial choice should be evaluated for every cluster number to determine the optimal number of clusters using appropriate methods. Therefore, calculating the optimal number of clusters from other methods takes enormous time.

These two inputs can be used to perform clustering analysis. After performing clustering analysis, firstly, the cluster in which the unknown sequence's datapoint is present is identified. Then, the other data points in the respective cluster are identified, and the mapped sequence name is found. The sequence name has already defined characteristics, which can help find unknown sequences' properties.

Further, in the subsequent section, we will discuss the results of the scalable integrated framework on the real-life soybean crop JS-335.

5.3 Experiments and Results

In this section, the results for the real-life soybean crop JS-335, which is in its raw form, are presented in terms of other crops similar to JS-335. We have developed an integrated framework called Agri Genomics Data Management Platform that allows users to directly input short reads from an unknown sequence (in our case, JS-335) and a reference genome (in our case, the William82 Assembly 4 Genomic sequence). The attributes of inputs are illustrated in Fig. 5.4.

Firstly, this integrated framework assembles the short reads using the reference genome. Following assembly, SNP sequences are extracted. These sequences are then integrated with the open-source SNP50K Wm82 a2 dataset. Subsequently, the proposed feature extraction based on the complex network is performed for each sequence.

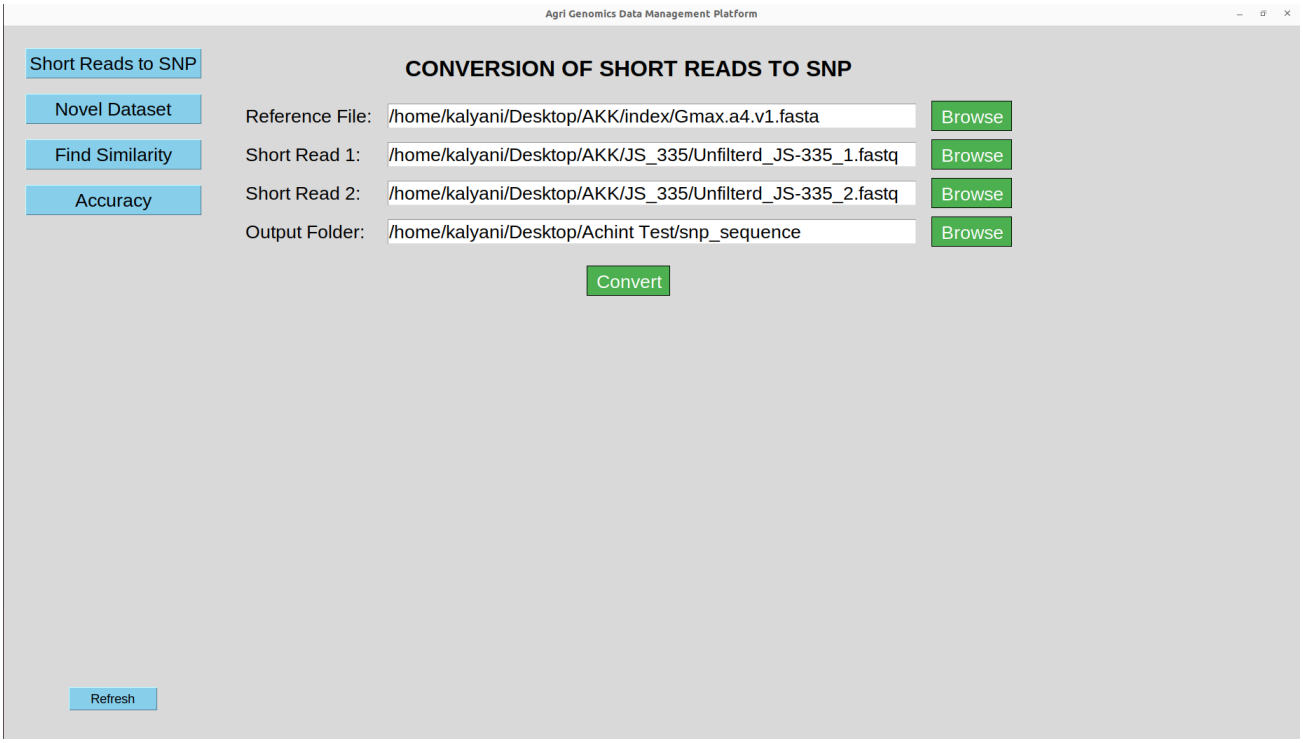


Figure 5.4: Input attributes in Agri Genomics Data Management Platform

The proposed S-MaxMin algorithm determines the cluster number, which is 2 in this case, and the FCM algorithm is used to cluster these sequences.

The output is a list of crops similar to those with the unknown sequence. As shown in Fig. 5.5, sequences similar to JS-335 are identified. After executing the proposed integrated framework, we got the results of the crop varieties from the SNP50K Wm82 a2 dataset similar to JS-335. Some of which are: 'OHS305', 'PI208439', 'PI339867', 'PI340052', 'PI360834', 'PI374160', 'PI374178', 'PI407287', 'PI407998B', 'PI407999-1', 'PI416792', 'PI423843', 'PI424140', 'PI424352', 'PI424377', 'PI437902C', 'PI438257B', 'PI503333', 'PI506677', 'PI506837', 'PI506903', 'PI507069', 'PI509091B', 'PI533656', 'PI546051', 'PI548207', 'PI548287', 'PI548972', 'PI567447C', 'PI567615', 'PI567698B', 'PI572240', 'PI587979B', 'PI594433B', 'PI594465', 'PI594552A', 'PI594583', 'PI603741B', 'PI603778', 'PI612610', 'PI615446', 'PI628962', 'PI632905', 'PI653916B', 'PI654042'.

This framework assists plant breeders in identifying soybean crops that are genetically similar to an unknown plant sequence, thereby helping in the identification of

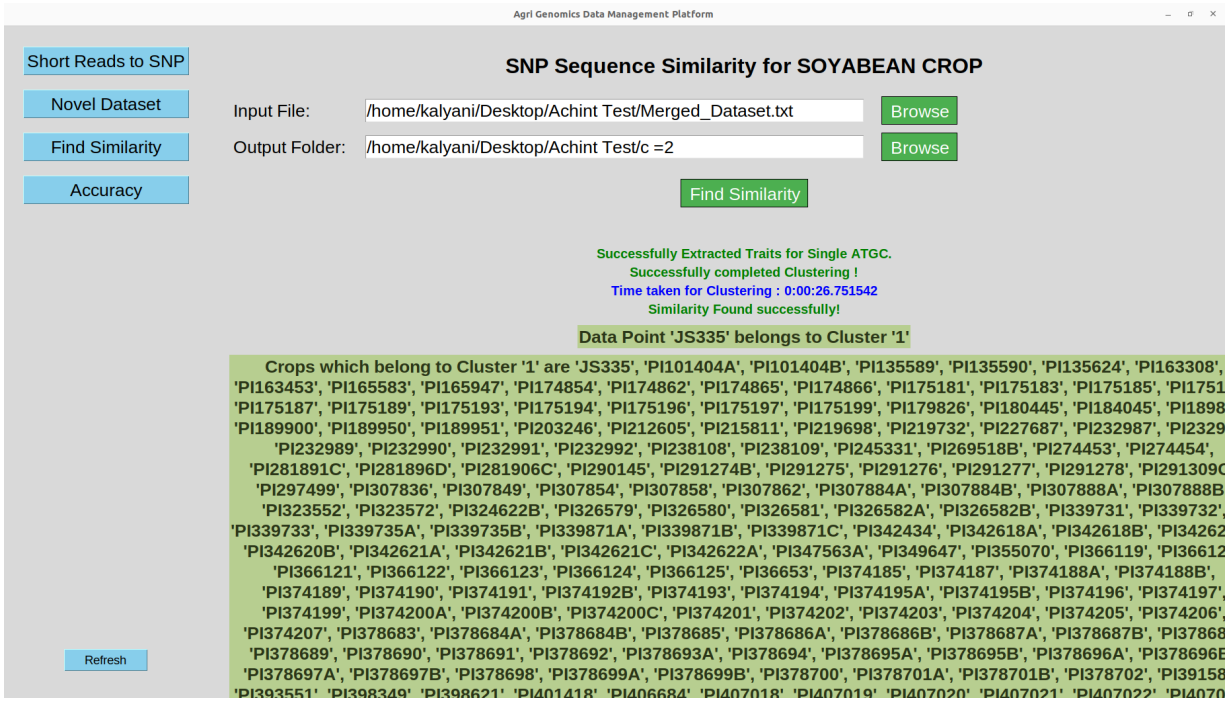


Figure 5.5: Results of crops for Identification of similar crops as JS-335

crops similar to those from the unknown sequence. This integrated framework can serve as a general-purpose tool for plant breeders working with various crop varieties, including rice, wheat, etc. The only requirement is to replace the open-source dataset (in this case, SNP50K Wm82 a2 dataset) in the framework with the corresponding dataset for the specific crop, along with the respective input for that crop.

5.4 Summary

This chapter discusses the preprocessing of the short reads of JS-335 (a local Indian soybean variety) and the architecture for an integrated framework, an end-to-end solution for plant breeders. The preprocessing includes filtering the raw short reads and performing genome assembly to make a WGS. From the WGS, the SNP sequences were extracted using the variant calling function. After preprocessing, the new sequence was merged into the existing dataset. The clustering analysis was performed using the proposed scalable feature extraction technique based on the complex network and the proposed S-MaxMin algorithm. After the clustering analysis, different crops similar

to the JS-335 variant were identified, which will help plant breeders identify crops similar to the unknown sequence. The same is shown with the developed integrated framework for JS-335 short read.

Chapter 6

Conclusion and Future Work

This thesis primarily investigates the feature extraction approaches for SNP genome sequences and methods to find the optimal number of clusters. In particular, we have proposed a scalable alignment-free feature extraction technique using complex network theory, integrating the Apache Spark Big Data framework to handle vast amounts of genomic sequences. Additionally, the S-MaxMin algorithm was proposed to determine the optimal number of clusters.

The plant breeders face the challenge of identifying crops similar to unknown crops as no end-to-end solution has been developed, and each process has to be done with separate tools. Therefore, using these two proposed approaches, we have developed an integrated framework to help plant breeders identify similar crops for unknown sequences.

6.1 Summary of Contributions

The objectives specified in Section 1.3 have been successfully fulfilled by the following main contributions:

1. **A Scalable method for extracting features using a complex network from SNP sequences:** We have proposed a scalable algorithm for feature extraction to handle large SNP sequences (Big Data) based on the complex network by distributing the sequences on various cores using the Apache Spark Big Data

processing framework. Our research utilized several real-life plant genome SNP datasets, including Modified Williams82 a2, Williams82 a1, MAGIC-Rice, and the 248Entries Rice dataset. These datasets were used to compare our feature extraction technique against other state-of-the-art alignment-free methods. Our experimental analysis demonstrates that our approach surpasses these existing techniques. To statistically verify that our proposed approach for feature extraction is better than the other approaches, the Friedman test was used, followed by the Nemenyi post-test. It was found that the majority of the algorithm's rank differences from the average rank of our proposed approach are greater than the critical difference. Hence, our proposed approach for feature extraction performs better statistically.

2. **A Scalable Algorithm for finding the optimal number of clusters:** We have proposed a scalable algorithm to find the optimal number of clusters, called the S-MaxMin algorithm, for the partitioned dataset using a distance-based, linear-time approach. This algorithm was applied to eight benchmark datasets: Iris, Parkinsons, PIDD, Wine, WDBC, Diabetes, Vehicle Silhouettes and Customer Churn to validate its effectiveness. Our results correctly identified the number of clusters compared to the ground truth. Additionally, unlabeled SNP datasets were used to determine the optimal number of clusters, achieving high Silhouette scores when experimental analysis was conducted using clustering algorithms, further validating our method.

3. **Scalable Integrated Framework for Genome Assembly and Special Trait Identification for Real-Life Crop data:** We have developed an integrated framework for plant breeders to identify crops similar to unknown plant sequences using the proposed scalable approach for feature extraction based on the complex network (Contribution 1) and the proposed S-MaxMin algorithm (Contribution 2). This integrated framework is named Agri Genomics Data Management Platform, and it assists plant breeders in identifying soybean crops that are genetically similar to those of an unknown plant sequence, which helps

in identifying unique special traits in unknown plant species such as rust resistance, drought resistance, etc., which can then be used to perform genetic engineering to improve those traits.

6.2 Future Research Directions

This work suggests some interesting directions for future research. This work presents a scalable feature extraction approach wherein features are extracted, followed by applying the clustering algorithms to identify special traits of unknown species. However, deep learning methods can effectively capture features from complex genome sequences, but several challenges need to be addressed. Firstly, the variable and extensive length of sequences may lead to the loss of contextual information. Secondly, there are a limited number of samples, which may overfit the model. Additionally, the interpretation of results, i.e. features derived from autoencoder architectures or RNN-based models, remains problematic as derived features are crucial for validating the biological relevance of the clusters. Another research direction is to determine the optimal number of clusters. In the S-MaxMin algorithm, only partitioned datasets are considered. It has not yet been explored for other datasets to predict the optimal number of clusters.

Appendix

Appendix A

Biotech tools

A.1 Short Reads

Short reads involve small overlapping fragments of the Whole Genome Sequence of the current species, typically ranging from 50 to 300 nucleotides. A typical short read is shown in Figure A.1. In this, the first line represents the label for short reads,

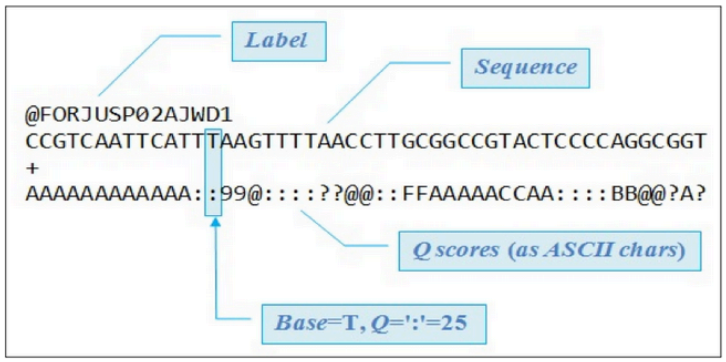


Figure A.1: Short Reads

the second line is the sequence of short reads, and the third line corresponds to the quality of each associated nucleotide. In Figure A.1 'T' is associated with ':' have a quality score $Q = 25$. Generally, the quality score >= 25 is considered a good quality score.

A.2 FASTQC: To check the quality of reads

We used the FastQC package to visualise the short reads. It is a program designed to spot potential problems in high throughput sequencing datasets such as short reads. It runs a set of analyses on one or more raw sequence files in fastq format and produces a report summarising the results. Fig. A.2 mentions one of the short reads reports. The short reads are of soybean species JS-335 collected from ICAR-Indian Institute of Soybean Research, Indore.

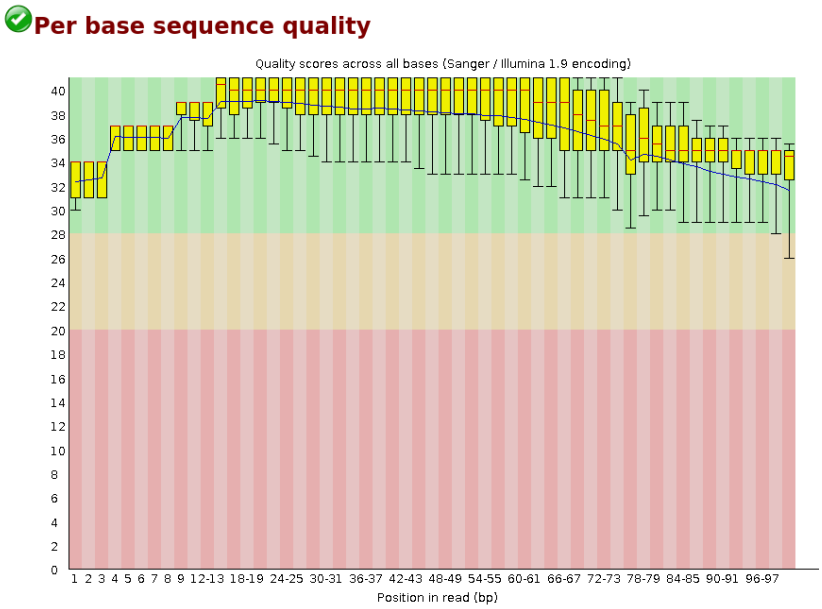


Figure A.2: Quality Check of Short Reads

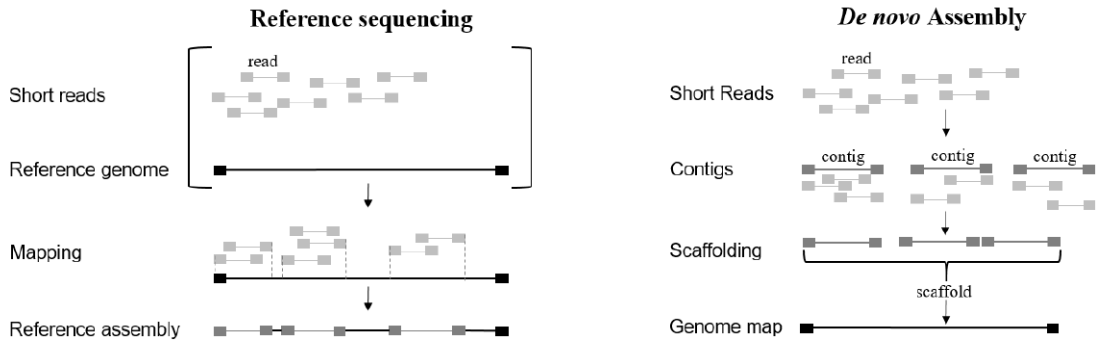
A.3 Genome Assembly

Genome assembly refers to putting new short reads into the correct order. Assembly is required because sequence read lengths are much shorter than most genomes. There are two types of Genome Assembly:

- **Reference Assembly:** Reference assembly [42] as shown in Figure A.3a, the process involves aligning the new short reads to the pre-existing reference genome

and filling in gaps or resolving discrepancies between the reference genome and the new short reads. This approach can lead to higher accuracy and continuity of the final assembly because the pre-existing genome provides a structure for the new short reads to align to.

- **De-novo Assembly:** De novo assembly [42] as shown in Figure A.3b is used when there is no available reference genome or when the genome is highly divergent from known reference genomes. In de novo assembly, the reads are assembled into contigs or longer contiguous sequences without any prior knowledge of the genome. The resulting assembly is often fragmented and may contain errors due to repeats, sequencing errors, or other factors. Additional methods, such as scaffolding or gap-filling, may improve the assembly quality.



(a) Reference Assembly (b) De Novo Assembly

Figure A.3: Genome Assembly

A.4 Reference Assembly: BWA-MEM algorithm

BWA-MEM is a widely used alignment algorithm that stands for Burrows-Wheeler Aligner - Maximal Exact Match. It is designed to align high-throughput sequencing reads to a reference genome. The algorithm is divided into three main steps: seeding, chaining, and extension.

- **Indexing:** BWA first constructs an index of the reference genome. This is done using the Burrows-Wheeler transform (BWT), which compresses the genome into a reversible, suffix-sorted permutation of its characters. BWA then constructs a suffix array from the BWT, which is used to quickly find exact matches between the reads and the reference genome.

- **Seed generation:** BWA divides each read into short substrings called seeds, typically 19-20 nucleotides long. It then uses the suffix array to search the reference genome for exact matches to these seeds. These matches are called seed hits.

- **Seed extension:** For each seed hit, BWA extends the alignment in both directions, using a dynamic programming algorithm to find the optimal alignment between the read and the reference genome. The extension process considers mismatches, insertions, and deletions and is guided by a quality score that reflects the confidence in the base calls in the read.

- **Scoring:** BWA scores the alignments based on the number of matching bases, mismatches, and gaps (insertions or deletions) in the alignment. It also considers the quality scores of the base calls in the read.

- **Output:** BWA reports the best alignment for each read, its score and any other relevant information, such as the location of mismatches or gaps in the alignment.

A.5 SAM (Sequence Alignment Map) File Format

In the SAM format, each alignment line usually signifies the linear alignment of a segment. The line contains 11 or more fields, separated by TAB characters.

The mandatory fields in the SAM file are shown in Table A.1.

Table A.1: Attributes of SAM File

QNAME	FLAG	RNAME	POS	MAPQ	CIGAR	RNEXT	PNEXT	TLEN	SEQ	QUAL	OPTIONAL

Table A.2: Overview of Fields

SNo.	Field	Datatype	Brief Description
1	**QNAME**	STRING	Unique NAME of Short Read
2	**FLAG**	INT	BitWise Flag (65536 Different Values)
3	**RNAME**	STRING	Reference Sequence Name
4	**POS**	INT	MAPPING POSITION
5	**MAPQ**	INT	MAPPING QUALITY
6	**CIGAR**	STRING	CIGAR String
7	**RNEXT**	STRING	Reference Name of the Next Read
8	**PNEXT**	INT	Position of the Next Read
9	**TLEN**	INT	Observed Template Length
10	**SEQ**	STRING	Segment Sequence
11	**QUAL**	STRING	ASCII of Phred Score

Table A.2 gives an overview of the fields in the SAM file format. In this, the most important attributes are POS and $CIGAR$ string. $CIGAR$, which stands for Concise Idiosyncratic Gapped Alignment Report, is a compact representation of an alignment used in the SAM file format.

A.6 Variant Calling: Extraction of SNP sequence

The BWA-MEM algorithm plays a crucial role in aligning short reads to the reference genome, thereby performing reference assembly. The output of the BWA-mem algorithm is the aligned short reads stored in the SAM (Sequence Alignment Map) file. The SAM file is a tab-delimited text file that contains information for each individual to read and its alignment with the genome. The compressed binary version of SAM is called a BAM file. We use this version to reduce size and allow for indexing, which enables efficient random access to the data contained within the file. Then, we will perform a variant calling function to extract SNP sequences using the GATK (Genome Analysis Toolkit) package.

Bibliography

[1] Annarita D'Addabbo, Orazio Palmieri, Anna Latiano, Vito Annese, Sayan Mukherjee, and Nicola Ancona. Rs-snp: a random-set method for genome-wide association studies. *BMC genomics*, 12:1–7, 2011.

[2] Alexander R. Pico, Ivan V. Smirnov, Jeffrey S. Chang, Ru-Fang Yeh, Joseph L. Wiemels, John K. Wiencke, Tarik Tihan, Bruce R. Conklin, and Margaret R. Wrensch. Snplogic: an interactive single nucleotide polymorphism selection, annotation, and prioritization system. *Nucleic Acids Research*, 37:D803 – D809, 2008.

[3] Libin Liu, Yee-kin Ho, and Stephen Yau. Clustering dna sequences by feature vectors. *Molecular phylogenetics and evolution*, 41(1):64–69, 2006.

[4] Wolfgang Kaisers, Holger Schwender, and Heiner Schaal. Hierarchical clustering of dna k-mer counts in rnaseq fastq files identifies sample heterogeneities. *International Journal of Molecular Sciences*, 19(11):3687, 2018.

[5] Anil K Jian. Data clustering: 50 years beyond k-means, pattern recognition letters. *Corrected Proof*, 2009.

[6] Hae-Sang Park and Chi-Hyuck Jun. A simple and fast algorithm for k-medoids clustering. *Expert systems with applications*, 36(2):3336–3341, 2009.

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

[8] Robert L Thorndike. Who belongs in the family? *Psychometrika*, 18(4):267–276, 1953.

[9] MA Syakur, B Khusnul Khotimah, EMS Rochman, and Budi Dwi Satoto. Integration k-means clustering method and elbow method for identification of the best customer profile cluster. In *IOP conference series: materials science and engineering*, volume 336, page 012017. IOP Publishing, 2018.

[10] Mohamed Cassim Alibuhtto and Nor Idayu Mahat. Distance based k-means clustering algorithm for determining number of clusters for high dimensional data. *Decision Science Letters*, page 51–58, 2020.

[11] Congming Shi, Bingtao Wei, Shoulin Wei, Wen Wang, Hai Liu, and Jialei Liu. A quantitative discriminant method of elbow point for the optimal number of clusters in clustering algorithm. *Eurasip Journal on Wireless Communications and Networking*, 2021:1–16, 2021.

[12] Robson P Bonidia, Lucas DH Sampaio, Douglas S Domingues, Alexandre R Paschoal, Fabrício M Lopes, André CPLF de Carvalho, and Danilo S Sanches. Feature extraction approaches for biological sequences: a comparative study of mathematical features. *Briefings in Bioinformatics*, 22(5):bbab011, 2021.

[13] Jorge Veiga, Roberto R Expósito, Xoán C Pardo, Guillermo L Taboada, and Juan Tourifio. Performance evaluation of big data frameworks for large-scale data analytics. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 424–431. IEEE, 2016.

[14] 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.

[15] 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*, pages 1–18, 2023.

[16] 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*, pages 1–35, 2023.

[17] Andrzej Zielezinski, Susana Vinga, Jonas Almeida, and Wojciech M Karlowski. Alignment-free sequence comparison: benefits, applications, and tools. *Genome biology*, 18:1–17, 2017.

[18] Christopher M Biship. *Pattern recognition and machine learning (information science and statistics)*. 2007.

[19] Sergios Theodoridis and Konstantinos Koutroumbas. *Pattern recognition*. 2006.

[20] L da F Costa, Francisco A Rodrigues, Gonzalo Travieso, and Paulino Ribeiro Villas Boas. Characterization of complex networks: A survey of measurements. *Advances in physics*, 56(1):167–242, 2007.

[21] Ye Zhu, Jun Gong, Simeng Wu, et al. An efficient community detection method based on path proximity and local edge betweenness centrality. *Academic Journal of Computing & Information Science*, 4(1):1–6, 2021.

[22] Sergei N Dorogovtsev and José FF Mendes. The shortest path to complex networks. *arXiv preprint cond-mat/0404593*, 2004.

[23] Thomas F Coleman and Jorge J Moré. Estimation of sparse jacobian matrices and graph coloring blems. *SIAM journal on Numerical Analysis*, 20(1):187–209, 1983.

[24] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

[25] M A Syakur, B K Khotimah, E M S Rochman, and B D Satoto. Integration k-means clustering method and elbow method for identification of the best customer profile cluster. *IOP Conference Series: Materials Science and Engineering*, 336:012017, April 2018.

[26] Francis S Collins, Ari Patrinos, Elke Jordan, Aravinda Chakravarti, Raymond Gesteland, LeRoy Walters, members of the DOE, and NIH planning groups. New goals for the us human genome project: 1998-2003. *science*, 282(5389):682–689, 1998.

[27] 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.

[28] Michael A Quail, Iwanka Kozarewa, Frances Smith, Aylwyn Scally, Philip J Stephens, Richard Durbin, Harold Swerdlow, and Daniel J Turner. A large genome center's improvements to the illumina sequencing system. *Nature methods*, 5(12):1005–1010, 2008.

[29] C. Greene, J. Tan, M. Ung, J. H. Moore, and Chao Cheng. Big data bioinformatics. *Journal of Cellular Physiology*, 229, 2014.

[30] T. Head-Gordon and J. Wooley. Computational challenges in structural and functional genomics. *IBM Syst. J.*, 40:265–296, 2001.

[31] Jyoti Nandimath, Ekata Banerjee, Ankur Patil, Pratima Kakade, Saumitra Vaidya, and Divyansh Chaturvedi. Big data analysis using apache hadoop. In *2013 IEEE 14th International Conference on Information Reuse & Integration (IRI)*, pages 700–703. IEEE, 2013.

[32] 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*, 1:145–164, 2016.

[33] Yin Huai, Ashutosh Chauhan, Alan Gates, Gunther Hagleitner, Eric N Hanson, Owen O'Malley, Jitendra Pandey, Yuan Yuan, Rubao Lee, and Xiaodong Zhang. Major technical advancements in apache hive. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 1235–1246, 2014.

[34] Shanjiang Tang, Bingsheng He, Ce Yu, Yusen Li, and Kun Li. A survey on spark ecosystem: Big data processing infrastructure, machine learning, and applications. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pages 3779–3780. IEEE, 2023.

[35] 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.

[36] Ujjwal Maulik and Sanghamitra Bandyopadhyay. Performance evaluation of some clustering algorithms and validity indices. *IEEE Transactions on pattern analysis and machine intelligence*, 24(12):1650–1654, 2002.

[37] Qijian Song, Jerry Jenkins, Gaofeng Jia, David L Hyten, Vince Pantalone, Scott A Jackson, Jeremy Schmutz, and Perry B Cregan. Construction of high resolution genetic linkage maps to improve the soybean genome sequence assembly glyma1. 01. *BMC genomics*, 17:1–11, 2016.

[38] Qijian Song, David L Hyten, Gaofeng Jia, Charles V Quigley, Edward W Fickus, Randall L Nelson, and Perry B Cregan. Development and evaluation of soysnp50k, a high-density genotyping array for soybean. *PloS one*, 8(1):e54985, 2013.

[39] 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.

[40] 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*, 10:1–17, 2017.

[41] Catherine L Blake. Uci repository of machine learning databases. *http://www. ics. uci. edu/˜ mlearn/MLRepository. html*, 1998.

[42] Jeongkyu Kim, Mingeun Ji, and Gangman Yi. A review on sequence alignment algorithms for short reads based on next-generation sequencing. *IEEE Access*, 8:189811–189822, 2020.