Chapter 2

2 Gene Expression Data Analysis

2.1 Background of Molecular Biology

In order to understand the volume and nature of the data pertaining to gene expressions, the biology of cells and their mechanism to replicate and code information should be understood. A cell can be divided into two classes called prokaryotic and eukaryotic, with the latter containing a “true” nucleus i.e. it has a nuclear membrane. The cell is enclosed and protected by a phospholipid bilayer with the nucleus embedded in the cell’s cytoplasm. The nucleus has its own nuclear envelope with nuclear pores located around it to allow the DNA (deoxyribonucleic acid) to interact with the rest of the machinery in the cytoplasm.

Molecular Biology is the study of all the molecules in living things. These molecules include large molecules such as DNA or proteins, the smaller molecules like nucleotides and amino acids that are the building blocks for DNA and protein, and other small molecules such as vitamins, glucose and fats. All living organisms store information that is necessary for growth, reproduction and evolution in genes, which is a region of the DNA. The DNA is the major carrier of genetic material in living organisms; i.e. it is responsible for inheritance. In 1953 James Watson and Francis Crick deduced the three-dimensional structure of DNA.

The DNA is a double helix of complementary strands composed of four basic molecules called nucleotides, which are identical, except that each contains a different nitrogen base. Each nucleotide contains phosphate, sugar (of the deoxy–ribose type) and one of the four bases: Adenine, Guanine, Cytosine and Thymine (usually denoted as A, G, C, and T respectively). Each nucleotide molecule from one chain always bonds with a complementary nucleotide molecule from the other chain and form a nucleotide pair called *base–pair*. A base-pair is simply an interaction between the bases standing opposite of each other as shown in *Figure* 2.1. The complementary bases adenine and thymine are joined by hydrogen bonds, and so are the complementary bases cytosine and guanine.

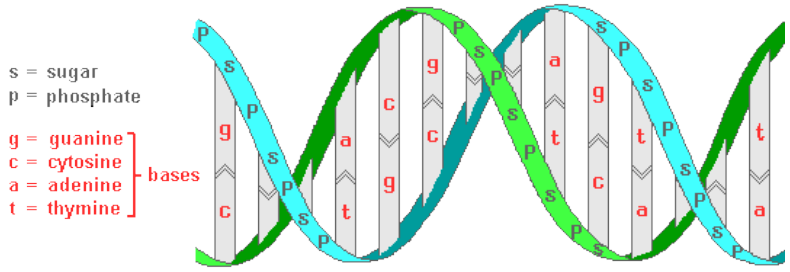


Figure 2.1: Double helix structure of the DNA

The cell uses DNA to transmit its hereditary information to the next generation via segments of the DNA called a gene [6]. The information transmitted by the DNA pertains to the construction of proteins, which are the functional units of life. The DNA molecule is directional, due to the asymmetrical structure of the sugars, which constitute the skeleton of the molecule. The directions of the two complementary DNA strands are reversed to one another.

The genome is all the genetic material and collection of genes that is required by an organism to produce its proteins [6]. A gene is a region of DNA that controls a discrete hereditary characteristic, usually corresponding to a single mRNA (messenger ribonucleic acid) carrying the information for constructing a protein. The human genome has about 30,000 to 40,000 genes whereas a simple yeast cell has about 6,000 genes [6]. The remarkable fact of life is that every multicellular organism has its entire genome contained in every cell [7]. The cells of different tissues however can differ in terms of the amount and type of proteins produced in the cells.

A gene is said to be expressed if the protein which it codes for is produced or synthesized [7]. In an average human there are expression levels for about 10,000 different genes, which are collectively referred to as the expression profile of the cell [7]. A large number of genes located in all the cells of an organism share common functions, metabolism being such an example. The various internal and external factors however can adjust the amount of some gene expressions in different cells and even in the same cell. The ribosomes are the protein synthesizing factories for the cell and situated outside the nucleus in the cytoplasm whereas the DNA is protected inside the nuclear envelope. The direct interaction is therefore broken between the ribosomes and genes. The communication occurs via a linear molecule called messenger ribonucleic acid (mRNA), which is an exact copy of the gene that is being expressed. The mRNA is transcribed inside the nucleus and transported out to the ribosomes where it is translated into amino acids and subsequently into protein. A single gene is able to produce numerous identical protein molecules by manufacturing multiple copies of the corresponding mRNA molecule, as illustrated in *Figure* 2.2.

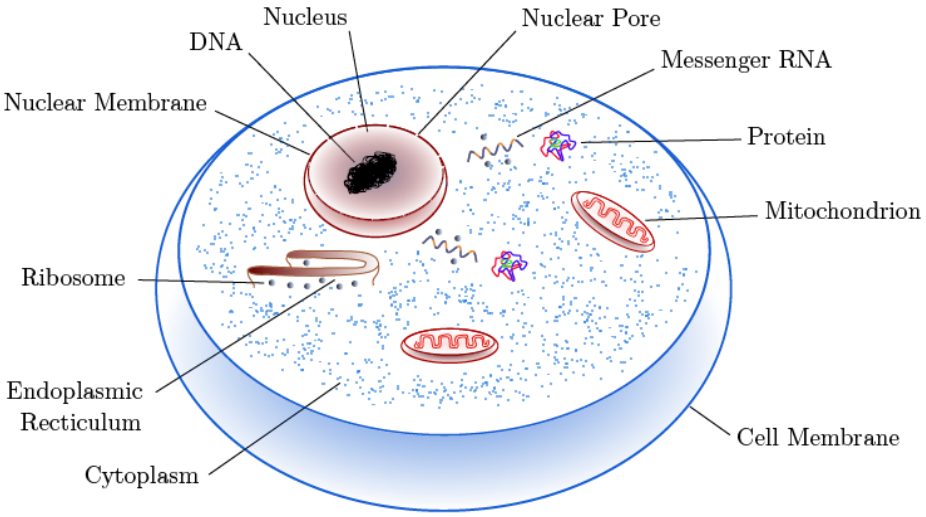


Figure 2.2: Transcription and translation of mRNA into protein.

The transcription process of the gene into mRNA is regulated by factors known as transcription factors [7]. The transcription factors bind to upstream promoter elements (UPEs) or an enhancer which increases the accuracy and rate of mRNA synthesis respectively [6]. The transcription factors can also be used to repress the expression of a certain gene. The gene expression profile therefore provides information about the

biological state of the cell, and is measurable through the concentration of the respective mRNA molecules produced by a cell.

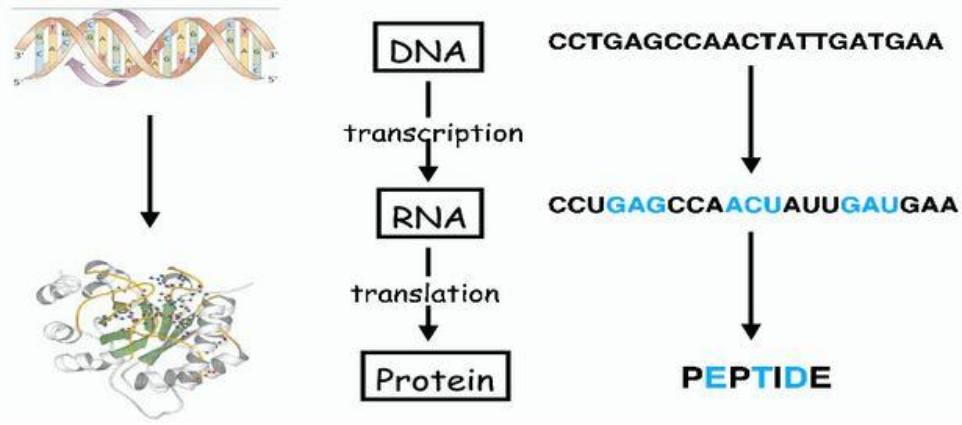


Figure 2.3: Central Dogma of Molecular Biology

The *Central Dogma* is the transcription of DNA to RNA and subsequent translation of RNA to protein, as shown in *Figure* 2.3. In the transcription phase, the enzyme RNA polymerase creates a copy of a gene from the DNA to *messenger* RNA (*m*RNA) inside the nucleus. The *m*RNA travels from nucleus to the cytoplasm for protein synthesis, where it then binds with ribosome, a complex molecule based on *ribosomal* RNA (*r*RNA) and proteins. In the translation phase, the *m*RNA is used as a blueprint for the production of a protein. The *m*RNA moves along the protein synthesis site i.e. ribosomes, with a set of three-nucleotides called codons. *Transfer* RNA (*t*RNA) provides a compatible anticodon and is hybridised onto the *m*RNA. Finally, the amino acids bound to the RNA form polypeptide chain. This process continues until the translation process reaches a stop codon, which terminates the polypeptide synthesis. The entire process is called gene expression.

2.2 Microarray Data – Generation and Analysis

The importance of microarray technology lies in the fact that microarrays can measure the expression levels for thousands of genes simultaneously during essential biological processes across collections of related samples [8]. Specifically the microarray measures the amount of mRNA in a cell, which is quantitatively related to

the amount of protein synthesised [8]. The amount of mRNA for various genes is assumed to be directly proportional to the gene expression levels [9].

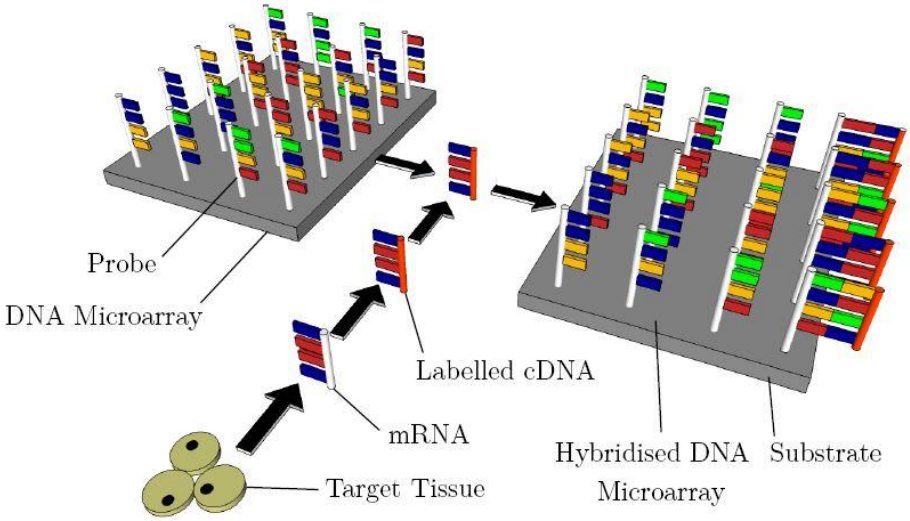


Figure 2.4: Illustration of the DNA microarray

The basic structure of the DNA microarray, shown in *Figure* 2.4, consists of a substrate (silicon, glass or plastic) onto which single stranded DNA molecules, each with different sequences, are deposited [9]. The single stranded DNA molecules are referred to as probes, and are arranged in a regular grid-like pattern on the substrate [10]. The types of probes deposited on the substrate depend on the purpose of the array.

One of the most popular ways to measure the gene expression in a microarray is to compare the expression level of a set of genes from a cell maintained in a particular condition (test condition) to the same set of genes from a reference cell maintained under normal conditions (normal condition).

The procedure firstly involves extracting the mRNA molecules [11] of a biological sample and then reverse transcribing them into complementary DNA (cDNA) sequences. The sample containing these cDNA molecules is often referred to as the target [9]. The target sample is then transcribed back to cRNA that is labelled with biotin. The solution is then placed onto the array where it diffuses and hybridises to the corresponding probes. The mixture is then washed, stained and finally exposed to an appropriate light source with the correct wavelength for excitation of the dye. The image captured contains multiple features, or hybridised spots, with the intensity of

each feature related to the amount of mRNA [8]. The various steps of a microarray expression study are shown in *Figure* 2.5.

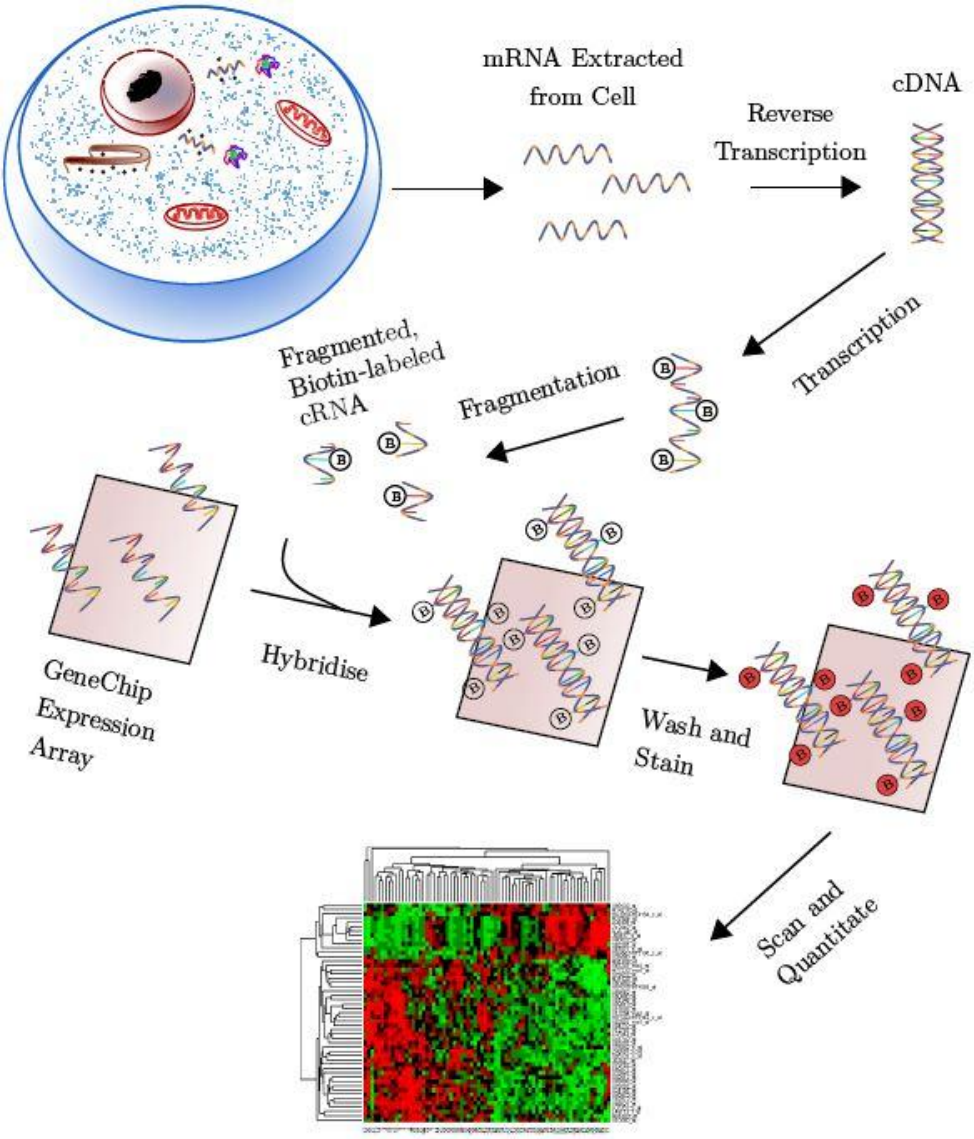


Figure 2.5: Oligonucleotide array with the steps involved in an expression study

The DNA microarray is therefore a useful and capable tool for measuring the large amounts of data embedded in the human genome. The most important aims when analysing a gene expression experiment, as mentioned by Domany [7], can be summarised as follows:

1. Identifying the genes that are associated with cancers and other important processes by using their expression profiles.

2. Partition tumours into classes based on their expression profiles and in familiar clinical classification. Expression profile classification can be used as a diagnostic or therapeutic tool.

3. Use the data analysis to obtain information relating to the unknown functions of certain genes.

In the field of molecular biology, gene expression profiling is the measurement of the activity (the expression) of thousands of genes at once, to create a global picture of cellular function. Gene expression profiling of cancer tissues is expected to contribute to the understanding of cancer biology as well as development of new methods of diagnosis and therapy.

Though the assembly of time-series data collected through repeated sampling across an entire disease process would provide essential information for the lucid understanding of the system, it is ethically infeasible to collect time-series data to study disease progression due to the need for immediate treatment upon diagnosis. Oncogenesis [3,12] is the process by which normal cells acquire the properties of cancer cells leading to the formation of a cancer or tumour. Hence a static sample can be regarded as a snapshot of the dynamic cancer process and so it is possible to construct a cancer progression model using data acquired from static samples [13].

In the cancer gene expression database (CGED), the gene expression data is derived from static expression experiments that analyze samples from many individuals. These samples are often snapshots of the progression of a disease such as cancer [14] and used for the purpose of data mining. Presently, cancer research makes use primarily of DNA microarrays but due to lowering costs, a technique superior to microarray technology known as RNA-Sequencing [15,16], is becoming more common as a method for cancer gene expression profiling.

2.3 Gene Expression Data

A microarray experiment is generally carried out to monitor the expression level of genes at a genome scale. The processed data can then be represented in the form of a

matrix, called gene expression matrix, where each row corresponds to a particular gene and each column could either correspond to an experimental condition or a specific time point at which expression of the genes has been measured. The expression levels for a gene across different experimental conditions are cumulatively called the gene expression profile, and the expression levels for all genes under an experimental condition are cumulatively called the sample expression profile.

An expression profile (of a gene or a sample) can be thought of as a vector [17] and can be represented in vector space. For example, an expression profile of a gene can be considered as a vector in *n* dimensional space (where *n* is the number of conditions), and an expression profile of a sample with *m* genes can be considered as a vector in *m* dimensional space (where *m* is the number of genes). In the example given below, the gene expression matrix *X* with *m* genes across *n* conditions is considered to be an *m* x *n* matrix, where the expression value for gene *i* in condition *j* is denoted as x_{ij}.

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}$$

The expression profile of a gene *i* can be represented as a row vector:

$$G_i = [\, x_{i1} \quad x_{i2} \quad \cdots \quad x_{in}]$$

The expression profile of a sample *j* can be represented as a column vector:

$$G_j = \begin{bmatrix} x_{1j} \\ x_{2j} \\ \vdots \\ x_{mj} \end{bmatrix}$$

2.4 Data Mining in Gene Expression Data Analysis

Analysis of gene expression data can be classified into two main categories namely supervised and unsupervised approach.

2.4.1 Supervised Approach

In a supervised approach or classification, prior knowledge about genes is directly exploited by the learning algorithm, known as the *learner*. The learner is trained by a *teacher* (the classification labels), to identify a particular class of a gene to which it belongs and assign the gene to the set of classes. In the presence of proper training samples, supervised methods can yield very high performance in grouping genes with particular functions together. However, obtaining labelled data is a very tedious, time-consuming and costly task and is sometimes not even possible due to the dependence on human annotators. In the case of supervised learning, the annotation of either the gene or the sample can be used and clusters of genes or samples can be created in order to identify patterns that are characteristic for the cluster. For example, sample expression profiles can be separated into 'disease state' and 'normal state' groups, and then one can look for patterns that separate the sample profile of the 'disease state' from the sample profile of the 'normal state'.

Some of the most popular algorithms under this approach belong to the categories of decision tree, instance-based and Bayesian Networks.

Decision tree learners such as C4.5 [18] use a method known as *divide and conquer* to construct a suitable tree from a set of labelled training data using the concept of information entropy. The divide and conquer algorithm partitions the data until every leaf contains one case, or until further partitioning is not possible because two cases have the same values for each attribute but belong to different classes. CART [19] (Classification And Regression Trees) is a robust classifier for any real-life application that attempts to construct an optimal decision tree to classify new instances. The decision tree J48 implements the C4.5 algorithm for generating a pruned or unpruned C4.5 tree. However, decision tree learners suffer from limitations such as (i) empty branches with many nodes having zero or close to zero values which makes the tree bigger and more complex, (ii) insignificant branches, since all the selected discrete attributes used to build a decision tree are not significant for classification task, and (iii) over-fitting, that takes place when the model tries to correctly classify all training cases in the absence of conflicting cases. *Nearest neighbour* or *instance-based* classifiers [20] such as the *k*NN classify unknown instances

by computing the *k* closest neighbours of an instance of an unknown class and the class is assigned by voting among those neighbours. The limitations of such algorithms are that (i) all the training samples are stored and a classifier is not built until a new (unlabeled) sample needs to be classified, (ii) additional computational costs is incurred due to comparison of the new unlabelled data with the stored training samples, and (iii) they assign equal weight to each attribute in the data irrespective of the relevance of the attribute. A *Bayesian Network* (BN) is a directed acyclic graphical model that depicts the probability relationships among a set of variables features. Naïve Bayes classifier [21] is a probabilistic classifier that takes into consideration that all attributes (features) independently contribute to the probability of a certain decision with equal importance. However, they are (i) not suitable for datasets with many features due to the infeasibility in terms of time and space, and (ii) before the induction, the numerical features need to be discredited in most cases.

2.4.2 Unsupervised Approach

The unsupervised approach involves the use of clustering algorithms to identify genes with similar expression patterns. Clustering is unsupervised because the learning methods try to find an interesting structure in the gene expression data in the absence of training samples and without any knowledge of the genes' functions. For example, if in a given cluster many genes are found to be in the same class, it may be hypothesized that the genes have some functional or regulatory relationship. Unlike supervised learning, there is no teacher in unsupervised learning to provide the true classifications, which is also convenient as unannotated data can be utilized [22]. The expression data is analysed to identify patterns that can group genes or samples into clusters without the use of any form annotation. For example, genes with similar expression profiles can be clustered together without the use of any annotation. However, annotation information may be taken into account at a later stage to make meaningful biological inferences. This issue is addressed by using partitional, hierarchical, density-based, model-based or subspace clustering algorithms, based on the proximity between genes or conditions in the expression matrix.

In the case of *partitioning approaches*, algorithms like PAM [23] and CLARANS [24] have been observed to be robust. However, they suffer from limitations such as (i) the

number of clusters are to be known *apriori*, (ii) the proximity measure used may be inadequate in finding the 'true' number of biologically relevant clusters. Similarly, the algorithms following the *hierarchical approach* such as CURE [25], ROCK [26] and BIRCH [27] can be found advantageous from the biologists' perspectives as they help to represent the cluster-cluster association, apart from individual co-expressed gene group representation. However, they also suffer from limitations such as (i) difficulty in deciding the appropriate stopping criteria, and (ii) simultaneous representation of disjoint, embedded and intersected clusters. *Density-based approaches*, DBSCAN [28] and DENCLUE [29] have already been established as being good at finding clusters of all shapes. They are capable of identifying global as well as local (embedded) clusters. However, two limitations of such approaches are that (i) they are input parameter sensitive, and (ii) ineffective in finding intersected patterns over high dimensional data. A *model-based approach*, such as COBWEB [30] and SOM [31] provides an estimated probability that a single gene may exhibit membership in more than one cluster indicating that a gene may have a high correlation with two totally different clusters. It discovers good values for its parameters iteratively and can handle various shapes of data. However, it suffers from the limitations that (i) it can be computationally expensive since a large number of iterations may be required to find its parameters, and (ii) it assumes that the dataset fits a specific distribution which is not always true. *Graph theoretic* algorithms are suitable for subspace and high dimensional data clustering, such as CLICK [32] and CAST [33]. The algorithms under this approach do not require a user-defined number of clusters, can handle outliers efficiently and they are capable of discovering intersected and embedded clusters. However, they are limited by (i) the difficulty faced in determining a good threshold value, and (ii) they require apriori knowledge of the dataset. Among the *soft computing* approaches, Fuzzy C-Means (FCM) [34] and Genetic Algorithms (GA) such as GENCLUST [35] have been used effectively in clustering gene expression data. The Fuzzy C-Means algorithm requires the number of clusters as an input parameter. The GA based algorithms can detect biologically relevant clusters but are dependent on proper tuning of the input parameters.

It has been found that partitioning approach is not suitable for gene expression data because the number of clusters '*k*' is not known *apriori*. Also, the hierarchical

approach has its limitations since determining the termination criteria is difficult. The density based, model based and graph theoretic based approaches have been found to be more suitable for clustering gene expression data.

2.4.3 Discussion

Generating high-quality gene clusters and assigning data objects to a set of classes with an aim to identify the underlying biological mechanism of the gene clusters are the important goals of clustering gene expression data. It is essential to have relatively high-quality clusters first, in order to get a correct, informative biological explanation of the gene cluster. To get high-quality cluster results, most of the current approaches rely on choosing the best cluster or classification algorithm, in which the design biases and assumptions meet the underlying distribution of the dataset. Hence a clustering algorithm attempts to organize the data based on (i) an internal criterion, (ii) the characteristics of the used (dis)similarity function and (iii) the dataset itself.

The existing clustering/classification approaches are not without their biases and limitations, as detailed below.

- **Prior information of the dataset**: Different clustering solutions may seem equally plausible without a priori knowledge about the underlying data distributions. Usually, the underlying data distribution of the gene expression datasets is unknown. Every clustering algorithm implicitly or explicitly assumes a certain data model and it may produce erroneous or meaningless results when these assumptions are not satisfied by the sample data. Thus, the availability of prior information about the data domain is crucial for successful clustering, though such information can be hard to obtain, even from experts.

- **The best clustering algorithm**: Many clustering algorithms are available and different clustering algorithms may generate different clustering results in the same dataset due to their bias and assumptions. It is well known that no single clustering algorithm performs best across various datasets. Therefore, it is a challenging and daunting task for genomic researchers to choose the best clustering algorithm for a particular gene expression dataset, since the results of different clustering algorithms may not be consistent.

- **Evaluation of Results**: Quite naturally, two different clustering algorithms when applied to the same dataset can produce different results. One way to evaluate the results is to use cluster validity indexes [36] but, again, it may not be an impartial evaluation of the clustering results. The probability may be that different solutions obtained by different clustering algorithms may be equally possible, especially in the absence of any previous knowledge about the best way to evaluate the results.

- **Noise in the gene expression dataset**: The results of clustering algorithms are easily corrupted by the addition of noise, which is very common in gene expression analysis as the experimental measurement may not be very accurate or error may be introduced by the data transformation. Therefore, obtaining high quality clustering results in presence of noise in the gene expression data is a very challenging task.

- **Similarity Measure**: Many clustering algorithms require a definition of a metric to compute the distance between data points. Thus, their performance is often directly influenced by the dimensionality of the dataset used for calculating the chosen distance metric.

- **Repeated runs**: Cluster analysis frequently involves repeated runs of different clustering algorithms with random restart, followed by a selection of an individual solution that maximizes a user defined criterion.

- **Biological Interpretation**: Another drawback is that the clustering quality and cluster interpretation are treated as two isolated research problems and are studied separately. But cluster quality and cluster interpretation are closely related and must be addressed in a coherent and unified way. It is essential to have relatively high-quality clusters first in order to get a correct, informative biological explanation of the gene cluster. Otherwise, the biological explanation will be incorrect or misleading, no matter how good or robust the text summarization technique is.

Hence, one requires an effective combination of several clustering algorithms to improve the clustering quality, which brings us to the question as to how to combine

different clustering results and how to ensure symmetrical and unbiased consensus with regard to all the component partitions.

2.5 Ensemble Approach

Existing classification and clustering algorithms alone cannot provide accurate analysis of high dimensional data, whereas ensemble methods, after combining the output of several algorithms, are able to improve the robustness and stability of the analysis and thus improve the overall prediction accuracy. Ensemble methods should combine the strengths of the individual clustering algorithms to provide an overall improved clustering of the dataset, which can go beyond what is typically achieved by a single clustering algorithm [37]. The result of an ensemble process is a consensus and combination of all the individual base algorithms that takes care of the possible errors made during clustering by a single algorithm, Moreover, cluster ensembles can also address the problem of local optima and obtain a globally optimum solution i.e. a stable clustering solution. More importantly, ensembles exempt the user from deciding on a particular clustering algorithm, thereby avoiding the risk of a poor choice, though it should be applied and interpreted with caution [22]. Hence ensemble-based methods are preferred for cancer classification considering the "curse of dimensionality" and the small sample size. A high degree of accuracy and a low computational complexity is critical for robust and accurate cancer classification [38].

An ensemble is expected to be:

- **Robust**, showing better than average performance than the single clustering algorithms
- **Consistent**, the result should be similar to the results of the individual clustering algorithm
- **Novel**, in uncovering solutions unattainable by single clustering algorithms
- **Stable**, i.e., should not be sensitive to noise and outliers

Ensemble approaches can be of three types: supervised, unsupervised and semi-supervised. The main features of these approaches are summarised in *Table* 2.1.

Table 2.1: Comparison among supervised, unsupervised and semi-supervised ensemble approaches

Supervised	Unsupervised	Semi-Supervised
• uses labelled data	• uses unlabeled data	• uses unlabeled data to either modify or reprioritize hypotheses obtained from labeled data
• makes use of classification algorithms	• makes use of clustering algorithms	• uses both classification and clustering algorithms
• can yield high performance in the presence of proper training samples (i.e. labelled data)	• generating high quality clusters is biased towards the dataset and input parameters	• attempts to combine all the supervised and unsupervised results by consensus
• obtaining labelled data is a daunting task, requires the efforts of human annotators	• suffers from label correspondence problem due to use of unlabeled data	• Unlabeled data may have a high likelihood of wrong interpretation if wrong model is selected

2.5.1 Supervised Ensembles

Research on ensemble based classifiers and their use have expanded rapidly in recent times and researchers have used many terms to describe combining models involving different learning algorithms. Elder and Pregibon [39] used the term 'Blending', Dietterich [40] called it an 'Ensemble of Classifiers', Steinberg [41] termed it a 'Committee of Experts', while Breiman [42] referred to it as 'Perturb and Combine (P&C)'. Several other terms can also be found in the literature [43]. However, the concept of combining models is actually quite simple: train several models using the same dataset, or from samples of the same dataset and combine the output predictions, typically by voting (for classification problems) or by averaging output values (for estimation problems).

2.5.1.1 The Ensemble Framework

The task of constructing an ensemble can be broken down into two subtasks: (i) selection of a diverse set of base level models or classifiers with consistently acceptable performance, and (ii) appropriate combination of their predictions with due weightage.

The building blocks for a classifier ensemble are as follows.

i. *Training set*, which is a labeled dataset used for ensemble training where the instances are described as attribute-value vectors.

ii. *Base Inducer* is an induction algorithm that obtains a subset of attributes of the

training set to form a classifier.

iii. *Diversity Generator* is responsible for generating diverse classifiers.

iv. *Combiner* combines the classifications of the various classifiers.

The building blocks for a classifier ensemble are shown in *Figure* 2.6. The dataset(s) D_1, D_2, …, D_n may be considered to be either multiple datasets or may also be considered as individual samples drawn from a single dataset.

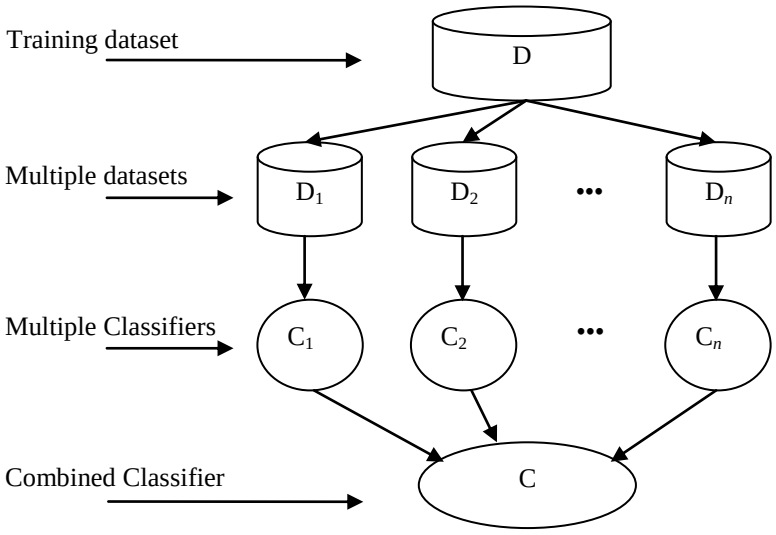


Figure 2.6: The general process of Classifier Ensemble

2.5.1.2 Ensemble Diversity

Diversity is the degree to which classifiers disagree in the errors they make. This allows the voted accuracy to be greater than the accuracy of any single classifier. It is well known that the combination of the output of several classifiers is only useful if they disagree on some inputs [44,45]. Constructing a diverse committee in which each hypothesis is as different as possible, while still maintaining consistency with the training data, is known to be a theoretically important property of a good ensemble method [46]. Some methods of achieving diversity are mentioned below.

Different classifier models: For effectiveness, one can use several types of learning algorithms from different backgrounds, e.g., decision trees, neural networks and nearest neighbour classifiers. However, the same classifier can also be used with a

slight change in the user-defined parameters, leading to significant variation in classification results.

Different feature subsets: Classifiers may be built using different subsets of features of the training dataset. An ensemble works only when some redundancy is present in the features in the training dataset. Deterministic and random approaches can be used for selecting different feature subsets of the input data. In the deterministic approach, prior knowledge of the input data is required, whereas the random approach uses a random subspace method for selecting feature subsets.

Different Training sets: A single learning algorithm is run on different random sub-samples of the training data to produce different classifiers. It works well for unstable learners, when the output of the classifier undergoes major changes given only small changes in the training data. Random sub-samples of the training data can be generated using re-sampling and re-weighing.

2.5.1.3 Combination methods

Some considerations in combining the results of the base-classifiers are the following. (i) Weighting methods are suitable when the same task is achieved with similar amounts of success by the base classifiers, (ii) Continuous output methods are used when each classifier outputs a vector of continuous-valued measures that can represent estimates of class posterior probabilities or class-related confidence values that represent support for possible classification hypotheses, and (iii) Meta-learning methods are useful when the instances are consistently classified or misclassified by the base line classifiers, such as in stacked generalization [47].

2.5.1.4 Popular Ensemble Methods

Classification tasks are generally improved by creating an ensemble or committee of base classifiers and their output is combined using some form of consensus, to arrive at a prediction for unseen data. Though this helps in achieving a more accurate classification of unknown data, it is at the expense of increased model complexity [48]. The generalization property of the ensemble approach is explained using the classic bias-variance decomposition analysis [49]. Bias is a measure of the quality or accuracy

of an algorithm, whereas variance is a measure of the specificity or precision of a match. A high bias and a high variance is an indication of a poor match; hence minimization of bias and variance is needed. Since this cannot be achieved independently, there is a trade-off between the two. Methods like bagging improve generalization by decreasing variance [50] while methods similar to boosting achieve this by decreasing bias [51].

Boosting

Boosting [52] is a model averaging method where one first creates a 'weak' classifier. A weak classifier is one whose classification performance is only slightly better than any random classifier of the ensemble committee. The models are built successively with each one being trained on a dataset in which the misclassified instances of the previous model are given more weight. Finally, the outputs of all the successive models are combined using voting according to their weights, into one classifier whose accuracy is higher than that of the individual classifiers. The original boosting algorithm combined three weak learners to generate a strong learner.

Adaptive Boosting, or *AdaBoost* [53], improves the boosting algorithm by iteratively increasing the weights of all misclassified instances, while the weights of correctly classified instances are decreased. *Arcing* [54] is a form of boosting that weighs incorrectly classified cases more heavily. A distributed version of AdaBoost, the *P-AdaBoost* algorithm [55] executes the AdaBoost algorithm for a limited number of steps to yield the estimated weights of the instances which are then used to train the classifiers. Zhang and Zhang [56] proposed a *boosting-by-resampling* version of Adaboost, where a local error is calculated for every training instance which is then used to update the probability so that the current instance is chosen for the training set for the next iteration.

Bagging

In view of the significant improvement in the classification accuracy by combining classifiers, Breiman [42] introduced the method of *Bagging*. Bagging, like boosting, is a technique that improves the accuracy of a classifier by generating a composite model

that combines multiple classifiers, all of which are derived from decision tree models generated from bootstrap samples (with replacement) of a training dataset. Both methods follow a voting approach, though implemented differently, in order to combine the outputs of the different classifiers. In bagging, each instance is chosen with equal probability, while in boosting, instances are chosen with a probability that is proportional to their weight. Furthermore, as mentioned earlier, bagging needs to have an unstable learner as the base inducer, while in boosting inducer instability in not necessary, only that the error rate of every classifier needs to be kept below 0.5. *Wagging* [57] is a variant of bagging in which each classifier is trained on the entire training set, but each instance is stochastically assigned a weight.

Random Forests

Breiman [58] proposed the method of *Random Forests*, which uses a large number of individual, unpruned decision trees and adds an additional layer of randomness to bagging. In addition to constructing each tree using a different bootstrap sample of the data, random forests construct the classification or regression trees by splitting each node using the best predictor for that node from a randomly chosen subset. The classification of an unlabeled instance is performed using majority vote. Kamath and Cantu-Paz [59] proposed the use of a sub-sample of the instances to determine the best split point for each feature. The feature and split value that optimize the splitting criterion are chosen as the decision at that node. This technique results in different trees for different sub-samples since the split made at a node may vary with the sample. These trees can now be combined into ensembles. Another method for randomization of the decision tree through histograms was proposed in Kamath *et al*. [60] to make the features discrete, at the same time reducing computation time to handle large datasets.

Although random forests were defined for decision trees only, this approach is applicable to all types of classifiers. An advantage of the random forest method is its speed of computation and its ability to handle a very large number of input attributes.

Stacked Generalization

Stacking is a technique that combines the base classifiers through a meta-classifier to maximize the generalization for achieving the highest accuracy [47]. The method attempts to work out the reliability of classifiers for optimum accuracy by using a meta-learner. Stacking combines models built by different inducers to create a meta-dataset containing a tuple for every tuple in the original dataset. It uses the predicted classifications by the classifiers as the input attributes in place of the original input attributes. A test instance is first classified by each of the base classifiers. These classifications are fed into a meta-level training set from which a meta-classifier is produced. This classifier combines the different predictions into a final one. The original dataset is partitioned into two subsets such that one subset is reserved to form the meta-dataset and the other subset is used to build the base-level classifiers. The performance of the base-level learning algorithms in correctly classifying the instances is reflected in the meta-classifier predictions. The performance of stacking can be improved by using output probabilities for every class label from the base-level classifiers.

Besides the development of more effective ensemble methods, current studies also focus on more objective comparison [61]. For example, a recent study by Ge and Wong [62] compared the single classifier of decision trees with six ensemble methods including random forests, stacked generalization, bagging, Adaboost, LogitBoost, and Multiboost using three different feature selection schemes (Student t-test, Wilcoxon rank sum test, and genetic algorithms). Wang *et al*. [63] employed stacked generalization to predict membrane protein types. A SVM and a kNN were used as the base classifiers and a decision tree was adopted to combine the base classifiers. Netzer *et al*. [64] developed a feature selection approach using the principle of stacked generalization. The feature selection algorithm termed stacked feature ranking is reported to identify important markers and improve sample classification accuracy

2.5.1.5 Comparison of the Four Supervised Ensemble Methods

The advantages of using the three basic supervised ensemble methods are summarized in *Table* 2.2.

Table 2.2: Comparison among the four major supervised ensemble methods

Advantages of			
Bagging	**Boosting**	**Random Forests**	**Stacked Generalization**
• works by reducing variance by voting	• improves generalization by decreasing bias	• robust, fast and accurate	• uses diverse base classifiers
• improves performance if the learning algorithm is unstable	• simple and easy to implement and can be applied on a wide variety of problems	• can estimate missing data and maintain accuracy	• uses a meta-learner to achieve high accuracy
• Simple and easy to understand and implement	• non-parametric and flexible, can work with any learning algorithm	• runs efficiently on large data bases and can handle a large number of input attributes	• shows good performance in lieu of storage and time complexity

2.5.1.6 Selection of Size of Ensemble Basket

Supervised ensemble selection is important for two reasons: efficiency and predictive performance [65]. A large ensemble incurs a higher computational cost than a smaller one. The main criterion for constructing an efficient ensemble is determining the number of base classifiers in the ensemble committee. Generally speaking, there is no ensemble method which outperforms other ensemble methods consistently. Strategies for creation of ensembles include (i) using a parameter to control the number of iterations, which will then determine the ensemble size as in bagging, and (ii) determining the ensemble size at the time of training by monitoring the performance of the ensemble. An algorithm that decides whether a sufficient number of classification trees have been created using an out-of-bag error estimate, incorporating bagging for ensemble construction, was proposed by Banfield *et al.* [66]. Rokach [67] proposed pruning of the ensemble after letting it extend in an unlimited manner in order to get a more effective and compact ensemble. Liu *et al.* [68] showed that a small ensemble can be constructed from a larger one while maintaining the accuracy and diversity of the full ensemble. Empirical studies conducted by Margineantu and Dietterich [69] suggest that pruned ensembles may be more accurate than the original ensemble.

Earlier, it was thought that using more base learners will lead to a better performance. However, Zhou *et al.* [70] proved the "many could be better than all" theorem which suggested selection of a few base learners instead of all to compose an ensemble. Such ensembles are known as *selective ensembles.*

2.5.1.7 Choosing the Best Ensemble Method for a Problem in Hand

Choosing the best ensemble method is a MCDM (Multiple Criteria Decision Making) problem since there is trade-off in the relationships among the criteria and some criteria cannot be measured in proportionate units.

The main selection criteria include [71]:

- *Accuracy* of classification shown by the ensemble
- *Computational Cost* for constructing the ensemble and also the time required for classifying an unseen instance
- *Scalability* of the ensemble method to work with large datasets
- *Flexibility* in order to provide a solution to binary and multiclass classification tasks
- *Usability* in terms of controllable parameters that are comprehensive and can be easily tuned
- *Interpretability* of the ensemble results
- *Software Availability* so that the practitioner can move from one software to another, without having to replace his ensemble method

Since there are different ensemble approaches available, it becomes difficult for a researcher to make an informed decision regarding the choice of the correct ensemble method. An appropriate ensemble technique can be selected considering the problem at hand, by keeping in mind the above mentioned selection criteria.

2.5.1.8 Use of Supervised Ensemble Methods in Microarray Data

Bagging and Boosting methods were applied for classification of normal and tumor cells using gene expression data by Ben-Dor *et al.* [72] and Dudoit *et al.* [73]. LogitBoost [74] gave a more accurate classification of gene expression data by replacing the exponential loss function used in AdaBoost with a log-likelihood loss function. To improve the performance of AdaBoost, Long [75] proposed several customized boosting

algorithms for the base classifiers used in AdaBoost for microarray data classification. By comparing the performance of bagging and boosting to a single tree classifier using seven publicly available datasets, Tan and Gilbert [76] demonstrated that the ensemble methods are more robust and accurate in microarray data classification.

The multiple feature subsets used in random forests are appropriate for high-dimensional microarray data, as shown by Lee *et al.* [77] in an experimental comparison of bagging, boosting and random forests. The experimental results using ten microarray datasets by Diaz-Uriarte and de Andres [78] suggest that random forests can accurately interpret data and at the same time output smaller gene sets as compared to other methods. Izmirlian [79] also pointed out a few other advantages of random forests when used for microarray data, such as robustness to noise, independence from tuning parameters and a favourable speed of computation.

Variants of random forests have also performed well in classification of microarray data. The results obtained by Zhang *et al.* [80] suggest that a deterministic forest of classification trees show better interpretation of high-dimensional data while the performance is comparable to that of random forests. A tree ensemble method called extra-trees proposed by Geurts *et al.* [81] has shown an improvement over random forests in performance.

2.5.2 Unsupervised Ensembles

The purpose of a cluster ensemble is to build a robust clustering portfolio that can perform as well, if not better, than a single best clustering algorithm across a wide range of datasets. Different clustering algorithms take different approaches. Hence a cluster ensemble can be used to generate cluster results using various clustering algorithms and then the results can be integrated using a consensus function to yield stable results. Given a set of objects, a cluster ensemble consists of two principal steps: (i) *Ensemble construction*, which is creation of a set of partitions using some clustering algorithm, and (ii) *Consensus Function*, where a new partition is obtained from the individual partitions of step (i).

2.5.2.1 Cluster Ensemble Framework

The two step process of a cluster ensemble, i.e. Ensemble Construction and Consensus Function is shown in *Figure* 2.7.

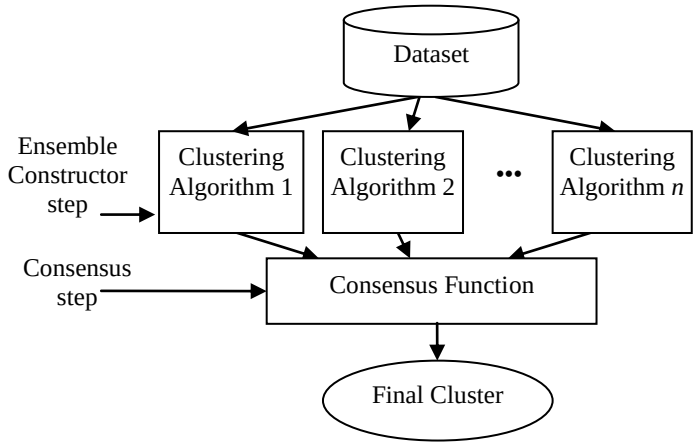


Figure 2.7: The general process of Cluster Ensemble

2.5.2.2 Ensemble Constructor

The set of clusters that will be combined is generated in this step. The selection of an appropriate generation mechanism is important as the final result will depend on the clusters generated in this step. The generating process should be diverse to compensate for clustering error of one algorithm by another.

Topchy *et al.* [82] demonstrated that during the ensemble constructor step when a proper consensus function is applied to weak clustering algorithms, these simple and fast procedures can produce quality consensus clusters. Hence, one should make a judicious choice of diverse clustering algorithms to uncover more information about the data.

Diversity in the individual clustering of a given dataset can be achieved by a number of approaches, such as

a) Using different clustering algorithms to produce partitions for combination [83,84],

b) Changing initialization or other parameters of a clustering algorithm [85,86],

c) Using different features via feature extraction for subsequent clustering [82,83],

d) Partitioning different subsets of the original data [84,87,88] and

e) Projecting data onto different subspaces [83,89].

These diversity generation mechanisms are presented in *Figure* 2.8.

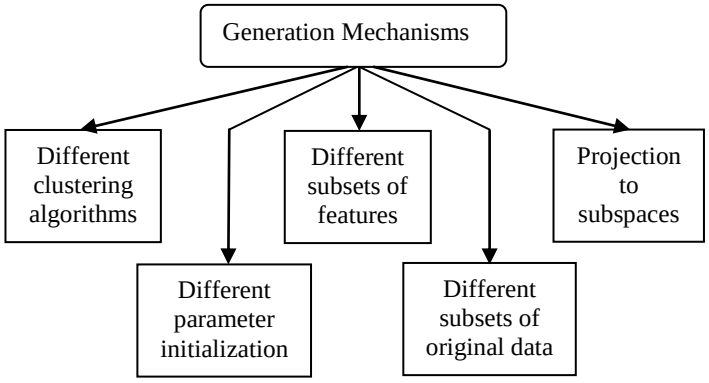


Figure 2.8: Clustering Ensemble generation mechanism

2.5.2.3 Consensus Functions

Finding consensus among the individual clustering algorithms is a challenging and daunting task. The consensus function should be capable of improving the results, leading to two main consensus function approaches.

a) The *Object co-occurrence* approach deals with the association of a cluster label with each object. This is achieved by analyzing the number of times an object belongs to one cluster or the number of times two objects belong to the same cluster together. The consensus is obtained by a voting process among the objects. The problems faced in the objects co-occurrence approach are the selection of an appropriate clustering algorithm and arriving at the correct parameters.

b) A *Median partition* is defined as the partition that maximizes the (dis)similarity with all other partitions in the cluster ensemble. The main problem here is selecting the correct (dis)similarity measure to solve the problem or to come close to the solution.

Some of the main (dis)similarity measures used to compute the distance between partitions are the following.

- *Counting pairs*: These measures count the pairs of objects on which two partitions agree or disagree, e.g., Rand index [90], Jaccard coefficient [91], Mirkin distance [92] and their variations.
- *Set matching*: These measures are based on set cardinality comparisons, e.g., Purity and Inverse Purity [93], F-measure [94] and Dongen measure [95].
- *Information Theory based*: These measures quantify the information shared between two partitions, e.g., Class Entropy [96], Normalized Mutual Information [97] and Variation of Information [98].
- *Kernel measures* are defined specifically for the median partition problem, e.g., a Graph Kernel based measure [99] and a Subset Significance based measure [100].

In principle, it is not necessary for a particular consensus clustering to strictly follow the object co-occurrence approach or the median partition approach. For example, there could be two consensus clustering methods based on genetic algorithms, one following the co-occurrence approach and the other the median partition approach. *Figure* 2.9 presents the main consensus functions.

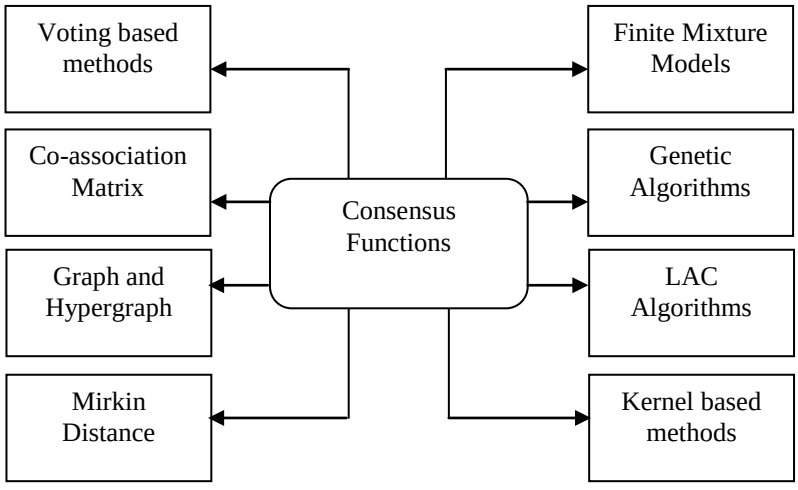


Figure 2.9: Principal consensus functions techniques

Voting-Based Methods

Label correspondence, i.e., associating a cluster label with an object, is one of the main reasons that make unsupervised clustering ensembles difficult to build. It appears when there is no relationship between the sets of labels given by different

individual clustering algorithms. The Voting approach attempts to solve this problem by using heuristics such as *bipartite matching* and *cumulative voting*.

A voting consensus algorithm using the assumption that the number of clusters in each partition is the same as the final number of clusters in the consensus partition was proposed by Dudoit and Fridlyand [84] and Fischer and Buhmann [101]. It is similar to the plurality voting used in a supervised classifier ensemble [102], and is applied to obtain the final cluster for each object, after solving the labeling correspondence problem through maximum likelihood computation using the Hungarian [103] method.

The Voting-Merging [104] method combines clusters in a two step process: a voting process is used to solve the label correspondence problem, followed by the merging of the votes to decide the final partition.

Co-Association Matrix Methods

To avoid the label correspondence problem, co-association methods [105] are used to map the partitions in the cluster ensemble into an intermediate representation: the co-association matrix. This matrix can be viewed as a new similarity measure between two objects from the set of objects *O*. Objects o_i and o_j are similar if they tend to appear together in the same cluster. Using the co-association matrix as the similarity measure between objects, the consensus partition is obtained by applying a clustering algorithm.

Fred and Jain [105] proposed an algorithm where the co-association matrix is viewed as an adjacency matrix of a graph. From this a minimum spanning tree is obtained, one that contains all the nodes of the graph with the minimum weights in their edges. Then the weak links between nodes are cut using a threshold *r*. Algorithms such as *Single Link* (SL) [106], *Complete Link* (CL), *Average Link* (AL) and other hierarchical clustering algorithms can be used in variants of this method.

An algorithm based on the concept of normalized edges to measure the similarity between clusters was proposed by Li et al. [107]. It involves the application of a hierarchical clustering algorithm to the co-association matrix, to improve the quality of the consensus partition.

Graph and Hypergraph Methods

The combination problem uses a graph or hypergraph where the vertices represent the objects to be clustered and the objects on a hyperedge belong to the same cluster.

According to Strehl and Ghosh [97], the similarity measure between partitions decides the consensus partition, which is the information shared by all the partitions, measured by Normalized Mutual Information (NMI). They proposed three heuristics to obtain the consensus partition. The *Cluster-based Similarity Partitioning Algorithm* (CSPA) forms a hypergraph from a co-association matrix and then the consensus partition is obtained by partitioning this graph using the METIS [108] algorithm. In *HyperGraphs Partitioning Algorithm* (HGPA), the hypergraph is partitioned by eliminating the minimum number of hyperedges in *k* connected components of approximately the same dimension by the HMETIS [109] hypergraph partitioning package. The *Meta-CLustering Algorithm* (MCLA) first forms a similarity matrix between the clusters, which is the adjacency matrix upon which the graph is then constructed. The clusters obtained by partitioning this graph using the METIS algorithm are called meta-clusters. The final partition is obtained by assigning the object to the meta-cluster where it appears the maximum number of times.

Fern and Brodley [88] proposed the *Hybrid Bipartite Graph Formulation* (HBGF) algorithm where an edge exists between two nodes if one node represents a cluster and the other node represents an object belonging to this cluster. Then the METIS algorithm is used to obtain the consensus partition by partitioning this graph. Abdala *et al*. [110] proposed a graph based clustering ensemble algorithm building on the random walk algorithm for the combination of image segmentations [111].

Mirkin Distance-Based Methods

In these methods, the consensus partition is obtained by the solution to the median partition problem using the Mirkin distance as dissimilarity measure between partitions. Since it is suitable for a small number of objects and partitions, several heuristics have been proposed. Filkov and Skiena [112] proposed three heuristics: the output of the *Best-of-k* (BOK) heuristic is the partition in the cluster ensemble that

minimizes the distance from it to all the other partitions in the ensemble. *Simulated Annealing One-element Move* (SAOM) follows the idea of guessing an initial partition and iteratively changing it by moving an object from one cluster to another. *Best One-element Move* (BOM) also starts with an initial partition and generates new partitions by moving the object from one cluster to another, always checking if this partition is a better than the previous one.

Four new heuristics proposed by Gionis *et al*. [113] are the *Balls algorithm*, *Agglomerative algorithm*, the *Furthest algorithm* and *LocalSearch algorithm*. The Balls algorithm builds a graph where the edges are weighted by the distances between pairs of objects. The triangle inequality of the Mirkin distance is applied iteratively, yielding a new cluster for the consensus partition. The Agglomerative algorithm is based on the Average-Link agglomerative clustering algorithm. The Furthest algorithm starts by placing all the objects in one cluster and the objects which are the furthest away are placed in different clusters iteratively. The cost of the new partition is computed and the procedure is repeated until a worse solution is obtained. The LocalSearch algorithm starts with an initial partition and the cost of moving objects from one cluster to another is computed repeatedly until there is no move that can improve the cost. Two other algorithms used are the *CC-Pivot* [114] and *CCLP-Pivot* [114]. In the *CC-Pivot* algorithm, a partition is obtained using a relation between the objects and repeatedly selected pivot objects. *CCLP-Pivot* is a linear programming based version of the *CC-Pivot*.

Finite Mixture Model-Based Methods

A consensus function obtained as the solution of a maximum likelihood estimation problem using the EM [115] algorithm was proposed by Topchy *et al*. [116]. This approach is based on a finite mixture model for the probability distributions for assigning labels to the objects in the partitions.

Genetic Algorithm Methods

Consensus clustering is arrived at by determining which partitions of the set of objects (chromosomes) have the highest fitness value. The Heterogeneous Clustering

Ensemble proposed by [117,118] creates an ordered pair of partitions from the objects and a fitness value is computed for the comparison of the amount of overlap between the partitions in each chromosome. Luo *et al.* [87] proposed minimizing an information theoretical criterion using a genetic algorithm to obtain the consensus function. Analoui and Sadighian [119] used a finite mixture of multinomial distributions and the corresponding maximum likelihood problem is solved using a genetic algorithm to obtain the consensus function.

Locally Adaptive Clustering Algorithm Methods

Partitions obtained using Locally Adaptive Clustering (LAC) [120] algorithms can be combined to yield the consensus function. The dataset is generally made up of numerical data. Two consensus functions were proposed by Domeniconi and Al-Razgan [121]: In the *Weighty Bipartite Partition Algorithm* (WBPA), first the weighted distance of each object to every cluster in a partition is computed. Then the similarities between a pair of objects is calculated and stored in a matrix, which is repeated for all partitions. Next, a graph is built from this matrix and the METIS algorithm is used to obtain the consensus partition. The second consensus function algorithm, *Weighted Subspace Bipartite Partitioning Algorithm* (WSBPA), is based on the partitioning of a bipartite graph. A third heuristic proposed by Domeniconi and Al-Razgan [121] adds a weight factor to each cluster obtained by WBPA.

Kernel Based Methods

The *Weighted Partition Consensus via Kernels* (WPCK) algorithm was proposed by Vega-Pons *et al.* [100]. The consensus partition is defined through the median partition problem by using a positive semi-definite kernel [122] as a similarity measure between partitions. It follows the traditional methodology of the clustering ensemble algorithms but before combining, the relevance of each partition in the cluster ensemble is calculated in an intermediate step called the Partition Relevance Analysis, followed by the combination. Another clustering ensemble method called WKF was presented by Vega-Pons *et al.* [99], where the similarity measure is based on a graph kernel function and it was extended to GKWF to take care of categorical and mixed data [123].

2.5.2.4 Comparison of the Consensus Methods

A comparison based on the merits and demerits of the different consensus methods discussed in *Section* 2.5.2.3 is summarized in *Table* 2.3. For computational complexity, n is the number of objects, m is the number of partitions and k is the number of clusters in the consensus partition. The quadratic cost on the number of objects is used as a threshold to determine whether an algorithm has high or low computational complexity. The value *heuristic* is applied when it is very difficult to determine the computational complexity, since it is not easy to determine how many steps are needed to reach a convergence criterion.

2.5.2.5 Improving the Combination Process

Generally, clustering ensemble algorithms give equal importance to all partitions obtained in the ensemble construction step and hence use all of them in the consensus step. However, in particular situations, all clusters in the cluster ensemble may not have the same quality, i.e., the information that each one contributes may not be the same. Therefore, a simple average of all clusters does not have to be the best choice.

To account for difference in the quality of clusters, there are two approaches. Both approaches inspect the generated partitions and make a decision that assists the combination process. The first consists of selecting a subset of clustering algorithms to create an ensemble committee, whose results will be combined to obtain the final solution. The other approach consists of setting a weight to each partition in order to give a value according to its significance in the clustering ensemble.

These two techniques do not have to be exclusive. A selection based on weighting the partitions already selected could be performed. Any of these variants could improve the quality of the final result. However, their use implies extra computational cost. Therefore, in a practical problem, the user should analyze the characteristics of the problem at hand, and decide whether to use a clustering discrimination technique or not, according to their requirements.

Table 2.3: Comparison among different approaches of consensus function

Consensus Function	Merits	Demerits	Computational Complexity
Relabeling and Voting	• easy to understand and implement • suitable when the labels associated with each object is the same for all clustering algorithms	• labelling correspondence problem makes the combination of clusters difficult. • requires all partitions to have the same number of clusters • the label correspondence problem solved by the Hungarian algorithm leads to high computational cost	Heuristic dependent $O(k^3)$
Co-association matrix	• very easy to implement and understand	• final clustering depends on the similarity measure and the clustering algorithm applied • not suitable for large datasets	High $O(n^2)$
Graph and hypergraph	• popular method, easy to understand and implement • has low computational complexity	• not strictly used as a solution for consensus clustering but proposed as a solution for the median partition problem • the methods need a (hyper)graph partitioning algorithm in the final step	Low HGPA $O(knm)$, MCLA $O(k^2nm^2)$ and HBGF $O(knm)$ CSPA $O(kn^2m)$
Mirkin distance	• uses Mirkin distance as dissimilarity measure between partitions • easy to understand and program	• has high computational complexity • not suitable for large datasets	Heuristic dependent
Finite mixture models	• has a low computational complexity	• data is modeled as random variables • it is assumed that data are independent and identically distributed • the number of clusters in the consensus partition has to be specified	Low $O(knm)$
Genetic algorithms	• uncovers partitions that other methods may skip	• algorithms cannot test whether a solution is optimal or not • heuristic nature of algorithms may produce different results in successive runs	Heuristic dependent
Locally adaptive clustering	• heuristics such as WBPA has a low computational complexity and is more efficient than others	• can be applied on datasets of numerical data • the number of clusters in the consensus partition has to be specified • the methods need a (hyper)graph partitioning algorithm in the final step	Low $O(n^2)$ $O(knm)$
Kernel based	• uses a kernel function as similarity measure between partitions	• Partition Relevance Analysis increases the quality of the consensus at the cost of computational complexity	Heuristic dependent

2.5.2.6 Use of Unsupervised Ensemble Methods in Microarray Data

Although unsupervised ensemble clustering techniques have improved the accuracy and the reliability of clustering results for microarray data, some issues exist considering the fact that classes of functionally correlated genes are not very distinct and same gene may belong to several functional classes. Gasch and Eisen [124] and Avogadri and Valentini [125] have proposed solutions to tackle this fuzzy nature of clusters in gene expression data. Nonetheless, unsupervised clustering methods have

been applied in the discovery of new subclasses of diseases [126,127] and for detection of subsets of co-expressed genes [128]. Methods for the improvement of accuracy and reliability of the clustering results are given in [84,129,130,131]. Yu Z *et al*. [132] have proposed a graph-based consensus clustering algorithm to determine the number of classes in microarray data.

2.5.3 Semi-Supervised Learning

Semi-supervised learning is a new direction in Machine Learning research and uses both labelled and unlabeled data for training and lies between supervised learning and unsupervised learning [133]. In other words, semi-supervised learning can be viewed as:

- supervised learning aided by additional unlabeled data;
- unsupervised learning aided by additional labelled data.

Semi-supervised learning tries to uncover relationship in a dataset by using a small amount of labelled data along with a large amount of unlabeled data. This leads to a considerable improvement in learning efficiency.

The goal of semi-supervised learning is to train a classifier *f* from both the labelled and unlabeled data, such that it is better than the supervised classifier trained on the labelled data alone. *Figure* 2.10 shows the general process of semi-supervised learning.

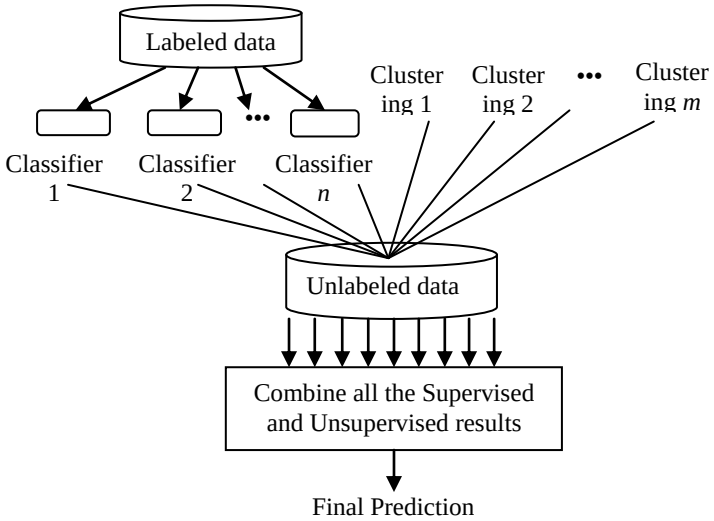


Figure 2.10: The General Process of Semi-Supervised Learning

To obtain labelled instances is often difficult, expensive, or time consuming as the efforts of experienced human annotators are required for manual classification whereas unlabeled data may be more readily available. Since semi-supervised learning uses a large amount of unlabeled data along with a small amount of labelled data to build classifiers, it requires less human effort while giving improved results. Hence semi-supervised learning is of great interest both in theory and in practice.

2.5.3.1 Methods Adopted For Semi-Supervised Learning

Being an active research area, semi-supervised learning is going through an experimental phase and there is no concrete taxonomy on the methods adopted for its approaches. Nevertheless, the semi-supervised methodologies can be broadly divided into five major approaches [134], which are shown in *Figure* 2.11.

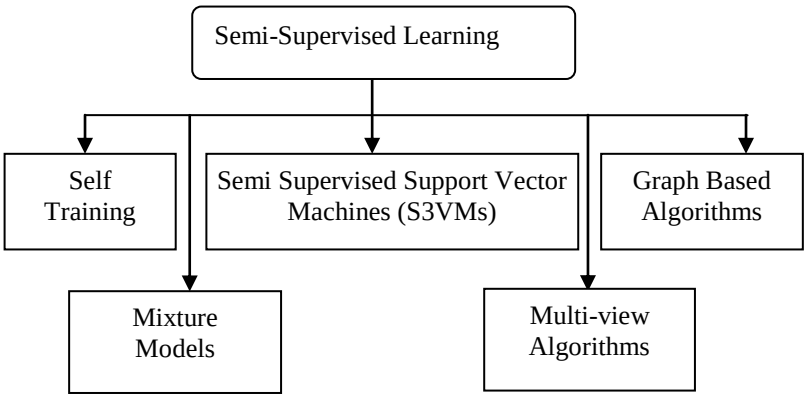


Figure 2.11: Semi-Supervised Learning Methodologies

Self-Training

Self-training first trains a classifier with a small amount of labelled data and then the classifier is used to classify unlabeled data. The most confident unlabeled points, along with the predicted labels, are then added to the training set. The classifier is re-trained on the now larger set of labelled data and the procedure is repeated. Self-training is a wrapper method, i.e., the choice of a classifier ranges from a simple algorithm to a complicated classifier. The assumption of self-training is that its own predictions (high confidence ones), tend to be correct though a classification mistake can reinforce itself by generating incorrectly labelled data. Various heuristics have

been proposed to avoid this problem such as 'unlearning' unlabeled points if the prediction confidence drops below a threshold.

Self-training has been used in natural language processing tasks such as word sense disambiguation [135], identification of subjective nouns [136], classification of dialogues [137] and object detection systems from images [138].

Mixture Models

The idea behind mixture models is that unlabeled data contains mixed instances from all classes. If the probability distributions of instances from each class are known, the mixture may be decomposed into individual classes. If the generative mixture model is correct, the unlabeled data may improve accuracy [139,140,141] and the semi-supervised learning is likely to be effective. But the unlabeled data may degrade performance if the choice of a model is wrong. It is thus important to carefully construct the mixture model to reflect reality.

One way to avoid the danger of using the wrong model is to use a model based on domain knowledge. Nigam *et al.* [142] applied the EM algorithm on a mixture of multinomials for text classification. Since EM is prone to local maxima, Nigam *et al.* [143] made use of active learning to select a starting point. Fujino *et al.* [144] made use of a 'bias correction' term and discriminative training using the maximum entropy principle in mixture models. Another method, 'Cluster-then-label', clusters the entire dataset using different clustering algorithms and then labels the clusters [145,146].

Co-Training and Multiview Learning

Co-training [147,148] assumes that (i) features can be split into two sets, i.e. two views; (ii) each sub-feature set is good enough to train a good classifier; (iii) the two sets are independent of each other for a given class [149]. Initially two separate classifiers are trained with the labelled data on the two sub-feature sets respectively. Each classifier then classifies the unlabeled data, and 'teaches' the other classifier with the few unlabeled examples and the predicted labels. Hence an iterative classification of unlabelled data is performed by the classifiers, with the help of the predicted labels.

Nigam and Ghani [150] compare co-training and EM and their experimental results show that co-training performs well if there is a natural split of the features. They also introduced the co-EM algorithm that uses EM to iteratively label unlabelled data. Jones [151] used co-training, co-EM and other related methods for information extraction from text. Balcan and Blum [152] show that co-training can be quite effective even when using one labelled point to learn the classifier, which was also confirmed by Zhou and Xu [153].

In contrast to splitting of features in co-training, Goldman and Zhou [154] use two different classifiers that use the entire feature set to teach one another. Zhou and Goldman [155] also proposed a single-view multiple-learner Democratic Co-learning algorithm. Tri-training using three learners was proposed by Zhou and Li [156], where majority voting on an unlabeled data point is used to train the third classifier. Johnson and Zhang [157] proposed a two-view model that uses a weaker conditional independence assumption of the feature set.

Li et al. [158] proposed a co-training algorithm for heterogeneous microarray datasets having both labelled and unlabeled samples. Qi *et al.* [159] introduced a Bayesian generalization approach that trains a kernel classifier using both labelled and unlabeled gene expression data.

Multiview learning is a generalization of co-training and *k* views; hence the algorithm has access to *k* learners. The learners might be of different types (i.e., decision trees, SVM, neural networks, etc.) but they take the same features of instance *x* as input. The goal of multiple learners is to produce multiple hypotheses to minimise risk and to make similar predictions, i.e. agree with each other on any given unlabeled instance. Multiview learning has been applied to semi-supervised regression [160,161] and to structured output spaces [162,163].

S3VMs

S3VM [133] is an extension of support vector machines that uses unlabeled data, whereas SVM uses only labelled data. The S3VM objective prefers the decision boundary to be in a low density gap in the dataset, such that only a few unlabeled

instances are close to the decision boundary. The goal is to find a labelling of the unlabeled data. The decision boundary has the smallest generalization error bound on unlabeled data [164] and the unlabeled data guides the linear boundary away from dense regions. Shi and Zhang [165] used a low density separation (LDS) approach for outcome prediction for different types of cancer while Wang *et al.* [166] extract information from unlabeled gene expression data for estimating the Bayes decision boundary by developing a large margin semi-supervised learning method.

Graph-Based Methods

The nodes in graph-based semi-supervised methods are the labelled and unlabeled instances in the dataset and an edge represents the similarity between two instances. Nodes that are connected by a large-weight edge tend to have the same label, and they are assumed to be similar to their neighbours in the graph, which in turn, are similar to their neighbours' neighbours. These methods usually assume label predictions to be smooth over the graph.

Graph-based methods can be viewed as estimating a function f on the graph that satisfies two things at the same time: (i) the prediction function $f(x)$ should be close to the given labels on the labelled nodes, and (ii) the label function f should be smooth on the whole graph. This can be expressed in a regularization framework where the first term is a loss function, and the second term is a regularizer. Graph-based semi-supervised learning algorithms differ from each other in the choice of the loss function and the regularizer. To construct a good graph is more important than to choose among the methods. A semi-supervised logistic model construction of a nonlinear discriminant procedure, based on both labelled and unlabeled datasets, using graph-based regularization is given in Kawano *et al.* [167].

Mincut, a graph-based semi-supervised learning algorithm, was proposed by Blum and Chawla [168], by finding a partition which minimizes a cost function defined on the graph. For a two class problem, the classes are labelled as source and sink. Mincut tries to find a minimum set of edges, the removal of which will block all flow from the sources to the sinks. The nodes connecting to the sources are labelled positive, and those to the sinks are labelled negative. Mincut can be applied to multiple perturbed

graphs and the labels are determined by a majority vote. Blum *et al.* [169] perturbed the graph by adding random noise to the edge weights. Pang and Lee [170] used mincut to improve the classification of a sentence.

A *harmonic function* is a graph-based semi-supervised learning algorithm. A harmonic function is a function that has the same values as the given labels on the labelled data and satisfies the weighted average property on the unlabeled data. In other words, the value assigned to each unlabeled vertex is the weighted average of its neighbours' values. Grady and Funka-Lea [171] applied the harmonic function method to medical image segmentation tasks.

Manifold Regularization: Mincut and the harmonic function are learning algorithms that learn a function *f* that is restricted to the labelled and unlabeled vertices in the graph. One cannot predict the label on an unseen test instance, unless the instance is included as a new vertex into the graph and the computation is repeated. The label function *f* assigns labels to instances which sometimes may not be correct due to label noise. Manifold regularization addresses these two issues by defining *f* in the entire feature space [172,173].

2.5.3.2 Comparison among Semi-Supervised Learning Methods

A comparison based on the merits and demerits of the different semi-supervised learning methods discussed in *Section* 2.5.3.1 is summarized in *Table* 2.4.

2.5.3.3 Use of Unsupervised Ensemble Methods in Microarray Data

By integrating consensus clustering with semi-supervised clustering for analyzing gene expression data showed an improvement in the clustering quality by reducing the impact of noise and high dimensionality in microarray data, as opposed to using consensus clustering or semi-supervised clustering separately [174]. Hence semi-supervised learning has been used with some amount of success to reveal functional associations among genes [158,174], outcome prediction for different types of cancer [165] and discovering human disease-causing genes [175,176]. It has also proved to be effective in protein classification [177], peptide identification in shotgun proteomics [178] prediction

of transcription factor-gene interaction [179] and gene expression-based cancer subtype discovery [180,181,182].

Table 2.4: Merits and demerits of the different methods of Semi-Supervised Learning

Method	Merits	Demerits
Self Training	• One of the simplest methods, easy to use • A wrapper method, applies to existing (complex) classifiers • Often used in real tasks like natural language processing	• Early mistakes in heuristic solutions could reinforce themselves, e.g., "un-label" an instance if its confidence falls below a threshold • Convergence to a solution is a problem but in special cases self-training is equivalent to the Expectation-Maximization (EM) algorithm
Generative Models	• Clear, well-studied probabilistic framework • Can be extremely effective for correctly selected models	• Model identifiability is a problem, difficult to verify the correctness of the model • Unlabeled data may have a high likelihood of wrong interpretation if generative model selected is wrong
S3VMs	• Applicable wherever SVMs can be used • Has a clear mathematical framework	• Optimization is difficult since it can be trapped in bad local optima • The assumption that 'unlabeled data from different classes are separated by a large margin' may lead to a potentially lower gain than generative model or graph-based methods
Graph Based Algorithms	• Has a clear mathematical framework • Performance is strong if the graph happens to fit the task • Can be extended to directed graphs also	• Performance is bad if the graph structure is incorrect • Sensitive to graph structure and edge weights
Multi-view Algorithms	• Simple wrapper method that can be applied to almost all existing classifiers • It is less sensitive to mistakes than self-training	• Natural feature splits (i.e., multiple views) may not exist • Models using both features, e.g., image and text, should give better performance

2.5.4 Discussion

Ensemble techniques have demonstrated their strength in supervised, unsupervised or semi-supervised scenarios where base models are combined by learning from labeled data or by consensus. This is particularly useful since the results of different classification and clustering algorithms may not be consistent. Hence, an effective combination of algorithms is required to improve clustering quality. There are two main issues that are involved in design of the ensembles.

a. **The diversity of the algorithms**: The diversity and the quality of the base classifiers and clustering algorithms highly influence the quality of results produced by the ensemble and should not be influenced by the dimensionality of the data or the biases of the participating classifiers or clustering methods.

b. The integration of the outputs of the base algorithms: A better understanding of the relationship between the performance of various combination and consensus functions to obtain a consensus that will work for any type of data (numerical, categorical and mixed) vis-a-vis the basic properties (diversity and quality) of ensembles is necessary.

The idea of ensemble learning is to employ multiple learners and combine their predictions for better accuracy, especially for an application like cancer data classification. An ensemble is largely characterized by the diversity generation mechanism and the choice of its consensus or combination procedure. Diversity is the degree to which classifiers disagree in the errors they make. This allows the voted accuracy to be greater than the accuracy of any single classifier. Diversity can be achieved by employing different classifier models, different feature subsets and different training data sets [183]. The result of a clustering ensemble process is a consensus of all the individual base classifiers and base clustering algorithms that takes care of the possible errors made during clustering by a single algorithm, and also gives more weight to the decision arrived at by a majority of the algorithms. More importantly, ensembles exempt the user from deciding on a particular classifier or clustering algorithm, thereby avoiding the risk of making a wrong selection. Ensembles are also characterized by their ability to deal with small sample size and high dimensionality; hence they have been widely applied to microarray data analysis.

An in-depth study of the above issues will provide useful guidance for applying ensemble techniques in practice.

2.6 K-Fold Cross Validation

K-fold cross validation is used in the field of machine learning to determine how accurately a learning algorithm will be able to predict data that it was not trained on. Suppose we have a model with one or more unknown parameters, and a data set to which the model can be fit (the training data set). The fitting process optimizes the model parameters to make the model fit the training data as well as possible. If an independent sample of validation data is taken from the same population as the

training data, it will be observed that the model does not fit the validation data as well as it fits the training data. This is called over-fitting, and generally happens when the size of the training data set is small, or when the number of parameters in the model is large.

To avoid over-fitting, cross validation is used to estimate the expected level of fit of a model to a data set that is independent of the data that were used to train the model. The basic idea behind cross validation is not to use the entire data set when training a learner. Some of the data is removed before training begins and when the training is done, the data that was removed can be used to test the performance of the learned model on "new" data.

In *k*-fold cross-validation, the data set is divided into k equal size subsets. Of the *k* subsets, one subset is retained as the validation data to be used as the test set and the other *k*-1 subsets are put together to form a training set, as shown in *Figure* 2.12. The cross-validation process is then repeated *k* times (the folds), with each of the *k* subsets used exactly once as the validation data. The *k* results from the folds can then be averaged (or combined) to produce a single estimation. The advantage of this method is that all observations are used for both training and validation and each data point gets to be in a test set exactly once and in a training set *k*-1 times. The disadvantage of this method is that the training algorithm has to be rerun from scratch *k* times, which means it takes *k* times as much computation to make an evaluation.

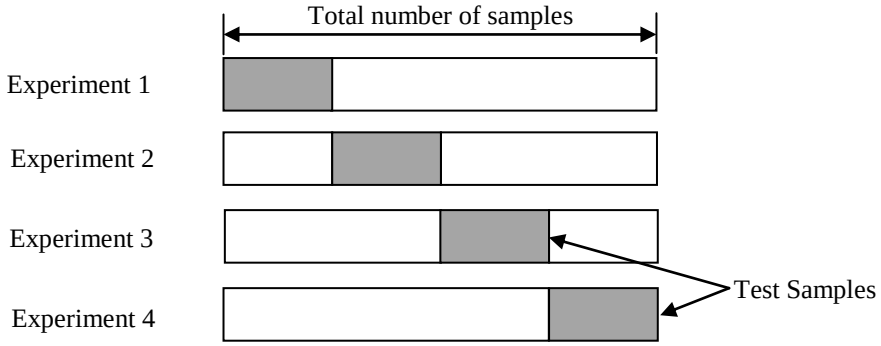


Figure 2.12: K-fold cross validation

Unfortunately, there is no theoretically 'perfect' way of determining the appropriate *k* value. Using the value $k = 10$ seems to be a good rule of thumb [184], although the true best value differs for each algorithm and each dataset.

2.7 Discussion

Ensemble techniques have demonstrated their strength in supervised, unsupervised or semi-supervised scenarios where base models are combined by learning from labelled data or by consensus. Use of ensembles has led to improvements in accuracy. In addition, information explosion has motivated the need for learning from multiple sources with the sole objective of understanding data better. This has led to rapid developments in the field of ensemble research, with an aim to combine the complementary predictive powers of multiple models. The availability of various learning packages has further fuelled growth and interest in this research area.

2.8 Validity Measures

Grouping of gene expression data results in groups of co-expressed genes, groups of samples with a common phenotype, or "blocks" of genes and samples involved in specific biological processes. However, different clustering and classification algorithms, or even an ensemble using different base clustering / classification algorithms, generally result in different sets of clusters [185]. The validation techniques have the potential to provide an analytical assessment of the amount and type of structure captured by a partitioning and should be a key tool in the interpretation of clustering results. Therefore, it is important to compare various clustering and classification results and select the one that best fits the "true" data distribution.

Validation is the process of assessing the quality and reliability of the cluster sets derived from various clustering processes. Generally, cluster validity has three aspects. First, the quality of clusters can be measured in terms of *homogeneity* and *separation* on the basis of the definition of a cluster. Objects within one cluster are similar to each other and different from objects in other clusters. The second aspect

comes from "ground truth" of the clusters. The "ground truth" could come from domain knowledge, such as the clinical diagnosis of normal or cancerous tissues. Cluster validation is based on the agreement between clustering results and the "ground truth". The third aspect focuses on the reliability of the clusters or the likelihood that the cluster structure is not formed by chance.

Validation techniques can be divided into two main categories: external and internal validation measures [186].

2.8.1 External Validity Measures

External validity measures generally use supervised information. These indices mainly quantify how good is the obtained partitioning with respect to prior class labelled information available or a given "gold" standard which is another partition of the objects. Evidently, this is useful to permit an entirely objective evaluation and comparison of clustering algorithms and clusters on benchmark data. Adjusted rand index, Rand index, F-measure, Purity, NMI are some common examples of external validity indices.

2.8.1.1 Rand Index

Rand index [90] is a measure of the similarity between two clusters. The Rand index is defined as the number of pairs of objects that are either in the same group or in different groups in both partitions divided by the total number of pairs of objects.

$$Rand\ index = \frac{a + d}{a + b + c + d} \tag{2.1}$$

Where a is the number of object pairs (g_i, g_j), where $C_{ij} = 1$ and $P_{ij} = 1$, b is the number of object pairs (g_i, g_j), where $C_{ij} = 1$ and $P_{ij} = 0$, c is the number of object pairs (g_i, g_j), where $C_{ij} = 0$ and $P_{ij} = 1$, d is the number of object pairs (g_i, g_j), where $C_{ij} = 0$ and $P_{ij} = 0$.

The Rand index lies between 0 and 1. The maximum value i.e., 1 is achieved when both partitions, C and P, agree perfectly.

2.8.1.2 Adjusted Rand Index

The problem associated with Rand index is that it does not show a constant value for random partitions [187]. So Hubert and Arabie [188] overcome the deficiency of Rand index and assume randomness for partitions. So modified Rand index (RI) is defined as follows:

$$ARI = \frac{\binom{n}{2}(a+d) - [(a+b)(a+c)+(c+d)(b+d)]}{\binom{n}{2}^2 - [(a+b)(a+c)+(c+d)(b+d)]} \quad (2.2)$$

As in the case of Rand index, here also a is the number of object pairs (g_i, g_j), where $C_{ij} = 1$ and $P_{ij} = 1$, b is the number of object pairs (g_i, g_j), where $C_{ij} = 1$ and $P_{ij} = 0$, c is the number of object pairs (g_i, g_j), where $C_{ij} = 0$ and $P_{ij} = 1$, d is the number of object pairs (g_i, g_j), where $C_{ij} = 0$ and $P_{ij} = 0$.

ARI [188] gives value between [0,1], 1 for best partitioning result and 0 for worst partition, -1 value shows random partitioning result.

2.8.1.3 Jaccard Coefficient

The Jaccard Coefficient measures the proportion of pairs that are in the same cluster C and in the same partition P from those that are either in the same cluster or in the same partition [189]. The Jaccard Coefficient is defined as

$$J = \frac{a}{a+b+c} \quad (2.3)$$

where a=SS if the pair belongs to the same cluster C and to the same group P, b = SD if the pair belongs to the same cluster C and to different groups P and c = DS if the pair belongs to different clusters C and to the same group P.

As in the Rand Index, the values of these coefficients lie between 0 and 1, and values close to 1 indicate high agreement between C and P.

2.8.1.4 Fowlkes-Mallows Index

The Fowlkes-Mallows Index [190] is the geometrical mean of two probabilities: the probability that two random objects are in the same cluster given they are in the same

group, and the probability that two random objects are in the same group given they in the same cluster [189,190]. The FM Index is defined as:

$$FM = \sqrt{\frac{a}{(a+b)}\frac{a}{(a+c)}} \tag{2.4}$$

Similar to Jaccard Coefficient, here a=SS if the pair belongs to the same cluster C and to the same group P, b = SD if the pair belongs to the same cluster C and to different groups P and c = DS if the pair belongs to different clusters C and to the same group P.

As in the Rand Index and Jaccard Coefficient, values close to 1 indicate high agreement between C and P.

2.8.1.5 F-Measure

The F-Measure [191] is the harmonic mean of two measures called *Precision* and *Recall*, as follows.

- **Precision**: This measure is defined as the fraction of a cluster that consists of objects of a specific class. The precision of a cluster V_j with respect to class U_i is represented by

$$Precision(U_i, V_j) = \frac{n_{i,j}}{n_j} \tag{2.5}$$

- **Recall**: This measure is defined as the proportion of objects of a class in a cluster. The recall of a cluster V_j with respect to class U_i is represented by

$$Recall(U_i, V_j) = \frac{n_{i,j}}{n_i} \tag{2.6}$$

  where $n_{i,j}$ equals number of data items belonging to class U_i and cluster V_j, n_j is the number of data items belonging to cluster V_j and n_i denotes number of data items belonging to class U_i.

The F-Measure is defined as the rate at which a cluster contains only objects of a particular class and all objects of that class. Thus, the F-Measure of a cluster U_i with respect to class V_j is represented by the following expression:

$$F - Measure = \frac{2 \text{ x } Precision(U_i, V_j) \text{ x } Recall(U_i, V_j)}{Precision(U_i, V_j) + Recall(U_i, V_j)} \quad (2.7)$$

F-Measure is useful in the sense that it provides an objective information on the degree to which a clustering algorithm is able to recover the original clusters. It is measured in the range [0, 1] and high values indicate a good quality of clustering.

2.8.1.6 Z-score

Z-score [192] is calculated by investigating the relation between a clustering obtained by an algorithm and the functional annotation of the genes in the cluster in terms of *mutual information* (MI). The z-score represents a standardized distance between the MI value obtained by clustering and those MI values obtained by random assignment of genes to clusters. Higher z-scores indicate that the clustering results are more significantly related to the gene function, indicating a more biologically relevant clustering result.

2.8.2 Internal Validity Measures

In cases where no "gold" standard is available, an evaluation based on internal validation measures becomes appropriate. Internal validation techniques do not use additional knowledge in the form of class labels, but base their quality estimate on the intrinsic information of the data. Specifically, they attempt to measure how well a given partitioning corresponds to the natural cluster structure of the data [193]. Some of the popular validity measures are discussed next.

2.8.2.1 Cluster Homogeneity

Homogeneity [194] measures the quality of clusters on the basis of the definition of a cluster, i.e. objects within a cluster are similar while objects in different clusters are dissimilar. It is calculated as follows.

- Compute the average value of similarity between each gene g_i and the centroid of the cluster C_i to which it has been assigned.

$$H(C_i) = \frac{1}{|C_i|} \sum_{g_i \epsilon C_i} \text{Similarity}(g_i, g_i') \tag{2.8}$$

where g_i' is the centroid of C_i.

- Calculate the average homogeneity for the clustering *C* weighted according to the size of the clusters as

$$H_{\text{avg}} = \frac{1}{|G|} \sum_{C_i \epsilon \text{C}} |C_i| \, H(C_i) \tag{2.9}$$

2.8.2.2 Connectivity

Let $nn_{i(j)}$ be defined as the j^{th} nearest neighbor of observation *i*, and let $x_{i,nn_{i(j)}}$ be zero if *i* and $nn_{i(j)}$ are in the same cluster and 1/*j* otherwise. Then, for a particular clustering partition C = {C_1,...,C_K} of *N* observations into *K* disjoint clusters, the connectivity is defined as [193]

$$Conn(C) = \sum_{i=1}^{N} \sum_{j=1}^{L} x_{i,nn_{i(j)}} \tag{2.10}$$

where *L* is a parameter that determines the number of neighbors that contribute to the connectivity measure. The connectivity has a value between zero and ∞ and should be minimized.

2.8.2.3 Silhouette Index

Silhouette index [195] is used to assess the quality of any clustering solution and reflects the compactness and separation of clusters. It is the average of each observation's silhouette value. The silhouette value measures the degree of confidence in the clustering assignment of a particular observation, with well-clustered observations having values near 1 and poorly clustered observations having values near -1. For observation *i*, it is defined as [195]

$$S_i = \frac{b_i - a_i}{max\ (b_i, a_i)} \tag{2.11}$$

where a_i is the average distance between i and all other observations in the same cluster and b_i is the average distance between i and the observations in the "nearest neighboring cluster".

2.8.2.4 Davies-Bouldin Index

Let s_i be a measure of dispersion of cluster C_i and $d(C_i, C_j) \equiv d_{ij}$ the dissimilarity between two clusters. The dispersion of a cluster C_i is defined as

$$s_i = \sqrt{\frac{1}{n_i} \sum_{x \in C_i} \left\| x - x_i' \right\|^2} \tag{2.12}$$

The Davies-Bouldin index [196] is defined as

$$DB_m = \frac{1}{m} \sum_{i=1}^{m} R_i \tag{2.13}$$

where $R_i = max_{j=1,\dots,m j \neq i} R_{ij}, i = 1, \dots, m$ and R_{ij} is a similarity index between C_i and C_j satisfying the condition $R_{ij} = \frac{s_i + s_j}{d_{ij}}$

2.8.2.5 Dunn Index

The Dunn Index [197] is defined as:

$$D_m = min_{i=1,\dots,m} \left\{ min_{j=i+1,\dots,m} \left(\frac{d(C_i, C_j)}{max_{k=1,\dots,m} diam(C_k)} \right) \right\} \tag{2.14}$$

where $d(C_i, C_j) = min_{x \in C_i, y \in C_j} d(x, y)$ is the dissimilarity between two clusters C_i and C_j.

Diameter of the cluster C is $diam(C) = max_{x,y \in C} d(x, y)$.

2.8.2.6 *p*-Value

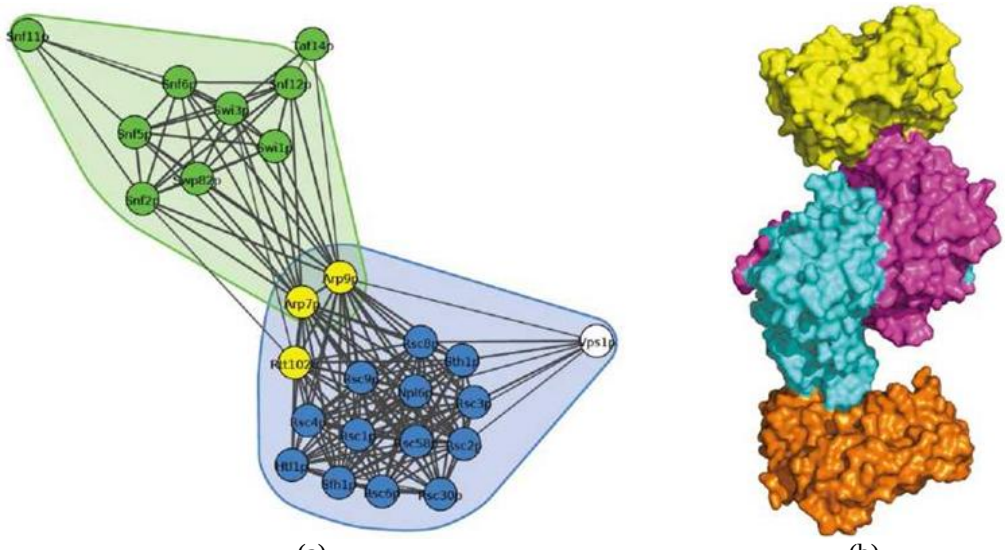
The reliability of the resulting clusters can be estimated by the *p*-value [198] of a cluster. It measures the probability of finding the number of genes involved in a given Gene Ontology (GO) term (i.e., function, process and component) within a cluster. From a given GO category, the probability *p* of getting *k* or more genes within a cluster of size *n*, is defined as:

$$p = 1 - \sum_{i=0}^{k-1} \frac{\binom{f}{i}\binom{g-f}{n-i}}{\binom{g}{n}} \tag{2.15}$$

A cumulative hyper-geometric distribution is used to compute the p-value. A low p-value indicates that the genes belonging to the enriched functional categories are biologically significant in the corresponding clusters.

2.9 Protein-Protein Interaction (PPI) Data

A Protein-Protein Interaction (PPI) network can be described as a complex system of proteins linked by interactions as shown in *Figure* 2.13(a). The computational analysis of PPI networks begins with the representation of the PPI network structure in the form of a mathematical graph consisting of nodes and edges [199].



(a) (b)

Figure 2.13: (a) A PPI network (b) a protein complex

Proteins are represented as nodes in such a graph; two proteins that interact physically are represented as adjacent nodes connected by an edge. Based on this graphic representation, various computational approaches, such as data mining, machine learning, and statistical approaches, can be designed to reveal the organization of PPI networks at different levels.

In PPI networks, clusters correspond to two types of modules: protein complexes and functional modules. Protein complexes are groups of proteins that interact with each other at the same time and place, forming a single multi-molecular machine. *Figure* 2.13(b) shows an example of a protein complex [200]. Functional modules consist of proteins that participate in a particular cellular process while binding to each other at a different time and place. The methods of data mining can be applied to identify various aspects of network organization [201]. For example: (i) Proteins located at neighbouring positions in a graph are generally considered to share functions ("guilt by association"). On this basis, the functions of a protein may be predicted by examining the proteins with which it interacts and the protein complexes to which it belongs. (ii) Densely connected subgraphs in the network are likely to form protein complexes that function as single units in a particular biological process. (iii) Investigation of network topological features can shed light on the biological system.

2.10 Analysis of PPI Data Using Data Mining Techniques

In the "post-genome" era, proteomics [202,203] has become an essential field and drawn much attention. Proteomics is the systematic study of the many and diverse properties of proteins with the aim of providing detailed descriptions of the structure, function, and control of biological systems in health and diseases.

A particular focus of the field of proteomics is the nature and role of interactions between proteins. Protein-protein interactions [202,204,205,206,207,208] play different roles in biology depending on the composition, affinity and lifetime of the association. It has been observed that proteins seldom act as single isolated species while performing

their functions in a living organism. The study of protein interactions is fundamental to understand how proteins function within a cell.

Protein-protein interaction plays a key role in the cellular processes of an organism. An accurate and efficient identification of protein-protein interaction is fundamental for us to understand the physiology, cellular functions and complexity of an organism. The knowledge of protein-protein interaction can provide important information on the possible biological function of a protein. Much effort has been done to detect and analyze protein-protein interactions using experimental methods such as the yeast two-hybrid system which is well known. Recently, several algorithms have been developed to identify functional interactions between proteins using computational methods which can provide clues for the experimental methods and could simplify the task of protein interaction mapping. As the prediction task becomes harder the need for methods that can accommodate high levels of missing values and are directly interpretable by biologists increases.

2.10.1 Properties of PPI networks

The simplest representation of PPI networks takes the form of a mathematical graph consisting of nodes and edges (or links). Proteins are represented as nodes and an edge represents a pair of proteins which physically interact. The degree of a node is the number of other nodes with which it is connected. It is the most elementary characteristic of a node. It has been determined that most proteins participate in only a few interactions, while a few participate in dozens of overlapping interactions. Hu and Pan [209] observed that a protein-protein interaction network has three main properties called scale invariance, dis-assortivity and small-world effect. There has been much work that has been done to study these properties and to uncover new ones.

2.10.2 PPI Network and Protein Complexes

A Protein complex (or multi-protein complex) is a group of two or more interacting proteins. No protein is an island by itself or, rather, very few proteins are. Most proteins seem to function with complicated cellular pathways, interacting with other proteins either in pairs or as components of large complexes. So identification of protein complexes is crucial for understanding the principles of cellular organization

and functions. As the size of protein-protein interaction sets increase the general trend is to represent the interaction as a network and to develop effective algorithms to detect significant complexes in such networks. There have been various methods that have been proposed to detect protein complexes.

Partitional clustering approaches can partition a network into multi separated sub-networks. As a typical example, the Restricted Neighborhood Search Clustering (RNSC) algorithm [210] arrived at the best partition of a network by using a cost function. The method starts with randomly partitioning a network and iteratively moves a vertex from one cluster to another with an aim to decrease the total cost of the clusters. When some moves have been reached without decreasing the cost function, the algorithms stops. This method can obtain the best partition by running multiple number of times. However, it needs the number of clusters *apriori* and the results depend heavily on the quality of initial clustering. Moreover, it cannot detect overlapping protein complexes since it assigns each vertex to a specific cluster.

Hierarchical clustering approaches build (agglomerative), or break up (divisive), a hierarchy of clusters. The traditional representation of this hierarchy is a tree called a dendrogram. Agglomerative algorithms are bottom-up algorithms that iteratively merge vertices, whereas divisive algorithms top-down algorithms that recursively divide a graph into two or more sub-graphs. For iteratively merging vertices, the similarity or distance between two vertices needs to be measured. The Super Paramagnetic Clustering (SPC) algorithm [211] is an example of iterative merging. For recursively dividing a graph, the vertices or edges to be removed are to be selected properly. Girvan and Newman [212,213] decomposed a network based on the graph theoretical concept of betweenness centrality. Hierarchical clustering approaches use a dendrogram to display the hierarchical organization of biological networks. As with most methods of predicting protein complexes from PPI data, hierarchical clustering approaches are also prone to false positives due to the noise present in PPI data [214].

Density-based clustering approaches detect densely connected sub-graphs from a network. However, all methods of protein interaction predictions are known to yield a non-negligible rate of false positives and to miss a fraction of existing interactions. Thus, only mining fully connected sub-graphs is too restrictive to be used in real

biological networks. In general, sub-graphs are identified by using a density threshold. A variety of alternative density functions have been proposed to detect dense sub-graphs [215,216,217,218]. The Clique Percolation Method (CPM) [219] detects overlapping protein complexes as k-clique percolation clusters. A k-clique is a complete sub-graph of size k. The Cluster Periphery-tracking algorithm (DPClus) [216] detects protein complexes by first selecting the node with the highest weight (seed node) as the initial cluster and then iteratively augments this cluster by including vertices one by one, which are closely related with the current cluster. Clique Finder (CFinder) [220] takes a parameter *k* as input and detects all the *k*-cliques of the input network. A *k*-clique percolation cluster is then constructed by linking all the adjacent *k*-cliques as a bigger subgraph. It can detect overlapping clusters. Dense-neighbourhood Extraction using Connectivity And conFidence Features (DECAFF) [221] incorporates functional information to detect dense and reliable subgraphs as protein complexes. Two subgraphs having a large overlap are merged using a hub-removal algorithm. False protein complexes with low reliability are filtered out using a probabilistic model. COre-AttaCHment (COACH) [222] defines core vertices from the neighborhood graphs as the hearts of protein complexes. Then it includes attachments into these cores to form biologically meaningful structures. The COACH method is able to detect the overlapping cores.

There are some other methods for protein complex detection. Jung *et al.* [223] proposed a protein complex prediction method based on simultaneous protein interaction networks. This concept is introduced to specify mutually exclusive interactions (MEI) as indicated from the overlapping interfaces and to exclude competition from MEIs that arise during the detection of protein complexes. Ozawa *et al.* [224] introduced a combinatorial approach for prediction of protein complexes focusing not only on determining member proteins in complexes but also on the PPI organization of the complexes. Cannataro *et al.* [225] proposed a new complex meta-predictor which is capable of predicting protein complexes by integrating the results of different predictors. It is based on a distributed architecture that wraps predictor as web/grid services that is built on top of the grid infrastructure.

Every new method proposed for protein complex detection comes up with its own comparative analyses with some earlier methods. It has been noticed that due to the

differences in PPI and benchmark datasets, evaluation criteria, threshold settings and parameters used, the results of the comparative analyses and surveys [226,227,228,229] on complex detection vary. Also, the "gold standard" complex data have become more enriched, new methods have emerged [230,217,200] and new evaluation measures have been proposed.

2.11 Discussion

There are two groups of people who are involved in the clustering of biological data. One is the biologist who uses an existing clustering algorithm to solve an underlying biological problem. The challenge before the biologist is to make an appropriate choice of an algorithm since different algorithms will produce different results. The other is the developer of clustering algorithms, who consistently strives to improve existing algorithms, so that the underlying biological problems can be solved efficiently. A proper amalgamation of these two groups will lead to rapid advancement in this field.

A clustering algorithm's suitability to cluster biological data depends upon certain desirable features such as speed, minimum number of input parameters, robustness to noise and outliers, redundancy handling and independence of object order input. Though the features of many clustering algorithms match these requirements, they have not yet been applied to clustering biological data. Moreover, not all validity measures are suitable for all gene datasets; hence a judicious choice of the applicability of the validity measure has to be made.

It is well known that most clustering methods are highly sensitive to input data and a slight variation or change in the data may result in very different gene clusters. If the information from genomic knowledge bases, such as GO, could be incorporated (data fusion) earlier in the analysis of genomic data, that additional information about genes and their relationship with each other will improve stability, accuracy and/or biological relevance of the cluster results.

Due to its ability to improve the results obtained from base classifiers and simple clustering algorithms, ensemble approaches have been very effective in combining independent, diversified models for the purpose of improving accuracy in prediction. The combination process integrates information from all partitions in the ensemble, so that possible errors of the individual algorithms could be compensated. That way the consensus obtained from a set of partitions of the same dataset may represent a better solution.

In the subsequent chapters the ensemble approaches for clustering microarray data will be applied with an aim to improve the stability, accuracy and biological relevance of the clustering results.