MINING FOR SIGNIFICANT INFORMATION FROM UNSTRUCTURED AND STRUCTURED BIOLOGICAL DATA AND ITS APPLICATIONS

A Dissertation
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Omar Ghazi Al-Azzam

In Partial Fulfillment of the Requirements
for the Degree of
DOCTOR OF PHILOSOPHY

Major Department:
Computer Science

March 2012

Fargo, North Dakota

North Dakota State University
Graduate School

Title

MINING FOR SIGNIFICANT INFORMATION FROM UNSTRUCTURED AND STRUCTURED BIOLOGICAL DATA AND ITS APPLICATIONS

By

OMAR Al-AZZAM

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

DOCTOR OF PHILOSOPHY

SUPERVISORY COMMITTEE:

Dr. Anne Denton
Chair

Dr. William Perrizo

Dr. Simone Ludwig

Dr. Shahryar Kianian

Approved:

7/02/2012
Date

Dr. Kendall Nygard
Department Chair

ABSTRACT

Massive amounts of biological data are being accumulated in science. Searching for significant meaningful information and patterns from different types of data is necessary towards gaining knowledge from these large amounts of data available to users. However, data mining techniques do not normally deal with significance. Integrating data mining techniques with standard statistical procedures provides a way for mining statistically significant, interesting information from both structured and unstructured data. In this dissertation, different algorithms for mining significant biological information from both unstructured and structured data are proposed. A weighted-density-based approach is presented for mining item data from unstructured textual representations. Different algorithms in the area of radiation hybrid mapping are developed for mining significant information from structured binary data. The proposed algorithms have different applications in the ordering problem in radiation hybrid mapping including: identifying unreliable markers, and building solid framework maps. Effectiveness of the proposed algorithms towards improving map stability is demonstrated. Map stability is determined based on resampling analysis. The proposed algorithms deal effectively and efficiently with multidimensional data and also reduce computational cost dramatically. Evaluation shows that the proposed algorithms outperform comparative methods in terms of both accuracy and computation cost.

ACKNOWLEDGMENTS

I am truly indebted and thankful to all people who have helped and contributed to create this work. This dissertation would not have been possible without your inspiration, exhortation, and support.

I am especially grateful to my advisor Dr. Anne Denton for hosting me in her data mining and bioinformatics research group as a research assistant. I am highly grateful for her supervision and endless support through my research work. Through continous discussions and meetings with her, I have gained sound knowledge on my research area. Appreciation is due to other committee members, Dr. Shahryar Kianian, Dr. William Perrizo, and Dr. Simone Ludwig.

I am very grateful to Dr. Ajay Kumar, who works as a postdoc on the RH mapping project, for his thoughtful contribution on my research. I thank Dr. Mohammad Javed Iqbal, RH mapping project manager, and Dr. Shahryar Kianian, RH mapping principle investigator, for all the support and thoughtful comments on this project. Thanks are due to all faculty members, postdocs, graduate students, and researcher who work on RH mapping project. Through general meetings and discussion I gained much understanding and many research ideas.

Personal thanks to all friends in Fargo, ND and back home for your support.

Most importantly, I would like to specially thank my family: my parents, my sisters, and my brothers for your endless support. Thank you all for the continuous encouragements, patients, and love through the years that enabled me to complete this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

LIST OF APPENDIX TABLES

CHAPTER 1. GENERAL INTRODUCTION

Gaining interesting information from large amounts of data is the major role of data mining techniques. Data mining techniques [60, 49, 15, 27] range from information extraction, supervised and unsupervised learning to pattern mining. Typically, none of these techniques address statistical significance. Conversely, standard statistical analysis [53, 54] cannot solve some of the complex problems in sciences. For some applications, determining significance can be as important as the result itself. Integrating data mining algorithms with standard statistical analysis procedures, provides means for mining significant information from both unstructured and structured data sources.

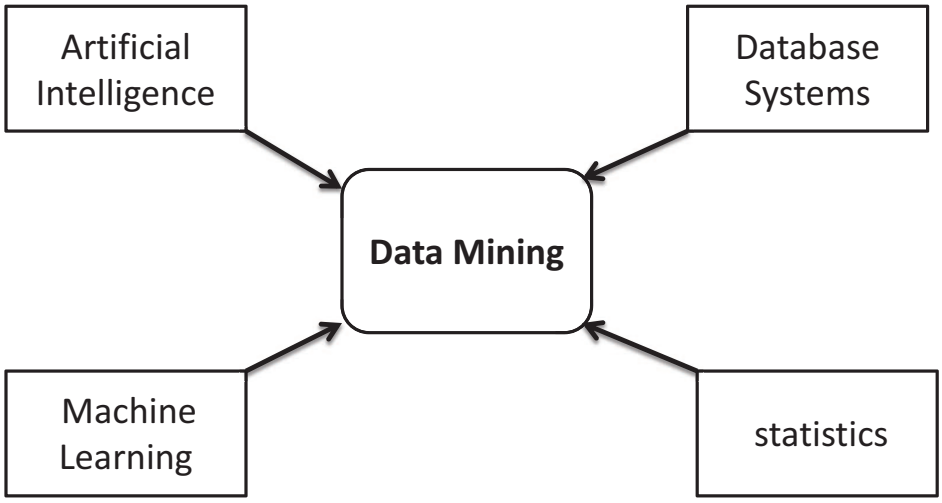

Figure 1: The relationship between data mining and other disciplines.

Figure 1 shows how data mining utilizes concepts and methods from database, artificial intelligence, machine learning, and statistics. Statistical methods are commonly used in

data mining. However, measuring how statistically significant the results are is uncommon. In this dissertation, we provide several algorithms towards further integrating data mining and statistical methods on both unstructured and structured data.

1.1. Mining for significance from the Unstructured Data Forms

Unstructured data that do not follow a specific model or is not represented in a relational fashion are the predominant data representation [69, 62, 43]. For that reason, new means for mining from the unstructured data are required. One application of mining significant information from unstructured data is the evaluation whether prediction from text is promising. Large parts of the biological science are represented through published documents. Valuable information can be extracted from such unstructured forms of data through bioinformatics analysis [69, 62, 43]. Classification techniques [64, 25] on textual forms can be used for prediction of gene ontology terms [9], and classifying gene expression [66].

Predicting functions directly from available protein domains is a common task in bioinformatics. However, since most biological science information is available through publications, it would be interesting to know whether or not this information can be used directly for prediction purposes. Standard classification techniques might produce misleading results because of the multi-relational nature between biological publications and item data. Figure 2 is a schematic that represents the problem of finding how useful the textual representations are in predicting class labels. Each publication can be related to many genes, which in turn can be related to many items, such as protein domains or ontology functions. Each item also can be related to many genes. The high dimensionality and muti-relational nature of this problem makes standard classification techniques and probabilistic relational models unsuitable to be used in this context.

As discussed above, standard classification techniques cannot be used in a multi-relational setting. Other techniques, such as probabilistic relation models [33] cannot be

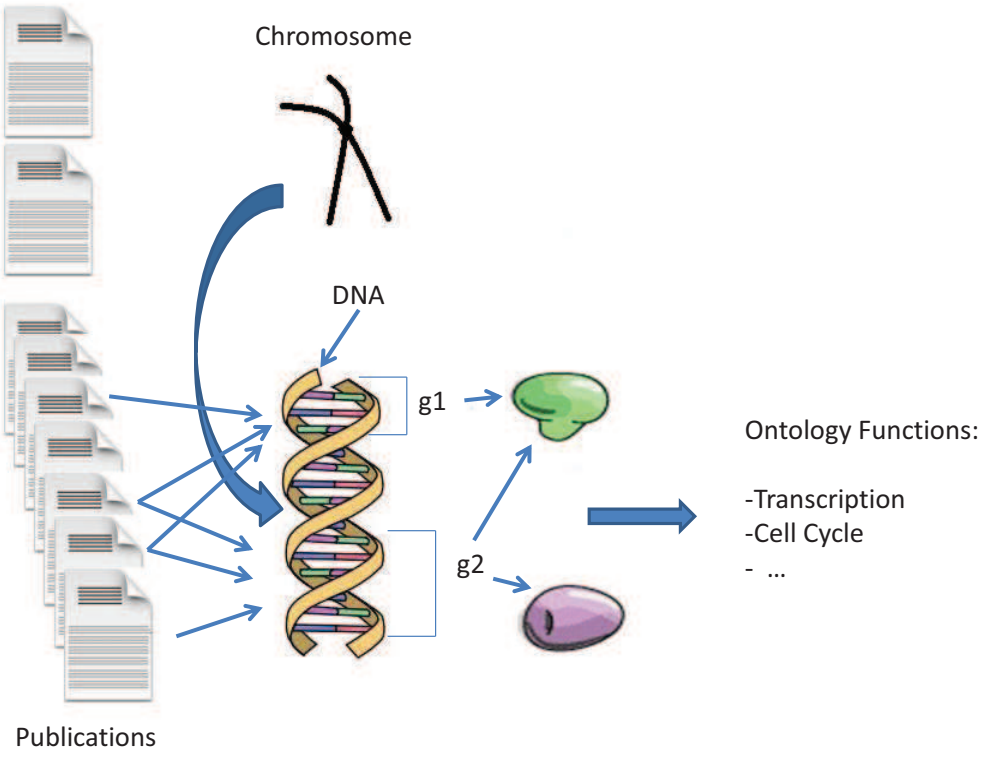

Figure 2: The problem of predicting item data (protein domains or ontology functions) from textual representations.

used for high dimensional data. On the other hand, integrating a proper re-weighting model with a density-based algorithm provides a solution to the multi-relational nature of the problem and finds whether prediction from text is promising. The proposed model is discussed in detail in Chapter 2.

1.2. Mining for significance from the Structured Data Forms

Significant information can also be mined from structured forms of data. These structured forms follow a data model, such as the relational model [19]. An evaluation of significance is important in the ordering problem in radiation hybrid (RH) mapping [24, 63, 32, 38, 34, 22, 26, 50] which is computationally equivalent to the traveling salesman problem (TSP) with the exception that the first and last markers in the map need not be

linked. Identifying unreliable markers in RH mapping (Chapters 3 and 4) is an application examples.

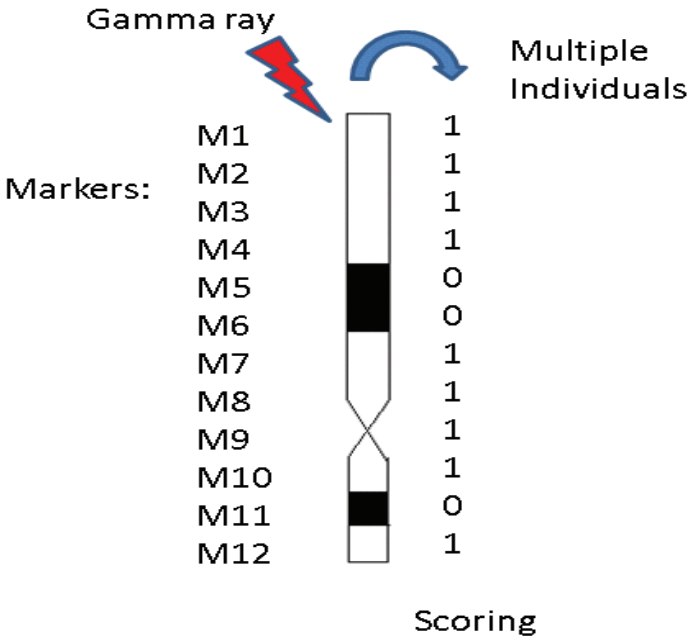

Figure 3: The process of scoring in radiation hybrid mapping.

Genome mapping [5, 32] is considered to be an important step in the sequencing of genomes since it provides valuable information towards the assembly of the sequenced data. In genome sequencing, a large number of DNA sequences are found. However, putting all these sequenced DNA pieces together in the correct order is a key task. Radiation hybrid (RH) mapping [24] is an experimental technique for ordering markers within the chromosomes of species. Figure 3 shows the experimental process where the chromosomes are irradiated with $X-$ or $\gamma-$ radiation to create deletions. These chromosomes are scored using a binary scoring system. If a marker is present in an individual it is scored 1, otherwise 0. This process is repeated on multiple individuals to create a mapping

population. Heuristic algorithms can be used to analyze these deletion patterns and find the best possible order. Table 1 is a toy example of a RH mapping data of a population with 10 individuals (v1 to v10) and 3 markers M1, M2, and M3. The goal of RH mapping is to order markers using the frequency of co-deletions/co-retention pattern between them. As can be seen from Table 1, markers $M1$ and $M2$ are retained together in the individual $v6$ and while markers $M2$ and $M3$ are co-deleted in individual $v5$. So, based on co-retention/co-deletion patterns, the best possible order, with minimum number of breakage, in this case is $\{M1, M2, M3\}$. However, ordering markers for RH mapping is computation intensive for many reasons. First: since the mapping problem is equivalent to the TSP problem, the ordering problem scales exponentially with number of markers. Second: there is a possibility of human scoring errors (mis-scoring problem). Some markers might be mistakenly mis-scored in the experimental process. Third: for many species, especially plants, sequence sections are repeated across the genome. The detection of markers might then not be sequence specific and might result in mistakes when scoring the data. Fourth: the missing data problem; it is very common that some of the experimental data are missing. The detection of markers for some markers cannot be clearly identified as 0 or 1, so these data are scored as missing data.

Table 1: A toy example of RH experimental data. The data consists of 3 markers tested on a population of 10 individuals.

$Mrk/Indv$	$v1$	$v2$	$v3$	$v4$	$v5$	$v6$	$v7$	$v8$	$v9$	$v10$
$M1$	1	1	1	0	**1**	1	0	1	1	1
$M2$	1	1	1	0	**0**	**1**	0	1	1	1
$M3$	1	1	1	0	0	**0**	0	1	1	1

Genome mapping is an application of a TSP problem. The standard TSP problem has to be adapted to the problem of genome mapping by modifying it such that the first and the last markers are not required to be linked. Heuristic algorithms can be used to find a map for every chromosome with the minimum cumulative physical distance. However,

even when using state-of-the-art heuristic algorithms for mapping, all the reasons discussed above may make a marker unreliable. If a marker cannot be placed reliably, such a marker may contribute to an overall poor map. These unreliable markers need to be detected and removed from the data to create a solid and reliable genome map. Chapters 3 and 4 describe two algorithms for detecting unreliable markers with the goal of both improving map quality, and building reliable solid framework maps.

Traditional methods for finding unreliable markers [39, 40, 41, 51] are computationally expensive. These methods rely on mapping data by resampling from the mapping population and creating histograms of the mapping results. Unreliable markers are removed iteratively using these histograms. In Chapters 3 and 4 we provide an alternative solution for discovering those unreliable markers without the need to map all the resampled data. The proposed network-based approach is computationally fast and outperforms clustering-based approaches in many aspects, including accuracy, map alignment with a baseline model, and physical map distance.

Building solid framework maps using the most reliable markers is another application of the techniques we develop. Under a scenario of mapping large numbers of markers with missing data and mis-scoring, the best strategy is to start with a solid framework map and iteratively insert other markers in the best possible position. Traditional techniques for filtering unreliable markers that rely on mapping data from resampling analysis as they are discussed above are not useful in this scenario due to their high computational complexity. The support network algorithm described in Chapter 4 is a successful solution in this scenario.

1.3. Organization of the Dissertation

This dissertation is organized into five chapters. In Chapter 2 a weighted-density-based approach is described for identifying items that can be predicted using the unstructured biological literature. An algorithm for distinguishing those pieces of information that

can be predicted, while other predictions might be spurious, is provided. The evaluation is done on data related to the model species yeast. Unstructured textual abstracts were used for identifying which protein domains and gene ontology annotations can be successfully predicted. This work was published in the SIAM SDM 2011 Text Mining Workshop proceedings [3].

Algorithms for mining significant information from structured binary data are provided in Chapter 3 for genome mapping. This chapter addresses the problem of identifying markers that cannot be placed reliably in the map and contribute to an overall poor mapping outcome if included. Description of the similarity network algorithm is provided. Our proposed algorithm largely matches other conventional approaches while reducing the computation cost by more than two orders of magnitude. The evaluation of the proposed approach is based on data from the radiation hybrid mapping of the wheat genome. This work was published at the ICMLA 2011 main conference [4].

The problem of creating solid framework genome maps is addressed in Chapter 4. The supported network algorithm is presented for this purpose. An iterative approach is followed for filtering unreliable markers to create a final solid framework map. The goal is to find a consistent map skeleton by filtering out unreliable markers. This work will be submitted to the BMC Bioinformatics journal. Chapter 5 concludes the dissertation.

CHAPTER 2. A WEIGHTED DENSITY-BASED APPROACH FOR IDENTIFYING STANDARDIZED ITEMS THAT ARE SIGNIFICANTLY RELATED TO THE BIOLOGICAL LITERATURE

Chapter 1 introduced different applications for mining for significance from unstructured and structured data forms. In Chapter 2, a weighted-density-based approach is introduced for identifying the significance between item data and unstructured textual information.

2.1. Abstract

A large part of scientific knowledge is confined to the text of publications. An algorithm is presented for distinguishing those pieces of information that can be predicted from the text of publication abstracts from those, for which successes in prediction are spurious. The significance of relationships between textual data and information that is represented in standardized ontologies and protein domains is evaluated using a density-based approach. The approach also integrates a weighting system to account for many-to-many relationships between the abstracts and the genes they represent as well as between genes and the items that describe them. We evaluate the approach using data related from the model species yeast, and show that our results are in better agreement with biological expectations than a comparison algorithm.

2.2. Introduction

Much information in the sciences is stored in textual form, whether in scientific publications or on the World Wide Web [69, 62, 43]. It is tempting to use this information directly for prediction purposes rather than making an effort of representing experimental results in a structured form. Controlled vocabularies, such as ontologies [9, 37], which are more directly suited to predictive modeling, have been developed in many fields, especially in the life sciences, but training scientists to use them is time consuming. This

paper presents an algorithm that evaluates the usefulness of text in predicting different potential class labels by testing for significant relationships between the attributes and the text data. The rationale is that knowing whether prediction from text is promising may be as important as the prediction result itself.

A common task in bioinformatics is the prediction of protein function [9]. When scientific abstracts are to be used for the prediction, it can not only happen that an abstract relates to more than one gene, and correspondingly protein, but also that the gene is discussed in more than one abstract. In other words, the textual documents are often related to attributes in a many-to-many fashion, resulting in a need for multi-relational techniques. Standard classification algorithms, when applied to one joined table of document-word and protein-function information, may erroneously appear to produce significant classification results. One could consider using relational techniques such as probabilistic relational networks [33]. Documents are, however, typically represented using the bag-of-words model, which results in high-dimensional vectors that are not suitable towards techniques that are derived from a Bayesian framework. Density-based approaches, in contrast, scale well with high dimensions. We demonstrate that the significance of the relationships between the textual information and functional annotations can be tested using density-based techniques with a suitable re-weighting scheme.

Figure 4 illustrates the multi-relational nature of the problem of predicting functional annotations from publication abstracts. Document records correspond to publication abstracts, and their attributes are the normalized frequencies of all words in the textual corpus according to the bag-of-words representation. Gene records hold the binary information on presence or absence of protein domains or gene ontology items. Each document record may be related to multiple gene records if the publication abstract refers to more than one gene, and genes may be discussed in any number of publication abstracts. The DocumentGene table captures this many-to-many relationship.

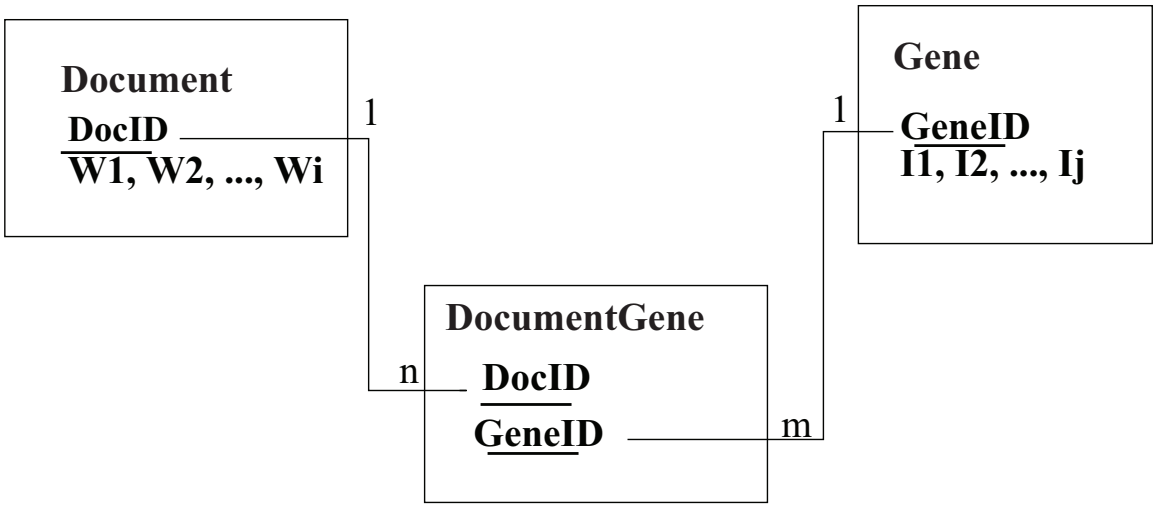

Figure 4: Relational skeleton of our problem domain. Notice the many-to-many relationship problem between documents and genes.

In this chapter, we propose an algorithm for evaluating whether the text data represented in the Document table have the potential of allowing the prediction of the protein domains or gene ontology items in the Gene table. For this purpose we use the concept of vector-item patterns [13]. The density-based nature of this approach allows an integration of the multi-relational nature of the problem through a re-weighting scheme that is similar to the term weighting common in text data mining. As a result, we show that predictions may be spurious even if they appear strong when classification is performed on the table that results when joining the Document, DocumentGene and Gene tables of Figure 4. Our goal is to develop means for identifying those properties that can be successfully predicted from text.

Figure 5 illustrates the problem of identifying significant relationships between multiple continuous attributes and items, which can be considered as potential class labels. The upper part of the schematic shows data points in 2-dimensions, with each data point representing a text document. In a realistic example, the space would have as many dimensions as there are words in the corpus, but the concepts can be illustrated in this simple setting: The potential class label or item can be seen as selecting a subset of

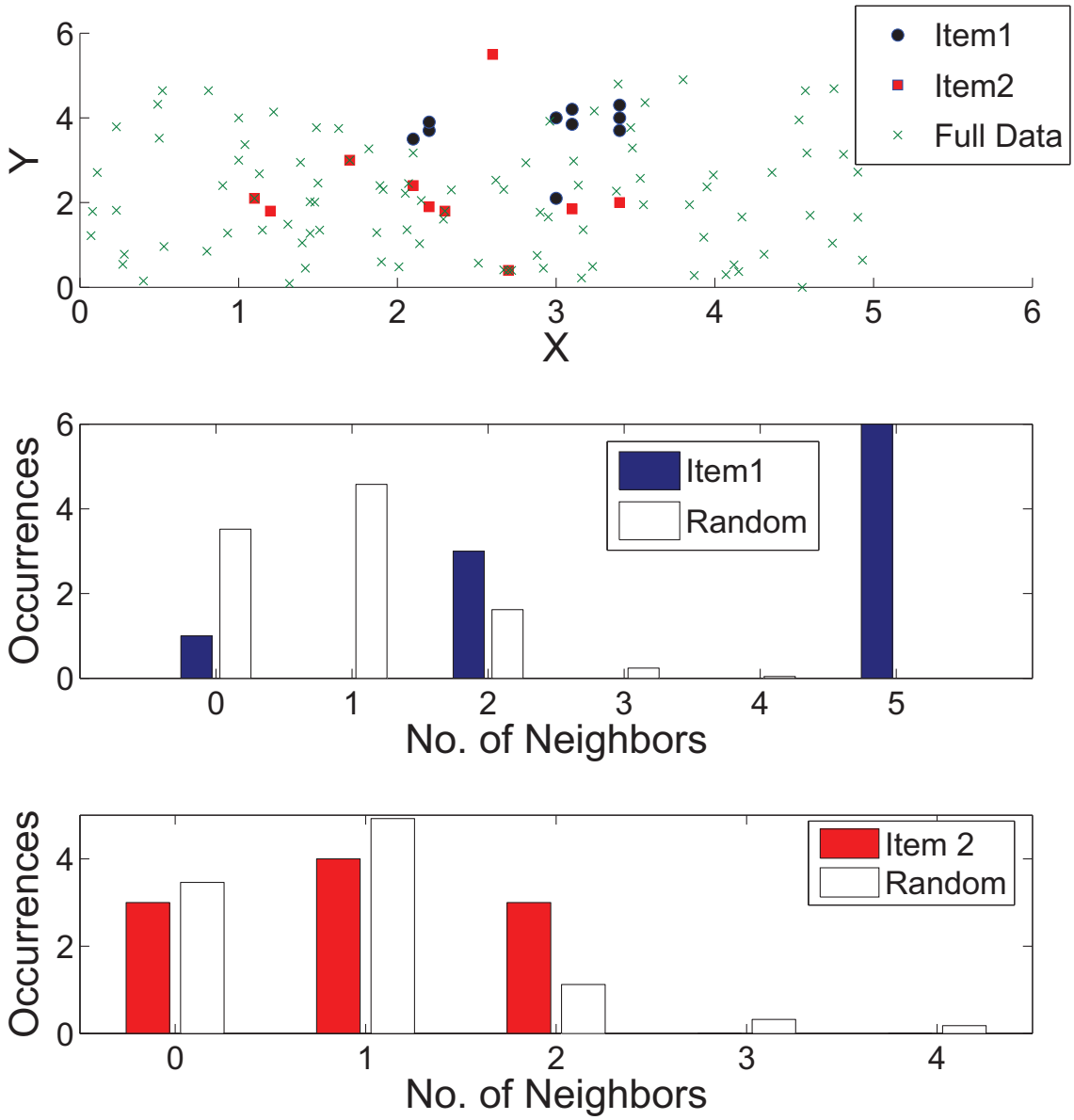

Figure 5: Schematic of a vector-item pattern between a 2-dimensional vector and two items. Blue filled circles represent objects that have item 1. Objects that have item 2 are shown as red filled squares. The remaining data set is represented as crosses. Middle: Objects with item 1 have more neighboring objects that also have item 1 than would be expected by random chance; vector-item pattern present. Bottom: Distribution of objects with item 2 does not differ significantly from the expected distribution; no vector-item pattern present.

data points. These items could be protein domains or gene ontology items. In Figure 5 two example items are shown, item 1 being represented by blue filled circles and item 2 by red filled squares. Data points that do not have either of these items are shown as

crosses. If the distribution of these items has the same statistical properties as a random subset, we conclude that there is no relationship. The statistical properties are summarized using histograms of the occurring densities. Densities are calculated as the number of neighboring data points with a cosine similarity that is larger than a predefined threshold. The density histogram of each item is compared with the average over histograms for several (in this case 20) random subsets using a χ^2 goodness-of-fit significance test. In Figure 5 the distribution of item 1 (middle of diagram) differs significantly from a random distribution. Therefore, item 1 represents a strong pattern. In contrast, the distribution of densities of item 2, which is shown on the bottom of the diagram, does not show a pattern.

Another way to calculate items significance is by calculating p-values from contingency tables [20]. For comparison purposes, we classified the same item data using naive Bayes classifier. We treated the classification results as contingency tables and calculated χ^2 goodness-of-fit. More details can be found in [20]. We proved that our results are more reliable than the comparison approach. It will be shown in the evaluation, that the patterns confirm the expectation that gene ontology item information often is significantly related to text, while protein domain information typically is not.

2.3. Related Works

Text mining [21, 44, 64, 25] is of interest in many areas, such as in bioinformatics. Recently text mining has become a focus area in genomics [69, 62]. Lexical methods have also been used on genomic sequences themselves [30]. Work has been done on discovering links and relationships of biomedical terms from biomedical text literature [43]. Natural language processing (NLP) techniques have been applied for biomedical text collections [28] and for classification [58].

Classification has been studied for text data [65, 57, 64, 25]. However, questioning whether textual data can lead to a successful classification of protein domains and ontology functions remains a major research question. Some significance tests have been applied for

testing classification results. The significance of gene ranking was studied in [67, 59]. A comparative studies of the significance tests used for information retrieval was conducted in [56, 55].

Probabilistic relational models (PRMs), which have been introduced in [33] are strong representations for structured relational data. These PRMs combine Bayesian networks with object and relational models [33]. PRMs specify probability distributions for the objects' attributes in the relational skeleton of the structured database. Specifying this probability distribution is done by defining the relational model of the domain and the dependencies between attributes by assigning parent-child relationships. The PRMs discussed in [33] are most suitable for domains that have objects with a limited number of attributes. However, in text mining, we almost always have a large number of attributes. Since the dominant textual representation is the bag-of-words [68] model, having several hundreds or even thousands of attributes is common. The time for constructing the dependencies between attributes does not scale well with the number of attributes.

Some work has been done to address the problem of a large number of attributes proposed in [42] by using Bayesian multinets. Bayesian multinets build a tree-like network that is used in the learning task. Since our approach is density based, it does not depend strongly on the number of attributes and scales well to high dimensions.

2.4. Concepts

In this chapter we introduce an algorithm for testing if textual information can be used for a successful classification of gene ontology items and protein domains. Our approach is to build a density histogram for every item (gene ontology item or protein domain). We compare the observed density histograms (of item data) with expected density histograms (of random data of equal size at large sampling rate). We measure an existence of a pattern based on the χ^2 goodness-of-fit test. If item data are significant; their textual information can be used in a classification task for predicting gene ontology items and protein domains.

Within our proposed framework we address the problem of many-to-many relationships by assigning appropriate weights to document vectors. The outline of our algorithm is:

- Data pre-processing
  - Stop word removal
  - Stemming
  - Term weighting
- For each item
  - Document vectors re-weighting
  - Construction of density histogram
  - Construction of expected histogram
  - Determining item significance (χ^2)

In the remainder of this section, data pre-processing is covered in Section 2.4.1, data representation is covered in Section 2.4.2, our proposed re-weighting framework is discussed in Section 2.4.3, deriving observed density histograms of item data are discussed in Section 2.4.4, computing expected density histogram is discussed in Section 2.4.5, significance testing is covered in Section 2.4.6, and the comparison algorithm is explained in Section 2.4.7.

2.4.1. Data Preprocessing

We apply the standard preprocessing steps that are commonly used in text mining [6, 8]. First, we remove stop words from every text document. Stop words are words that occur frequently in the text and are not predictive of any class label. We also remove other elements that are not useful within the bag-of-words model, such as digits, special symbols, punctuation marks, etc. Secondly, we apply stemming which has been shown to

have a positive impact on text mining and information retrieval systems [6], using porter stemmer [47]. Thirdly, we limit ourselves to terms that can be found in the standard English dictionary. The corpus used in the evaluation contains many names and identifiers that are not useful within the bag-of-words model. Finally, we use standard text normalization (TF*IDF)[17, 29] to weight terms in the text documents. Using this scheme each abstract document is represented by a vector of weighted terms (stemmed terms of those terms that can be found in standard dictionary).

2.4.2. Text Representation

Each text document d ∈ D is represented by a vector of weighted terms, where |D| is the total number of text documents in the corpus. The j^{th} document is represented as $\vec{d_j}$ = { w_{1j}, w_{2j}, ..., w_{nj} }, where n is the total number of terms in the corpus. $\vec{d_j}$ is the vector of the j^{th} document, and w_{ij} is the weight of term *i* in document *j*. Term weights are calculated using the expression:

$$w_{ij} = \frac{f_{ij}}{\max_l f_{lj}} \left[\log(\frac{\mid D \mid}{\mid d : m_i \in d \mid + 1}) \right]$$

where f_{ij} is the frequency of term *i* in document *j*, $\max_l f_{lj}$ is the frequency of the most frequent term in document *j*, and $\mid d : m_i \in d \mid$ is the number of documents that contain term *i*.

Item data (T: both gene ontology items and protein domains) are represented as bit vectors. Each item $t_k \in$ T, is a vector of zeros and ones of length = |D|. For each item we aggregate documents from DocumentGene table described in Fig. 4. If document d_j is related to item t_k then position *j* of the item vector is set to 1, otherwise 0.

2.4.3. Vector Re-weighting

The term weighting that is used for text documents can be seen as a way of addressing the imbalances in the number of terms associated with each document, because of varying document lengths, and the frequency with which terms appear in documents, because

of different term usage. A second need for re-weighting comes from the many-to-many relationship problem between documents and genes: Documents can be associated with a varying number of genes as seen in Figure 4 and genes can be discussed in a varying number of documents. The third need for re-weighting results from the nature of the Gene table: Each gene is associated with a number of items, that may also vary depending on how well-studied the gene is. The items are, in turn, associated with a varying number of genes, depending on how commonly the corresponding gene property is found. This section will discuss these additional two re-weighting schemes.

The problem can also be stated using two bipartite graphs that link documents to genes and genes to items. The first bipartite graph links the two disjoint sets, documents D and genes G, while the second bipartite graph links the two disjoint sets, genes G and items T. Consider the following two definitions:

Definition 1 (Document-gene bi-adjacency matrix)**.** Let G_{DG} = (D, G, $E^{(1)}$) be the bipartite graph between the two disjoint sets, documents (D) and genes (G), where D= $\{d_1, d_2, \ldots, d_n\}$ and G= $\{g_1, g_2, \ldots, g_m\}$, and let $E^{(1)}$ be the set of edges between these two disjoint sets. We define the bi-adjacency matrix $B^{(1)}$ as $B^{(1)}_{ij}$ = 1 if $(d_i, g_j) \in E^{(1)}$ and $B^{(1)}_{ij}$ = 0 otherwise.

Definition 2 (Gene-item bi-adjacency matrix)**.** Let G_{GT} = (G, T, $E^{(2)}$) be the bipartite graph between the two disjoint sets, genes (G) and items (T), where G= $\{g_1, g_2, \ldots, g_m\}$ and T= $\{t_1, t_2, \ldots, t_k\}$, and let $E^{(2)}$ be the set of edges between these two disjoint sets. We define the second bi-adjacency matrix $B^{(2)}$ as $B^{(2)}_{jl}$ = 1 if $(g_j, t_l) \in E^{(2)}$ and $B^{(2)}_{jl}$ = 0 otherwise.

Our proposed re-weighting scheme is inspired by the TF*IDF standard term weighting discussed in Section 2.4.2 with some adaptation. The main difference between our re-weighting measure and the TF*IDF measure is that the TF*IDF measure depends on the word counts while our measure depends on the relation existence. The TF*IDF weighting was developed to address the many-to-many relationship between words and documents.

Any one word can occur in several documents, and each document contains many words. This many-to-many relationship can be represented as a bipartite graph, in much the same way as the relationships between documents and genes and between genes and items. For the first bipartite graph, each document is linked independently to a varying number of genes and each gene can be linked to many documents. For the second bipartite graph, each gene is linked to many items (for example a gene can be annotated to many gene ontology items) and also each item is related to many genes independently of other items.

TF*IDF measures the importance of a word to a document. This measure of TF*IDF depends on the frequency of the words inside a document and its occurrence on other documents. However, some changes are necessary since our problem statement is different in some ways. Namely, the first part of the (TF) measure depends on the word count, while in our problem statement we are dealing with simple existence relationships. A document can be either linked to a gene or not linked at all, and similarly the relationship between a gene and an item. Hence, replacing the first part of the measure by a constant is mandatory.

Our proposed re-weighting measure is composed of 2 parts. The first part is a constant weight depending on the existence of the relationship between the two disjoint sets on its corresponding bipartite graph. The second part is derived in the same way of deriving the standard IDF measure. Our re-weighting scheme gives a measure of the importance of a documents to a gene, and for a gene to an item.

To illustrate our re-weighting scheme consider the first bipartite graph G_{DG} = (D, G, $E^{(1)}$) between documents D and genes G. First, for every document d_i we check for the existence of the relationship to the set G. If document d_i has any edge to the set G we give it a constant weight normalized by the maximum number of links from documents D to genes G. According to this, document d_i will have a weight:

$$W_i^{(d)} = \frac{1}{\max_l(\sum_{j=1}^{|G|} B_{lj}^{(1)})} \tag{1}$$

Following the same analogy of computing the IDF in text, the inverse gene frequency of gene g_j will be the natural logarithm of the total number of documents divided by the number of links between gene g_j and the set of documents D. (We add 1 to the denominator to avoid division by zero in case a gene is not linked to any document).

$$IW_j^{(g)} = \log[\frac{|D|}{\sum_{i=1}^{|D|} B_{ij}^{(1)} + 1}] \tag{2}$$

To calculate the relative weight of documents d_i in gene g_j, we multiply the constant weight of document d_i (Equation (1)) by the inverse gene frequency of gene g_j (Equation (2)).

$$RW_{ij}^{(d)} = W_i^{(d)} * IW_j^{(g)} \tag{3}$$

Weights for the second bipartite graph between genes and items are derived correspondingly. If gene g_j has any edge to the set T of items, its weight will be:

$$W_j^{(g)} = \frac{1}{\max_l(\sum_{k=1}^{|T|} B_{lk}^{(2)})} \tag{4}$$

Similarly, the inverse item frequency is:

$$IW_k^{(t)} = \log[\frac{|G|}{\sum_{j=1}^{|G|} B_{jk}^{(2)} + 1}] \tag{5}$$

To calculate the relative weight of gene g_j in item t_k, we multiply the weight of gene g_j (Equation(4)) by the inverse item frequency of t_k (Equation(5)).

$$RW_{jk}^{(g)} = W_j^{(g)} * IW_k^{(t)} \tag{6}$$

The two derived matrices (Equations(3 and 6)) are multiplied to give the total weight:

$$RW_{ik} = \sum_{j=1}^{|G|} RW_{ij}^{(d)} * RW_{jk}^{(g)} \tag{7}$$

Finally, we normalize our derived re-weighting factors using standard maximum normalization, which results in a re-weighting factor for each document relative to each item in the range [0,1]. Each re-weighting factor is multiplied by its corresponding document vector that we derived in Section 2.4.2 before deriving both observed and expected density histograms. We can imagine the re-weighted vectors as a 3-dimensional array of terms, documents, and items. The re-weighted vectors are calculated using the below expression:

$$\forall_{k=1}^{|T|} t_k : w_{ijk} = w_{ij} * RW_{jk} \tag{8}$$

where i, j, and k represents the term index, document index, and item index respectively.

2.4.4. Deriving Observed Density Histograms

The observed textual information, associated with each item, is summarized using a histogram. For each item $t_k \in$ T, we consider the set of genes S_k= $\{ g_j \in G \mid B_{jk}^{(2)} = 1 \}$ that represents this specific gene ontology item or protein domain. For each gene g_j we consider all abstract documents that are related to this gene. The set of documents is $\{ d_i \in D \mid B_{ij}^{(1)} = 1 \}$. The union of all these sets represents all abstract documents related to this specific item. According to this notation, each item is represented by $\bigcup_{j=1}^{|S_k|} \{ d_i \in D \mid B_{ij}^{(1)} = 1 \}$.

To derive the observed density histograms for each item, we need to calculate the number of neighbors for each data point that belong to each item. To determine neighbors we consider the following function:

$$\phi(d_i, d_j) = \begin{cases} 1, & \text{if } \cos(d_i, d_j) \geq h \\ 0, & otherwise \end{cases}$$

The function ϕ is the neighbor function determiner between any two documents d_i and d_j. This function is 1 if the cosine similarity between the two documents exceeds a specific predefined threshold *h*.

Definition 3 (Neighborhood selector function)**.** Document d_i is a neighbor to document d_j if and only if $\phi(d_i, d_j) = 1$.

The observed density histograms are calculated as follows: For each document d_j that belong to item t_k we determine its number of neighbors (assume n), then we increment the density histogram at the point n by 1. After finding the number of neighbors for every text document that belongs to item t_k, we derive a density histogram. This density histogram represents the number of neighbors for each text document versus their occurrences.

2.4.5. Computing Expected Histogram

For each item $t_k \in T$, we also computed its corresponding expected density histogram: Assume that item t_k has m text documents associated to it. We calculate the expected density histogram for item t_k by random sampling. For each of the *r* samples, we select a random subset of m documents and compute a density histogram. For each document in the random subset, we determine its number of neighbors using the ϕ function. After examining each text document in the random subset, we build an expected density histogram using the same terminology of building the observed ones. For every item, we calculate the expected density histogram by averaging over 20 histograms derived from random sampling.

2.4.6. Significance Test

A χ^2 goodness-of-fit test is used to determine if the observed density histogram differs from the expected one in a statistically significant way. We use a 99% significance level. If the p-value from the χ^2 goodness-of-fit test is less than 0.01, we consider this item to be significant. A p-value of 0.01 means that in 1% of cases, by random chance alone, we expect to see a result that is as extreme or more extreme.

2.4.7. Comparison Algorithm

As a comparison approach the χ^2 test is used to compare our results of significance (calculated from comparing density histograms) with the classification significance of naive Bayes classifier (calculated from comparing contingency tables). We have tested the output significance of the naive Bayes classifier. Every confusion matrix resulting from classifying each item was treated as a contingency table. We have carried out χ^2 test on each confusion matrix using one degree of freedom (since the confusion matrix consists of two rows and two columns). Details of calculating p-values from contingency tables can be found in [20]. Table 2 in Section 2.5.2 shows how we calculated the classification significance from the classification confusion matrix. The results of the two methods are discussed in the next section.

2.4.8. Algorithm

The details of the density histogram approach can be seen in Algorithm 1. The inputs of the algorithms are the unstructured textual corpus D and the set of items T of protein domains and gene ontology. The outputs are the obtained $p-$values for each item. Pre-processing steps explained in Sections 2.4.1 and 2.4.2 are carried out in lines 2 to 4. In Lines 6 to 11 we re-weight the vectors of the text documents for each item (Section 2.4.3) and we calculate the observed density histograms explained in Section 2.4.4. The expected density histograms are calculated in lines 12 to 18. Finding the significance for each item using χ^2 goodness-of-fit is carried out in line 19.

2.5. Experimental Results

We consider the model species yeast to evaluate our algorithm. This data was the training data for task 2 competition in KDD cup 2002 (http://www.sigkdd.org/kddcup/index.php?section= 2002&method=task). The textual data related to this task consists of 15234 scientific abstracts of publications, 18.9 MB in total. These abstract documents were originally downloaded from the MEDLINE database in NCBI web site (www.ncbi.nlm.nih.gov-

Algorithm 1: Density Histogram Algorithm.

```
Data: Docs;                                          /* Textual corpus */
Data: items;          /* gene domains and ontology functions */
Result: significance;                  /* p-value for each item */
foreach d ∈ Docs do
    StopWordRemoval(d);
    Stemming(d);
    TermWeighting(d) ;    /* vectors of weighted terms TF.IDF */
foreach i ∈ items do
    d_n=FindRelatedDocuments(i);    /* d_n ⊂ Docs related to item i */
    hist = zeros(1,NoOfElements(d_n));                  /* initialize */
    foreach d ∈ d_n do
        Reweighting(d);
        density = NumberOfNeighbors(d);
        hist(density)++;
    randHist = zeros(1, NoOfElements(d_n));
    for i=1 to SamplingRate do
        randDocs = SelectRandSubset(Docs,NoOfElements(d_n));
        foreach d ∈ d_n do
            density = NumberOfNeighbors(d);
            randHist(density)++;
    randHist = randHist/SamplingRate;
    significance(i) = chiSquaredGoodnessOfFit(hist, randHist);
return significance
```

/entrez/query.fcgi). Abstract documents associated the genes through the pointers of Saccharomyces Genome Database (genome-www.stanfo-rd.edu/Saccharomyces/) that are related to these scientific publications. There are 5013 protein domains in this data set. Only 1547 domains have been tested (those who have at least 10 abstract documents related to them). We also applied the same algorithm on ontology functions (GO slim). Following the same criteria of selecting items, we have considered 85 ontology functions out of 112 functions, which have at least 10 documents each. We have evaluated our algorithm on both protein domains and ontology functions. We use 99% significance level to determine item significance (p-value < 0.01).

2.5.1. Test Cases

Initially, we use random test cases to verify that they are indeed predicted to be insignificant. Each test case represents a comparison of a random set of abstract documents of equal size that correspond to random selection of genes. A density histogram of each test case is created. Then, the algorithm is applied to test if there is a strong pattern within this random test set. In total, 30 test cases of different sizes of random data were created. Using our algorithm, none of these test cases were found to be significant. The p-values of these test cases was in the range [0.57, 1]. As expected, these p-values indicate that the random histograms are insignificant.

2.5.2. Protein Domain Results

When testing, whether textual information is related to protein domain information we expect many domains to be insignificant, since the sequence information may not be represented in the articles written about the genes or proteins. We do expect some significant domains, since protein domains may be associated with functional information. We will show that the proposed algorithm finds both significant and insignificant domains whereas the comparison algorithm results in significant predictions for almost all domains.

Among the 1547 tested domains, we have identified 876 protein domains with strong patterns (i.e their textual data truly represents their particular domains). Also, we have determined 671 non-significant domains, that their distribution do not differ significantly from what we expect of random distribution. Below we will show two examples of two real domains along with their density histograms. The first one is the G3DSA 1.10.510.10 domain. Figure 6 shows the density histogram of this domain comparing it with the expected random distribution. This domain has been identified to be significant by our algorithm and by the χ^2 test of the confusion matrix from naive Bayes classifier (p-Value = 0 for both algorithms; i.e the p-value is too small to be represented). Since this domain has a know 3-d structure, it is plausible that its function is well enough preserved to result

in a function that is reflected in publications. The identification as not random, is therefore credible.

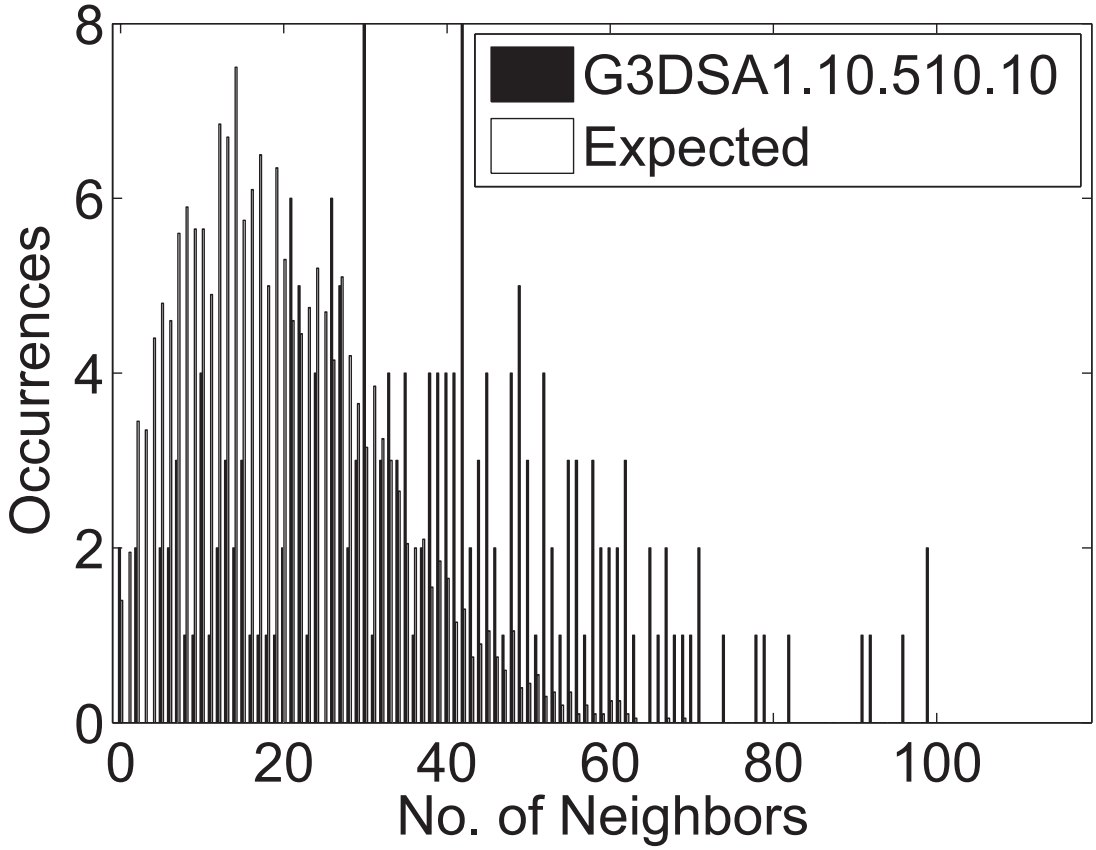

Figure 6: Density histogram of a real protein domain (G3DSA 1.10.510.10) that shows a significant pattern. Filled columns represent observed data of the domain. Unfilled columns represents the average over histograms of 20 random subset of abstract documents of equal number of the observed documents. Using the density histogram algorithm the domain was found to be significant.

The second example is the SSF52833 domain. This domain corresponds to a superfamily and is not likely to result in a particular type of abstract because superfamilies group proteins with too many different functions. Our algorithm appropriately identifies this domain as non-significant. Figure 7 shows the observed distribution of this domain comparing it with the expected distribution of random data. It can be seen from the density histogram that the two distributions do not differ significantly. For comparison purposes we classified the same domain data (textual documents) using naive Bayes classifier, and

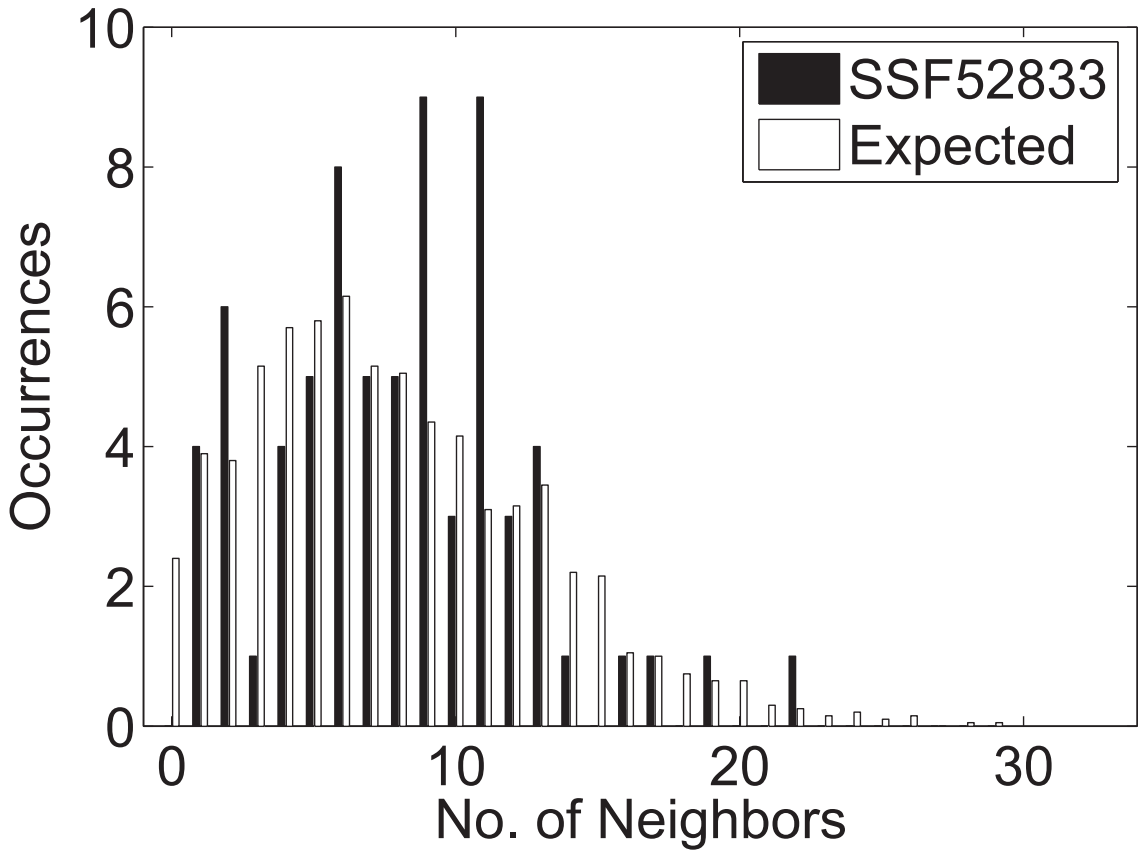

Figure 7: Density histogram of a real protein domain (SSF52833) that does not show a significant pattern. Filled columns represent observed data of the domain. Unfilled columns represents random subset of abstract documents of equal number of the observed documents at sampling rate=20. Using density histogram algorithm the domain found to be non-significant.

calculated the significance of the classifier output (confusion matrix). Using a χ^2 test of the confusion matrix this domain was considered significant. Table 2 shows the classification results of this domain. The table also illustrates the process of calculating the p-value for the comparison approach. Table 2 part a) represents the confusion matrix of the classification. Part b) represents what we expect of classifying random data. Part c) shows how we calculate the p-Value using the equation $\sum(O-E)^2/E$. Although this is a non-significant domain, using the χ^2 test on the confusion matrix we have obtained a p-Value = 0; which means that the p-Value is below the accuracy of the number type. This assumes that this domain has a strong pattern and is significant, while our algorithm predicted it to be non

significant with a p-Value = 0.0946. This result highlights our main contribution on this paper. Our algorithm predicted many non-significant domains while other tests cannot distinguish these non-significant domains (see Table 3).

Table 3 compares the results of the top 5 significant domains and top 5 non-significant domains of the two algorithms. By top 5 we mean those domains that have the most relevant text document to them. We notice that the top 5 significant domains were identified by both algorithms to be significant. However, for the top 5 non-significant domains, the χ^2 test of the confusion matrix from naive Bayes classifier failed to identify 2 out of 5.

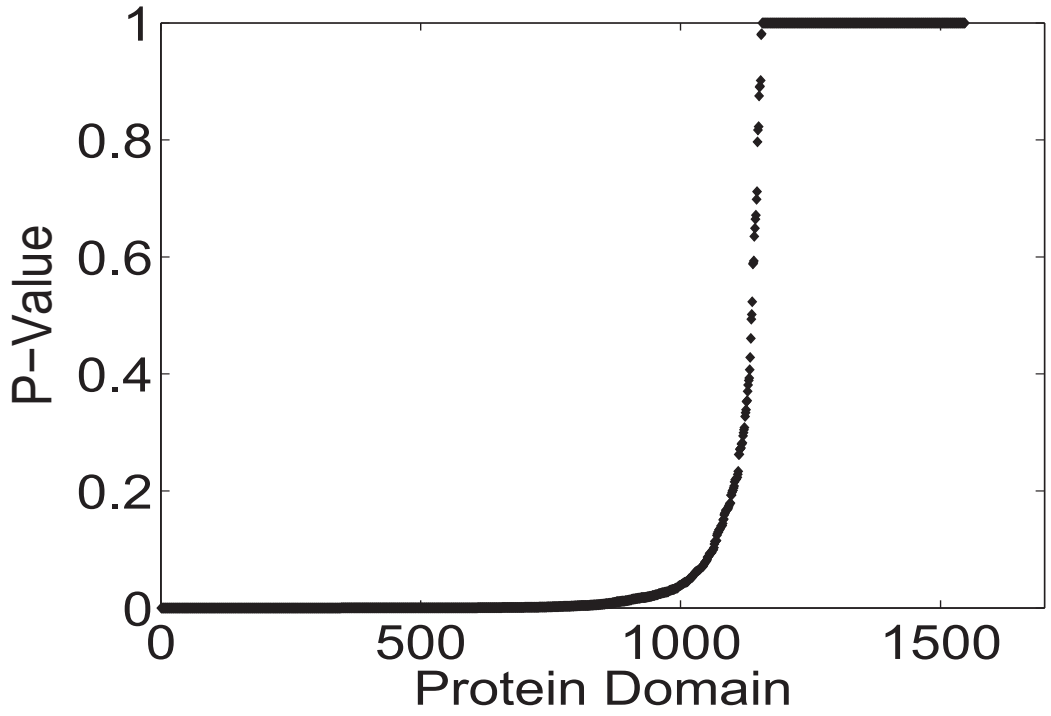

Figure 8: p-Value for all tested protein domains.

Figure 8 visualizes the results for the 1547 domains that we have tested sorted according to their p-values. We have identified 876 significant domains. Also, 671 non-significant domains were found. It can be inferred from this figure that the textual information of the significant domains strongly represents them and that the text is significantly related to these item data.

Table 2: Same data in Figure 6 classified using the naive Bayes classifier. The SSF52833 domain data is used as observed data. The confusion matrix for the Naive Bayes classifier is treated as contingency table, and its significance is tested using $\chi 2$ goodness-of-fit. a) Represent classifier results. b) Represents what we expect of classifying random data. c) the calculation of the $\chi 2$ goodness-of-fit test.

a)	Observed			b)	Expected		c)	$(O-E)^2/E$	
	1	0	Total		1	0		1	0
1	7	64	71	1	0.8016	70.198	1	47.927	0.547303521
0	165	14998	15163	0	171.2	14992	0	0.2244	0.002562722
Total	172	15062	15234						
							$\sum(O-E)^2/E =$		48.70152528
							p-Value		0

Table 3: Comparison between the results of the density histogram algorithm and the significance test of naive Bayes classifier. Top 5 significant domains (Upper part of table) and top 5 non-significant domains (lower part of table) are shown. Differences are highlighted in bold.

Domain	TP	FN	FP	TN	p-Value (confusion matrix)	p-Value (density histograms)
SSF52540	69	478	454	14233	0	2.11E-07
G3DSA 3.40.50.300	55	433	385	14361	0	2.07E-06
SSF48371	18	210	339	14667	0	0
SSF56112	51	173	495	14515	0	0
G3DSA 1.10.510.10	25	162	349	14698	0	0
SSF51735	6	114	289	14825	0.0145	0.50186
SSF48452	3	75	160	14996	0.0169	0.062985
SSF52833	7	64	165	14998	**0**	**0.0946**
PS00455	1	48	85	15100	0.1671	0.13
PF01842	2	35	48	15149	**0**	**0.15902**

2.5.3. Gene Ontology Annotation Results

The same algorithm has been applied to test the significance of gene ontology items. For gene ontology items we expect that many will be related to textual information, since publication abstracts are likely to be related to the function, process or localization of the protein. Note that, for simplicity, we will refer to the gene ontology items as ”functions” regardless of the actual category. The results confirm the biological expectation that most functions are significant but also present some exceptions of insignificant functions. These exceptions can be understood from a biology perspective. The insignificance of the highest-level items ”biological process”, and ”molecular function” confirms the biological expectations.

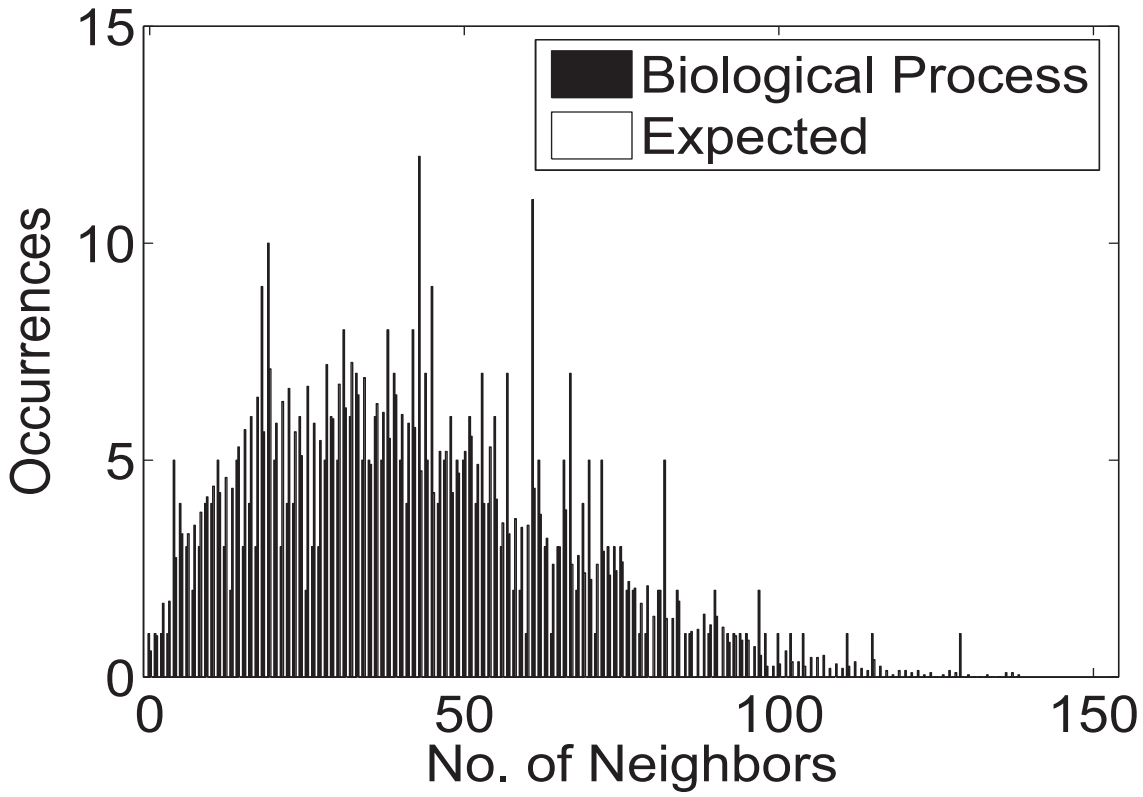

Figure 9: Density histogram of real non-significant gene ontology item (biological process). Filled columns represent observed data of the function. Unfilled columns represents random subset of abstract documents of equal number of the observed documents at sampling rate=20. Using density histogram algorithm the function found to be non-significant.

Biological process has been identified as non-significant function by our algorithm, while the significance test of naive Bayes confusion matrix could not identify it as non-significant. Figure 9 shows this distribution. Biological process is a general gene ontology item and it is located on the top level of the gene ontology tree. It is expected that an item that is at the top of the gene ontology items hierarchy, and does not contain any gene-specific information, is not a suitable candidate for prediction.

We also identified the insignificance of ”molecular function” item, which is also located at the top of the gene ontology items hierarchy. For this item we achieved the same result by the comparison algorithm.

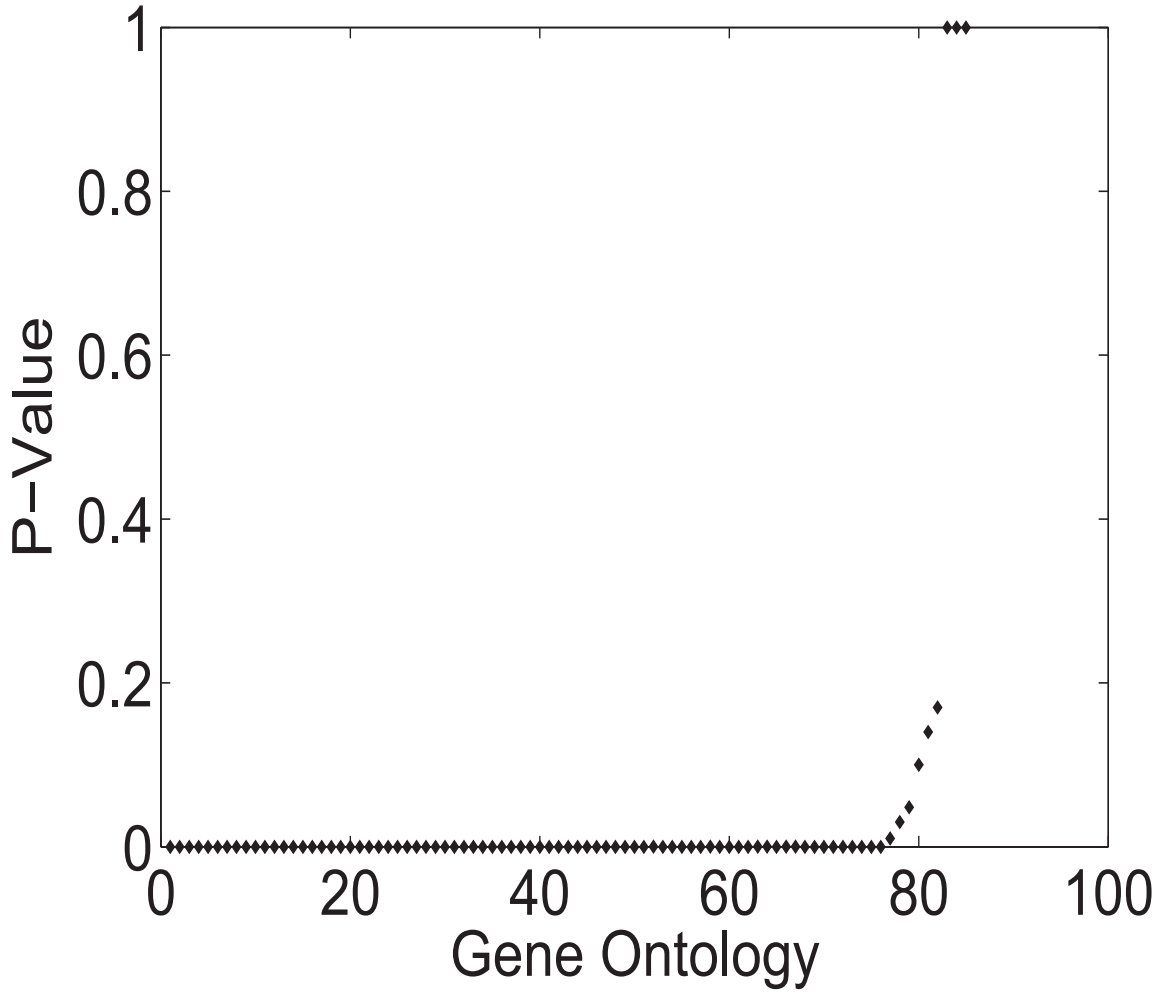

Figure 10: p-Value for all gene ontology functions.

Table 4: Comparison between the results of significance between density histogram algorithm & significance test of naive Bayes classifier.Top 5 significant ontology functions (Upper part of table) and top 5 non-significant ontology functions (lower part of table). Differences are highlighted in bold.

Ontology Function	TP	FN	FP	TN	P-Value (confusion matrix)	p-Value (density histograms)
Nucleus	145	618	992	13479	0	0
Cytoplasm	36	528	506	14164	0.0002	0
Organelle Organization and Biogenesis	66	371	506	14291	0	1.63E-07
Transcription	28	353	367	14486	0	0
DNA Metabolism	64	323	624	14223	0	0
Biological Process	28	353	367	14486	**0**	**0.1436**
Molecular Function	0	95	130	14998	0.3642	0.0482
Colocalizes Withmembrane	0	11	0	15223	1	1
Colocalizes Withvacuole	0	46	52	15136	0.691	1
Isomerase Activity	0	14	4	15216	0.9516	0.099158

We have tested 85 gene ontology items for significance. Only 8 ontology functions were found to be non-significant. We have noticed an agreement between our algorithm and the comparing approach of all gene ontology items except for the biological process function. Table 4 shows a comparison between the results of our algorithm and significance test of naive Bayes classifier for the top 5 significant ontology functions and top 5 non-significant functions. Figure 10 shows the obtained p-values for the 82 gene ontology functions.

2.6. Conclusion

In this chapter, we have presented an algorithm for identifying significant patterns between standardized items of information and textual representations of genomic information. The algorithm uses a re-weighting framework for document vector re-weighting that takes into account many-to-many relationships between documents and genes as well as between genes and item information. Our proposed re-weighted density-based algorithm correctly identifies some relationships as non-significant that are not expected to be significant based on domain knowledge, and that appear strong using Naive Bayes classification. Abstract text documents are represented using a vector space model. We evaluate the significance of patterns by considering their observed density histograms in comparison with expected ones. We compare with the results of a χ^2 test on the confusion matrix resulting from classification using the naive Bayes classifier. We evaluated the algorithm using publication abstracts as text data and protein domains and ontology functions as item data. We found our results to be in better agreement with biological expectations than the comparison results. As would be expected based on domain knowledge, many protein domain text relationships were insignificant according to our algorithm, far more than the comparison algorithm. Two highest-level gene ontology items that were expected to be insignificant were also confirmed as such by our algorithm but one of them was not by the comparison algorithm.

CHAPTER 3. NETWORK-BASED FILTERING OF UNRELIABLE MARKERS IN GENOME MAPPING

In Chapter 2, a weighted-density-based approach is discussed and applied for testing the significance of the unstructured data in predicting class labels. In Chapter 3 a network-based approach is presented for unreliable marker detection from the structured data.

3.1. Abstract

Genome mapping, or the experimental determination of DNA marker order on a chromosome, is an important step in genome sequencing and ultimate assembly of sequenced genomes. The presented research addresses the problem of identifying markers that cannot be placed reliably. If such markers are included in standard mapping procedures they can result in an overall poor map. Traditional techniques for identifying markers that cannot be placed consistently are based on resampling, which requires an already computationally expensive process to be done for a large ensemble of resampled populations. We propose a network-based approach that uses pairwise similarities between markers and demonstrate that the results from this approach largely match the more computationally expensive conventional approaches. The evaluation of the proposed approach is done on data from the radiation hybrid mapping of the wheat genome.

3.2. Introduction

Genome mapping [32] is important for determining the order of genes and markers (DNA sequences) within the chromosome of a species. It is an integral step in developing a marker scaffold, which is a prerequisite for the complete genome sequencing of a species. Molecular maps are also valuable for crop improvement and for identifying biotic and abiotic stress related genetic factors, both of which are of vital importance considering the increasing global demand for food and climatic changes. Radiation Hybrid (RH) mapping [24, 63, 32, 38, 34, 31] is a widely used mapping technique, in which parts of chromosomes are broken using radiation. This chapter addresses the problem of identifying markers

that cannot be consistently ordered and may, thereby, decrease the overall quality of the resulting map.

There are two potential problems in RH experimental data. First, some markers may have a higher percentage of miss-scorings for experimental reasons. Second, for some markers the amplification may not be sequence specific due to the repeated nature of underlying sequence. Recovering from scoring problems requires repeating the biological experiment. Since re-checking every single marker is costly, providing algorithms for detecting those unreliable markers will help in reduce the cost.

In this chapter we propose a fast algorithm for finding unreliable markers without using time consuming resampling techniques. The idea of our algorithm is to define neighbors based on markers LOD (logarithms of odds -base 10) scores. The LOD score is a measure of the likelihood that two markers are linked. Considering a fixed number of nearest neighbors, we construct a similarity network by linking only markers that are mutually neighbors to each other. Figure 11 shows the complete similarity network for a small artificial data set of 8 markers on 6 individuals visualized using graphviz [18]. The sequence $M1$ to $M8$ is the reference map created using all individuals. We filter markers based on a range of linkage. In this example if $r = 2$, markers M_3 and M_4 will be defined as unreliable because both of them failed to meet the linkage range of [1, 5] and [2, 6] respectively.

Another way to study the stability of the mapping results is through resampling analysis [23]. In resampling analysis, multiple data sets are created by sampling from the full data set. We use jackknife resampling, in which samples are created by considering all individuals except one. Consensus mapping proposed in [39, 40, 41, 51] depends mainly on mapping the re-sampled data and looking at the neighborhood relationships of the results. Unreliable markers are filtered iteratively based on the mapping results of the re-sampled data. Table 5 shows the mapping results for the same artificial data discussed in Figure 11.

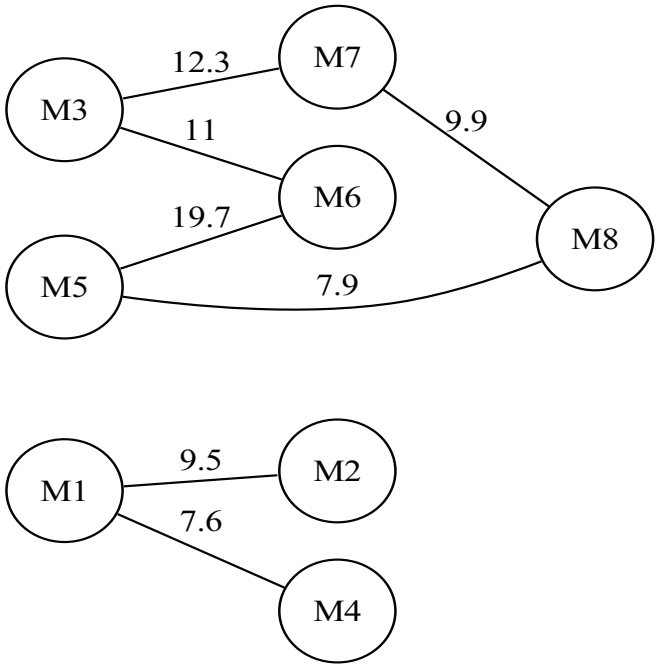

Figure 11: Similarity network for artificial data set. Nodes represent markers labeled according to their position in the reference map. Edges are labeled with LOD scores.

Table 5: Mapping results of data consist of 8 markers on 6 individuals. $J^{(0)}$ is the reference map created using all information. $J^{(l)}$, with $l \geq 1$ are the maps created using jackknife resampled data.

Data	Map
$J^{(0)}$	$M1, M2, M3, M4, M5, M6, M7, M8$
$J^{(1)}$	$M1, M2, M3, M8, M5, M4, M6, M7$
$J^{(2)}$	$M1, M2, M4, M3, M5, M6, M8, M7$
$J^{(3)}$	$M1, M2, M3, M4, M5, M6, M7, M8$
$J^{(4)}$	$M2, M1, M3, M7, M8, M4, M6, M5$
$J^{(5)}$	$M1, M2, M3, M5, M6, M4, M7, M8$
$J^{(6)}$	$M8, M7, M6, M5, M4, M3, M2, M1$

$J^{(0)}$ is the map created using all information. $J^{(l)}$, were $l \geq 1$ is the map created using jackknife re-sampled data using all individuals except individual l. The results can be summarized as a neighborhood matrix. $J^{(0)}$ is used as a reference map. For every map $J^{(l)}$ we look at the neighborhood relationship between every ordered pair of markers (M_i, M_j) and we increment both indexes (i, j) and (j, i) in the neighborhood matrix by 1. After

parsing every map we normalize by dividing by the number of individuals. Table 6 shows the neighborhood matrix for the maps in Table 5. This algorithm [39, 40, 41, 51] gives insights for the mapping results stability and can be used to filter unreliable markers. If we filter markers based on neighborhood threshold $t \leq 0.7$ in Table 6, the same markers M_3 and M_4 will be defined as unreliable as defined in Figure 11. However, the algorithm does not scale well with large data sets. Even with moderate-size data sets, distributed systems might be required to handle the intensive computation time. Construction of neighborhood matrix and markers filtering will be discussed in Section 3.4.2.

Table 6: Neighborhood matrix summarizes the mapping results shown in Table 5.

Mrk	$M1$	$M2$	$M3$	$M4$	$M5$	$M6$	$M7$	$M8$
$M1$	-	**1**	0.17	0	0	0	0	0
$M2$	**1**	-	**0.67**	0.17	0	0	0	0
$M3$	0.17	**0.67**	-	**0.5**	0.33	0	0.17	0.17
$M4$	0	0.17	**0.5**	-	**0.5**	0.5	0.17	0.17
$M5$	0	0	0.33	**0.5**	-	**0.83**	0	0.17
$M6$	0	0	0	0.5	**0.83**	-	**0.5**	0.17
$M7$	0	0	0.17	0.17	0	**0.5**	-	**0.83**
$M8$	0	0	0.17	0.17	0.17	0.17	**0.83**	-

Using our algorithm, only the original data set has to be mapped. Because defining the unreliable markers does not depend on the resampling analysis, computation time can be decreased dramatically. We will discuss the similarity network construction in more details in Section 3.4.1. We use the neighborhood matrix algorithm [39, 40, 41, 51] as a baseline model and compare with standard clustering provided by the Carthagene [12] software. We will show in the evaluation that our algorithm outperforms the comparison approach.

3.3. Related Works

RH mapping, first introduced by Goss and Harris [24], has been successfully used to map the human genome [63], animals [32] and most recently plants [32, 31, 34]. Unlike

genetic maps, RH maps [32] provide information about the physical distance between markers within the chromosome. Various mapping programs [12, 52] are available online for RH mapping. These programs use heuristic algorithms for finding the best map (or k maps), since RH mapping is related to the traveling salesman problem (TSP).

Resampling analysis [23] has been used in [39, 40, 41, 51] to check map stability and building map skeletons by filtering out unreliable markers by considering the neighborhood relationships between markers. However, resampling analysis does not scale well with the number of markers and individuals, which motivates the proposed similarity network algorithm.

3.4. Concepts and Algorithms

There is no ground truth for defining unreliable markers. For that reason we use the neighborhood matrix algorithm as baseline model. Markers that are considered to be unreliable by the baseline model are treated as truly unreliable markers. We evaluate our network algorithm (discussed in Section 3.4.1) against the baseline model (Section 3.4.2) and use clustering, which is provided by the Carthagene software [12], as comparison approach.

3.4.1. Construction of similarity network

Marker labeling: All markers are used in Carthagene to create a reference map. Each marker, M_i, is labeled according to its position, i, in the reference map.

Mutual K-Nearest Neighbors: For every marker in the data set, we find its k-nearest neighbors, according to the LOD score as a similarity measure. The higher LOD value between a pair of markers, the more likely those markers are to be linked. Our approach is computationally fast, since LOD scores do not depend on the mapping and are calculated upon loading the data set into Carthagene. The computational cost for calculating the LOD scores and finding the k-nearest neighbors for every marker is negligible in comparison

with the computational cost of mapping. We define the similarity network of markers as follows:

Definition 4 (Similarity network)**.** *Let $G = (M, E)$ be the undirected graph of the set of markers (M), where $M = \{M_1, M_2, \ldots, M_n\}$, and let E be the set of edges between markers. For every pair of markers $(M_i, M_j) \in E$ if and only if M_i is a neighbor to M_j and M_j is a neighbor to M_i. The k nearest neighbors for marker M_i, $KNN(M_i, k)$, are the k markers that have the highest LOD scores with respect to M_i:*

$$\forall_{i=1}^{n-1} \forall_{j=i+1}^{n} (M_i, M_j) \in E \quad \text{if } M_i \in KNN(M_j, k) \bigwedge M_j \in KNN(M_i, k) \tag{9}$$

We consider several parameter settings (k=3,5,7,10, and 15).

Marker Filtering: We define the unreliable markers, F, using two parameters k and r, where r is a range. M_i is defined to be unreliable if it is not linked to any marker in the range [M_{i-r}, M_{i+r}]:

$$\forall_{i=1}^{n} M_i : M_i \in F \quad \text{if } \nexists_{j=i-r}^{i+r} (M_i, M_j) \in E \tag{10}$$

Algorithm: The details of the process can be seen in Algorithm 2. The inputs of the algorithms are RH data, fixed number of k nearest neighbors, and range r. The outputs are the undirected graph G and the set F of unreliable markers. In line 1 we find the reference map using all individual information. Lines 2 and 3 reflects marker labeling explained in Section 3.4.1. In lines 4 to 7 the k nearest neighbors for every marker are found. Construction of the similarity network explained in Section 3.4.1 is carried out in

lines 8 to 11. In lines 12 through 19 we find the set F of unreliable markers explained in Section 3.4.1.

Algorithm 2: Similarity Network Filtering.

```
Data: RHData, M, I;                          /* RHDate: M by I matrix */
Data: k, r;                  /* K=number of neighbors, r:range element */
Result: G = (M, E);                          /* graph of connected markers */
Result: F ⊂ M;               /* list of potential markers to be filtered */
1  RefMap^(R) = Map(RHData) ;                /* find best map using carthagene */
2  foreach m ∈ M do
3      Label(m) = M&pos(m) ;                 /* label markers according to position */
4  KNNmatrix(n, k) = zeros(n, k);                                  /* initialize */
5  foreach m_i ∈ M do
6      for j=1 to k do
7          KNNmatrix(i, k)=KNN(m_i);
8  for j=1 to n-1 do
9      for l=j+1 to n do
10         if m_j ∈ KNNmatrix(l, k) AND m_l ∈ KNNmatrix(j, k) then
11             (m_j, m_l) ∈ E ;                 /* determine edges in graph */
12 foreach m_i ∈ M do
13     Flag = False;
14     foreach (m_i, m_j) ∈ E do
15         if m_j ∈ [m_{i-r}, m_{i+r}] then
16             Flag = True;
17             exitFor;
18     if Flag = False then
19         m_i ∈ F;
20 return G;
21 return F;
```

3.4.2. Construction of Neighborhood Matrix

Jackknife Resampling: The RH data is represented by a matrix of n rows (markers) and v columns (individuals), where each entry in the matrix is a binary value representing the presence or absence of a marker i in individual j:

$$RHData_{n,v} = \begin{pmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,v} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,v} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n,1} & d_{n,2} & \cdots & d_{n,v} \end{pmatrix}$$

where,

$$d_{(i,j)} = \begin{cases} 1, & \text{if marker } i \text{ is present in individual } j \\ 0, & \text{if marker } i \text{ is absent in individual } j \\ -, & \text{missing information} \end{cases}$$

To check the mapping consistency, we resample the RH data using jackknife resampling technique and build a map for every resampled data set. By removing one individual at a time from the mapping population, we create as many data sets as there are individuals in the mapping population.

RH Mapping: We use Carthagene software [12] for the actual mapping. We use the Linux version and have it installed on 120 high performance computing machines, each with 8 processors. Such computational power is required to map all resampled data. The following mapping strategy is used for all the resampled data:

- Merging double markers
- Pattern expansion algorithm (build)

- Greedy search algorithm
- Genetic algorithm
- Simulated annealing algorithm
- Sliding window permutations

We first merge groups of markers that have identical mapping information (double markers) and represent them by a single marker. Second, we build an initial map using a heuristic (build) that starts with the pair of most strongly linked markers and inserts the remaining markers incrementally. Third, we try to enhance the map by using the greedy search algorithm. Fourth, we use both a genetic and a simulated annealing algorithms to find a better map in case of a local optimum. Finally, we use a fixed sliding window to try all permutations within the window and check if a better map is achieved.

Neighborhood Matrix: The neighborhood matrix summarizes the mapping results for all the maps created using the resampled data. To construct the neighborhood matrix, we first use the map created using all individual information $(J^{(0)})$ as a reference map. Second, we treat every map created using resampled data $(J^{(l)})$, where $l \geq 1$ as an order list. For every neighboring pair of makers in the ordered list, we find the positions of those markers in the reference map and we increment the entry in the intersection of both indexes by 1. We parse the maps in both directions to maintain the information about the order of the map and its reverse. For each pair of markers $J^{(l)}_{(i)}$ and $J^{(l)}_{(i+1)}$ we increment the neighborhood matrix N for the corresponding index pair.

We normalize the neighborhood matrix by dividing by the number of individuals. Using this scheme every entry in the matrix is a value in the range [0, 1], where 1 means the pair of markers are always in the same order for every re-sampled data map:

$$\forall i \forall j N(i,j) = \frac{1}{v} \sum_{l=1}^{v} \sum_{k=1}^{n-1} (J_k^{(l)} = i)(J_{k+1}^{(l)} = j) \tag{11}$$

Marker Filtering: The neighborhood matrix can be used to define the set of unreliable markers. If the maps constructed using the resampled data are consistent with the reference map, the entries immediately beside the diagonal of the neighborhood matrix will have values close to 1. We define the set F of unreliable markers as follows: for every marker in the reference map, we check its neighborhood values in the neighborhood matrix. If the maximum neighborhood value is below a specific threshold t we define that marker as unreliable:

$$\forall_{i=1}^{n} J_{(i)}^{(0)} : \begin{cases} J_{(1)}^{(0)} \in F, & \text{if } N(J_{(1)}^{(0)}, J_{(2)}^{(0)}) \leq t \\ J_{(i)}^{(0)} \in F, & \text{if } max(a,b) \leq t \\ J_{(n)}^{(0)} \in F, & \text{if } N(J_{(n-1)}^{(0)}, J_{(n)}^{(0)}) \leq t \end{cases}$$

where, $a = N(J_{(i-1)}^{(0)}, J_{(i)}^{(0)})$ and $b = N(J_{(i)}^{(0)}, J_{(i+1)}^{(0)})$

Algorithm: The details of the process can be seen in Algorithm 3. The inputs of the algorithms are RH data and a predefined neighborhood threshold t. The outputs are the neighborhood matrix N and the set F of unreliable markers. In line 1 we find the reference map using all individual information. Jackknife resampling and mapping the resampled data is carried out in lines 2 to 4. Lines 6 to 14 reflects the creation of the neighborhood matrix explained in Section 3.4.2. In lines 15 to 17 we find the set F of unreliable markers explained in Section 3.4.2.

3.5. Experimental Results

3.5.1. Data Set

A radiation hybrid population of 1542 RH plants was generated in the laboratory for D-genome chromosomes and genotyped initially using 35 SSR (Simple Sequence Repeat) markers (5 from each chromosome) selected across the seven D-genome chromosomes of

Algorithm 3: Neighborhood Matrix Filtering.

Data: $RHData, M, I$; /* RHDate: NoOfMrk by NoOfindv matrix */
Data: t; /* normalized neighborhood frequency threshold */
Result: N; /* Neighborhood Matrix */
Result: $F \subset M$; /* list of potential markers to be filtered */

$RefMap^{(R)}$ = Map(RHData) ; /* find best map using carthagene */
foreach $i \in I$ **do**
    $J^{(i)} = Resampling(RHData)$; /* Jackknife Re-sampling */
    $Map^{(i)} = Map(J^{(i)})$;
$N(m, m)$ = zeros(m, m); /* initialize */
foreach $Map^{(i)}$ **do**
    for *j=1 to n-1* **do**
        $mrk_1 = Map^{(i)}(j)$;
        $mrk_2 = Map^{(i)}(j+1)$;
        $pos_1 = FindMarkerPosition(mrk_1)$; /* position in RefMap */
        $pos_2 = FindMarkerPosition(mrk_2)$;
        $N(pos_1, pos_2) + +$;
        $N(pos_2, pos_1) + +$;
$N = Normalize(N)$;
foreach $m_i \in M$ **do**
    if $max(N(i, i-1), N(i, i+1)) < t$ **then**
        $m_i \in F$
return N;
return F;

wheat. Based on the genotypic data of 35 markers, 178 RH lines showing maximum marker loss were selected and analyzed using Diversity Array Technology (DArT; Triticarte, Canberra) markers. DArT analysis yielded 641 D-genome specific markers which were then used along with 35 SSR markers (676 in total) to construct radiation hybrid maps for wheat D-genome chromosomes.

3.5.2. Similarity Network Results

Seven data sets were mapped (one data set per chromosome). We refer to markers using pseudonyms because the data and the maps are still in the process of being published. Figure 12 shows part of the similarity network for chromosome 2D visualized using graphviz [18]. At k=5 nearest neighbors and a range of $[i-1, i+1]$ only 3 markers where found unreliable (those with gray filling). Two of those markers were singletons (M32 and M36) that are not linked to any other marker. The third unreliable marker was M25 that is linked only to M30 (which is not in the range to consider it reliable). For the neighborhood matrix comparison approach, the same three markers where defined as unreliable using neighborhood thresholds of 0.9 and 0.95.

3.5.3. Baseline Model Results

We resampled the mapping population of 178 individuals using the jackknife resampling method discussed in Section 3.4.2 resulting in 178 resampled data per chromosome each of 177 individuals plus the main data set of 178 individuals. The total data sets mapped is 179 * 7 = 1253 data sets.

Figure 13 shows the resampling results for chromosome 1D. The X and Y axis represents the reference map and Z axis is the normalized neighborhood frequency. The figure is a visualization of the results, in which the matrix elements, as in Table 6, are represented as z-values in a 3-dimensional plot. For perfect results we would see two peaks of hight 1 immediately adjacent to the diagonal and height 0 elsewhere. The results for this chromosome show few peaks that are far from the diagonal. Using neighborhood threshold of 0.95 only 7 markers out of 59 where defined as unreliable markers.

3.5.4. Comparisons

Sensitivity, Specificity, and Accuracy: We first try to determine if the similarity network algorithm captures the same information as the neighborhood matrix algorithm. For that purpose, we calculate classification-style quality measured and compare them with the

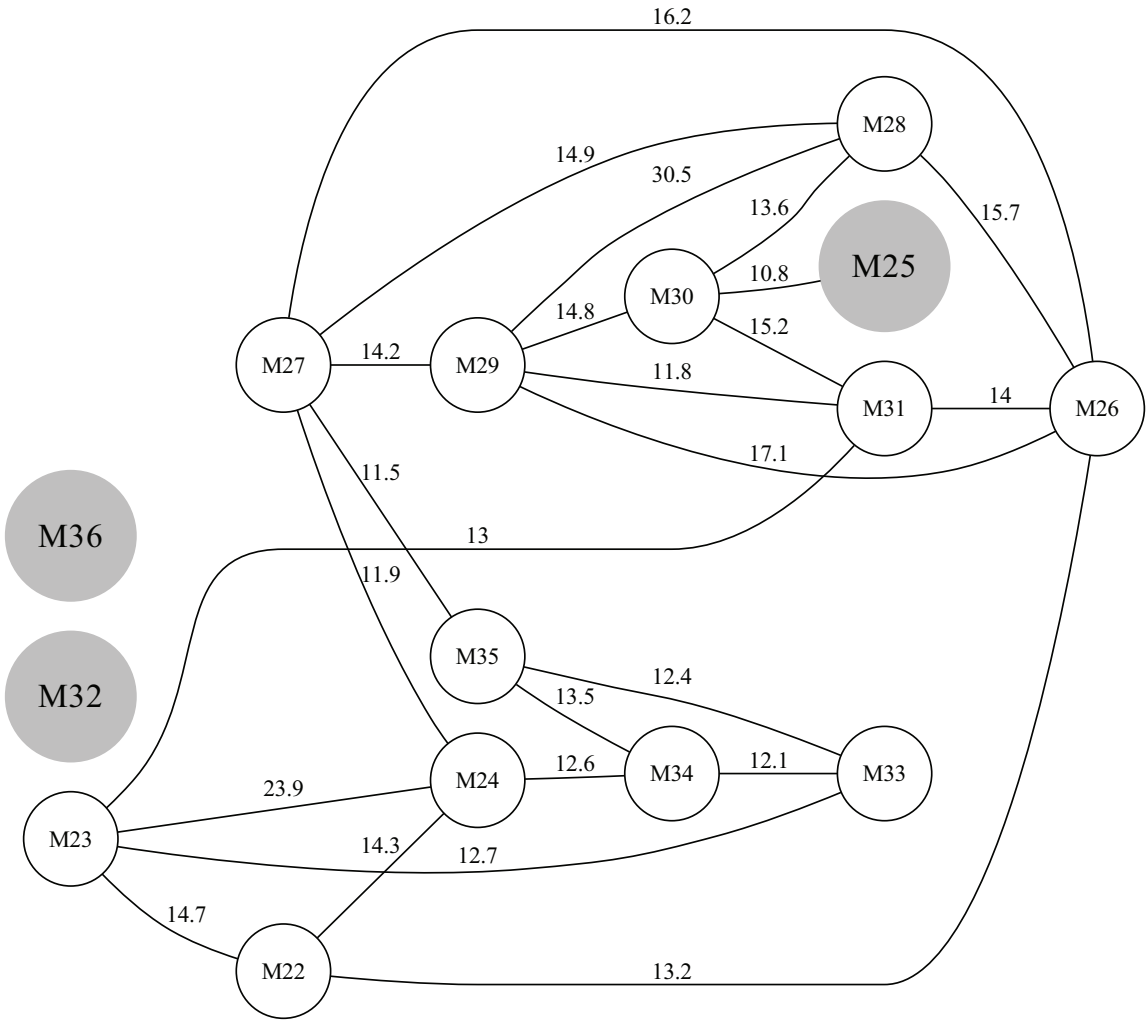

Figure 12: Part of chromosome 2D similarity network. Nodes represent markers labeled according to their position in the reference map. Edges are labeled with LOD scores. Unreliable markers are highlighted in gray.

clustering algorithm provided by Carthagene software. In the software, using specific distance and LOD score, markers can be clustered into several clusters. Any marker that cannot be grouped to any cluster (singleton) is defined as an unreliable marker.

Table 7 shows this comparison. We calculate the sensitivity, specificity, and accuracy for the whole D-genome results for both our algorithm and the comparison approach. The comparison in Table 7 shows that our algorithm outperforms the comparison approach. For all seven chromosomes we achieved better sensitivity and accuracy. The specificity of our

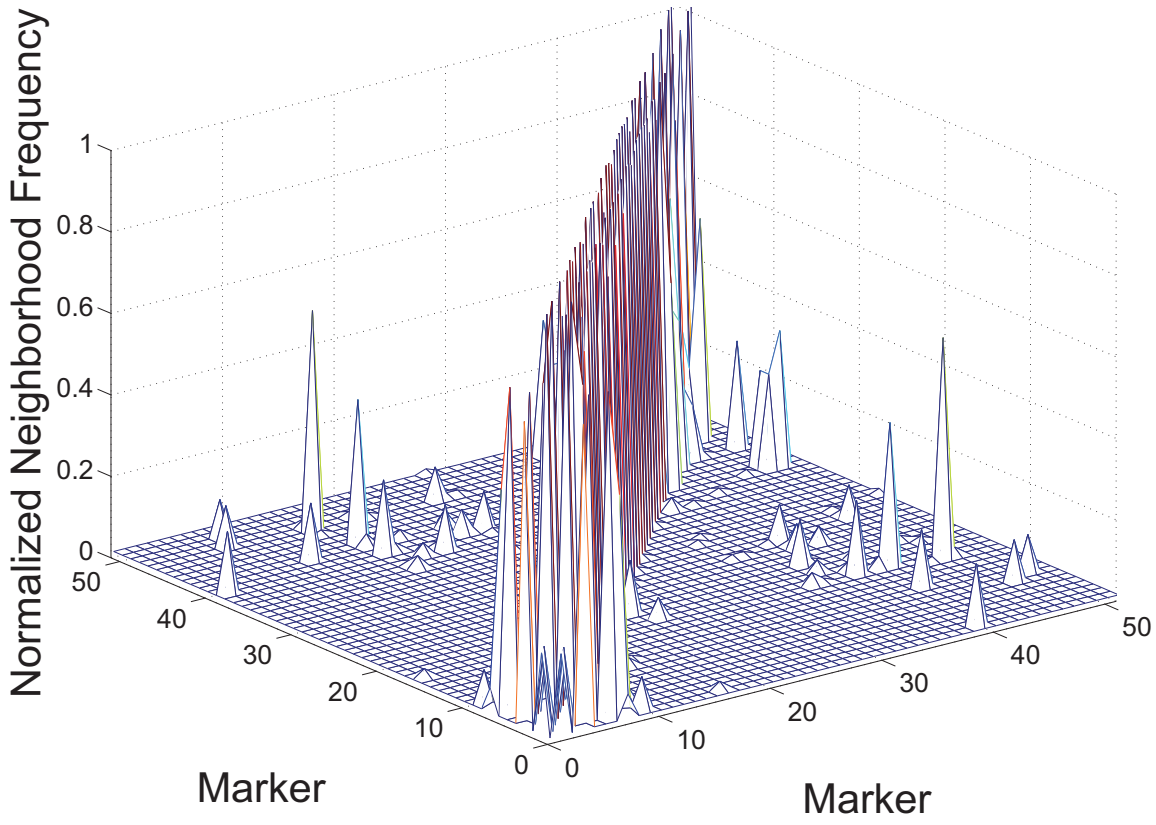

Figure 13: Neighborhood Matrix for the 1D chromosome.

algorithm was also better except for chromosome 1D and 5D where it was close. The three measures for our algorithm and the comparison approach are listed in Table 7.

Table 7: Comparison between the similarity networks filtering algorithm and clustering provided by the Carthagene software.

	similarity network Algorithm			Clustering		
Chr.	Sens.	Spec.	Acc.	Sens.	Spec.	Acc.
1D	0.63	0.95	0.90	0.38	0.97	0.79
2D	1	0.88	0.89	0.60	0.88	0.85
3D	0.80	0.90	0.90	0.50	0.87	0.86
4D	0.71	0.89	0.86	0.63	0.88	0.83
5D	0.56	0.85	0.81	0.50	0.90	0.79
6D	0.43	0.90	0.84	0.25	0.87	0.78
7D	0.82	0.76	0.77	0.75	0.69	0.69

Filtering Percentage: The Filtering percentage can be controlled in the neighborhood matrix algorithm by using neighborhood threshold t. To achieve good results, a value of t close to 1 is recommended. We used various neighborhood thresholds ranging from 0.5 to 0.95. In the same way filtering percentage using the similarity networks can be controlled by both parameters k and r. We used various number of neighbors ranging from 5 to 15 and r range factor from 1 to 6.

Figure 14 shows the filtering percentages for all seven chromosomes using both algorithms. The top part of Figure 14 represents the filtering percentage of the neighborhood matrix algorithm using different neighboring thresholds t. The filtering percentage was in the interval [0, 0.22]. The bottom part shows the filtering percentage using the similarity networks algorithm. We used different parameter settings of both k and r. The filtering percentage was in [0, 0.16]. The figure shows that filtering percentages for most chromosomes were comparable. Using the right parameter settings of k and r we can achieve any filtering percentage that is achieved by using t in the neighborhood matrix algorithm.

Time complexity: The neighborhood matrix algorithm is time consuming. The run time for moderate-size data set does not scale well with both number of markers and individuals. If we assume on average a data set contains n markers tested on 100 individuals, using the neighborhood matrix jackknife resampling-based algorithm means mapping 100 different data sets, while in our algorithm, we only need to map one data set. For this example the run time can be decreased by 2 orders of magnitude. Even if we decide to re-sample using 90% of the individuals at a time, our algorithm will be faster by one order of magnitude.

The testing data set for the wheat D-genome is for seven chromosomes with 178 individuals. Using the neighborhood matrix algorithm, 1253 data sets where mapped in total (for both the original 7 data sets and the re-sampled data). While using our similarity networks algorithm only the 7 original data sets need to be mapped. Our algorithm

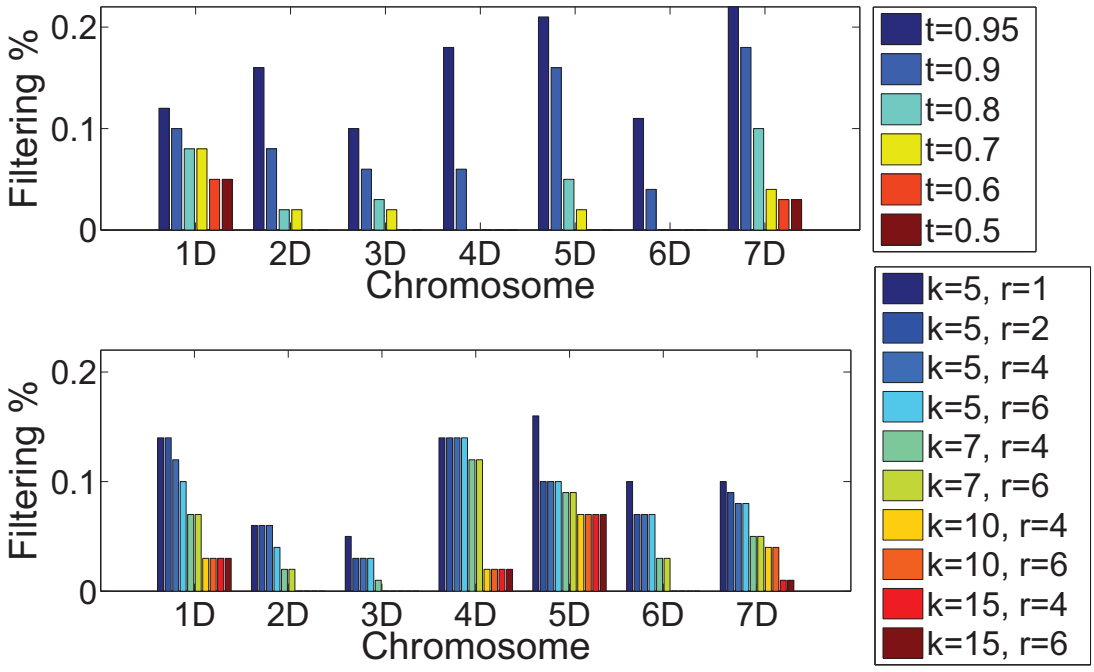

Figure 14: Comparison of filtering percentage for wheat D-genome using neighborhood matrix and similarity network algorithms. Different chromosomes are shown along the X axis. Top: neighborhood matrix filtering. Bottom: similarity network filtering. Different parameter settings of t, k and r are used.

decreased the run time by more than 2 orders of magnitude, and removed the need for high-performance computing equipment.

3.6. Conclusions

In this chapter, we presented an algorithm for identifying unreliable markers for radiation hybrid mapping. Our algorithm is based on building similarity networks and testing for connectivity relationships between markers based on their LOD scores. We consider the map built using all individual information as a reference map and checked for unreliable markers in the network. As a baseline model, we used a resampling-based algorithm that builds a neighborhood matrix summarizing the neighborhood relationships between markers for maps built on the different resampled data. We showed that our algorithm can capture this information much faster than the baseline model, decreasing the run time by more than two orders of magnitude. We tested our algorithm on a data set generated using

radiation hybrids developed for mapping of wheat D-genome chromosomes. For the seven wheat D chromosomes we confirmed unreliable markers and showed that our algorithm outperforms a clustering based algorithm provided by the mapping software.

CHAPTER 4. SCALING UP THE EVALUATION OF MARKER RELIABILITY FOR GENERATING ACCURATE FRAMEWORK MAPS TO LARGE GENOMES

In Chapter 3, the similarity network algorithm is discussed for filtering unreliable markers. In Chapter 4, an enhanced network-based approach that uses concepts from the previous algorithm is presented. The presented algorithm is a fast way for detecting unreliable markers and building solid framework maps.

4.1. Abstract

Background: Genome mapping is an important methodology to assist in the sequence assembly of large and complex genomes, especially when repetitive sequences are prevalent. The mapping process is affected by mis-scoring and missing data, resulting in potentially unstable maps. Increasing map size increases the chances for incorrect ordering. This problem can be alleviated by first creating a framework map of markers that are particularly stable. An algorithm is presented for eliminating loose markers that result in unstable maps and building framework maps in radiation hybrid mapping. Conventional approaches for discovering those loose markers depends mainly on resampling from the mapping population and mapping all the resampled data. By considering the mapping distribution, loose markers are filtered iteratively one marker in each iteration. Those techniques do not scale well to large genome sizes.

Results: In this chapter, we provide an alternative approach for discovering unreliable loose markers. We build a framework map by constructing networks from the resampled data. The support for each edge of the network is derived using all individual information. Loose markers are filtered based on network linkage. Our approach is computationally fast since building those networks does not depend on actual mapping results. We show that the framework maps created using our approach align very well with the framework maps

created using standard computationally expensive algorithms. In addition, the size of our framework maps are comparable with the size of framework maps created using standard approaches. We compare with other framework mapping techniques based on filtering out singletons from clustering results and show that those techniques are not suitable. Filtering out singletons from clustering does not match either our approach or standard conventional approaches for detecting unreliable markers and building framework maps in terms of map size and marker alignment on the framework. Evaluation is carried out on wheat 1D and 2D chromosomes from data generated in the laboratory from radiation hybrid technique.

Conclusions: We present a fast way of building framework maps in radiation hybrid mapping by filtering out loose markers from networks created using a pairwise similarity measure. The algorithm scales well with both number of markers and number of individuals. While our algorithm decreases the computation time dramatically, the results are comparable with more computationally expensive standard approaches in terms of marker alignment and framework physical map size. The results are clearly superior to a comparison algorithm based on clustering.

4.2. Background

Genome mapping, or the problem of the assignment of DNA sequences to chromosomes, has been widely studied for humans, animals, and most recently for plants. High-throughput genotyping platforms [2, 7, 61, 45] has a dramatic impact in increasing the pace of genome mapping, resulting in a large amount of new data. These high-throughput genotyping technologies helped in the development of high density marker scaffolds that can be used for genome assembly. Complex genomes such as wheat, with a genome of 17 Gb (approximately five times larger than human genome), for which $\geq$80% of the genome consists of repetitive sequences [36], require such scaffolds for assembly. Under such a scenario, Bacterial Artificial Chromosome (BAC) contigs of limited length are created, and a high quality high density map is used to align the contigs. The high density molecular

map can also provide the information to help fill gaps between contigs [46].

We are in the process of genotyping thousands of markers and creating high resolution radiation hybrid maps for wheat D-genome. These radiation hybrid maps would help in creating such a marker scaffold for sequence assembly of wheat. Radiation hybrid mapping provides both the resolution required to map a large number of markers, while minimizing the population size [11]. When using high-throughput genotyping platforms, a small degree of mis-scoring is expected. In addition, the amplification result of some markers may be ambiguous, and instead of scoring data points as 0 or 1, some are scored as missing data. In most cases, it is not economically feasible to validate the genotyping results by repeating the experiments. Under such a scenario, it is beneficial to identify the most reliable markers for building a framework map that can then be used for creating an accurate marker scaffold or molecular map of the chromosome. For that purpose, detecting DNA markers that contribute unstable, poor quality maps is a key task. Conventional techniques are based on using resampling analysis [39, 40, 41, 51] and filtering the loosest marker each time iteratively. These approaches evaluate the relevance and reliability of markers by mapping all resampled data and creating a histogram of the neighborhood relationships of the markers based on a reference map [39, 40, 41, 51]. These techniques are time consuming and are not practical for large or even moderate size data sets. The mapping problem scales exponentially with the number of markers to be mapped. This problem is aggravated by the need to run compute maps for each sample for a single neighborhood matrix algorithm [39, 40, 41, 51] and repeating this process for each filtered marker. Figure 15 shows the process of iterative filtering/resampling using the standard conventional approach for the wheat chromosome 2D. For this chromosome, 28 iterations were needed for algorithm convergence.

Mapping software [12, 52] provide options for fast alternatives to fast framework map building. However, those techniques do not measure map stability based on mapping

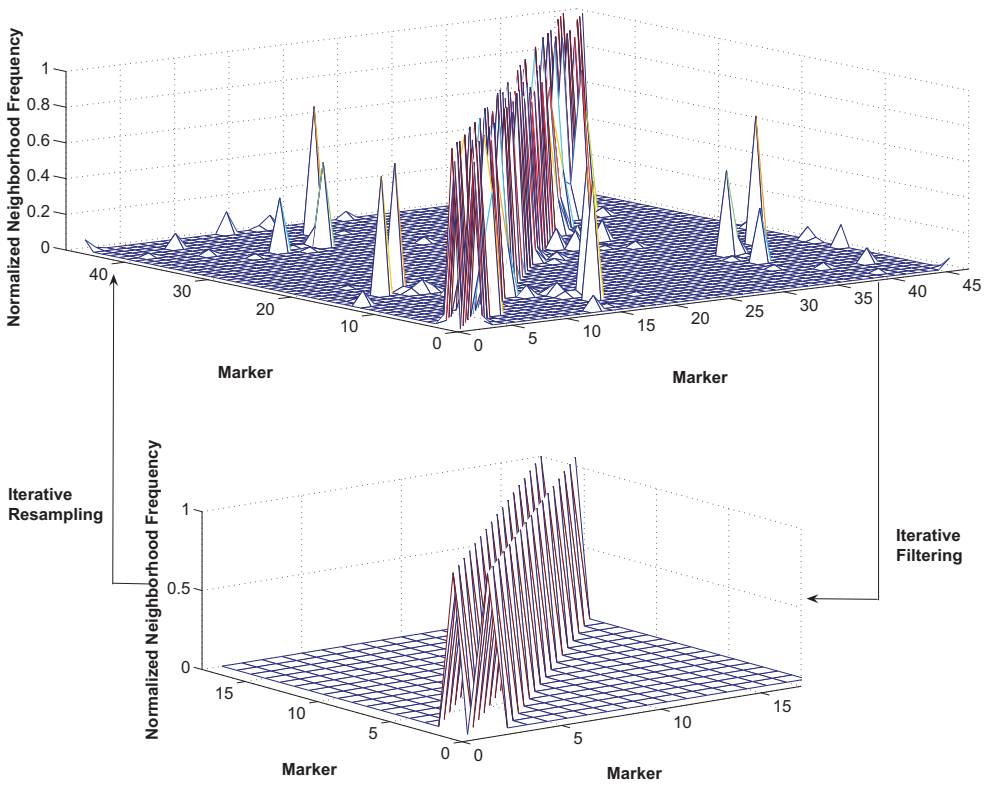

Figure 15: The process of iterative filtering of unreliable markers using neighborhood matrix algorithm for wheat chromosome 2D. X and Y axis represent the marker index in the reference map. Z axis is the normalized neighborhood frequency. Top: neighborhood matrix of all maps created using jackknife resampled data for the first iteration. The top sub figure shows that the mapping results have noise. Bottom: the neighborhood matrix for last iteration. The resampling results shows a stable map. In each iteration exactly one marker with the lowest neighborhood point is filtered out. The algorithm converges when every marker has a neighborhood value that exceeds or equals specific threshold (100% on this example).

results nor based on linkage results. The framework map generation provided by the mapping software [12] uses an incremental insertion procedure. It is recommended to use an LOD score of at least 3 for building a solid framework map. However, when using such a threshold, very large number of markers are discarded. In some cases, approximately 5% of markers remains as the framework map. Those techniques are considered to be too simple and might not be suitable for noisy data sets. After a framework map has been built, it is used as scaffold to merge the less stable markers generating and creating a full and

high quality molecular map. This work proposes an algorithm that is computationally less expensive than conventional approaches but results in maps of comparable quality.

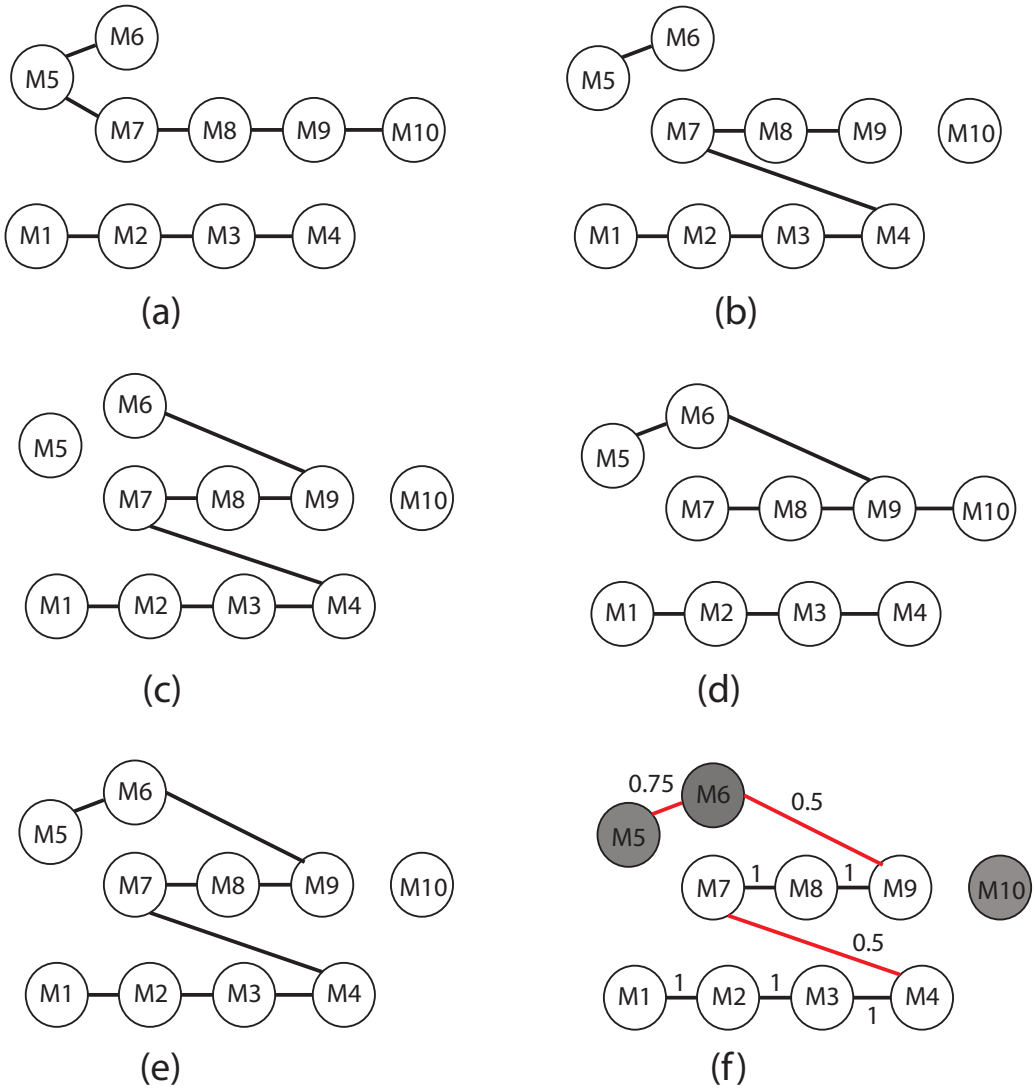

Figure 16: Schematic represents the process of creating support networks. Artificial data consists of 10 markers on 4 individuals. Nodes represents marker index in best map created using all individual information $J^{(0)}$. An edge is created between two markers only if they are mutual neighbors based on fixed number of neighbors K. Sub-figures (a) to (d) are networks created using resampled data $J^{(l)}$. Sub-figure (e) is the network created using all individual information $J^{(0)}$. Sub-figure (f) is the final network after calculating support for each edge. Edges are labeled with their support calculated from sub-figures (a) to (d). Edges in red are broken. Markers in gray fillings are defined as unreliable.

In this chapter, we propose a fast algorithm for creating solid framework maps by filtering out groups of markers iteratively from the network created based on a 2-point similarity measure. This process is fast because it does not depend on mapping the resampled data. We used the concept of the similarity network first introduced in [4]. We propose an algorithm that iteratively identifies unreliable markers based on marker linkage of all networks created on resampled data. The LOD (logarithm of odds-base10) score, which is a measure of the odds that two markers are indeed linked to the odds that the appearance of linkage is caused by random chance alone, is used to build these networks. Figure 16 shows a toy example of artificial data of 10 markers on 4 individuals. We first resample the data set using the jackknife resampling method [23, 10], in which exactly one individual is left out each time. For each resampled data set, we create the similarity network in the same way as described in [4] by only linking markers that are mutually neighbors to each other based on a fixed number of neighbors k. Those networks are shown in panels (a) to (d) of Figure 16. In the same way, we create the network for the full data that contains information on all individuals ($J^{(0)}$). This network is shown in panel (e). After creating these networks, we calculate the support for each edge in the network created using all individual information by finding its proportion on all resampled similarity networks. The support network for Figure 16 is shown in panel (f). Edges on the support network are removed using a predefined support threshold. In this example, edges marked in red in panel (f) are broken using support threshold $s \geq 0.80$. Markers that fail to meet a specific linkage range r can be defined as unreliable. In this example, if $r = 1$, then the set F of unreliable markers is : $F = \{M5, M6, M10\}$. This process can be carried out iteratively until no other marker can be added to the set F.

Once a solid framework map is found, other filtered markers can be inserted to the best position in this framework by enforcing the order of framework markers and inserting other markers to the best possible position.

4.3. Methods

4.3.1. Support Network Construction

The RH data is a binary matrix D of n markers and v individuals, where each entry is a binary value indicating the presence or absence of a marker in an individual:

$$D = \begin{pmatrix} d_{1,1} & d_{1,2} & \cdots & d_{1,v} \\ d_{2,1} & d_{2,2} & \cdots & d_{2,v} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n,1} & d_{n,2} & \cdots & d_{n,v} \end{pmatrix}$$

where,

$$d_{i,j} = \begin{cases} 1, & \text{if marker } i \text{ is present in individual } j \\ 0, & \text{if marker } i \text{ is absent in individual } j \\ -, & \text{missing information} \end{cases}$$

We formulate the problem of detecting the unreliable markers as an undirected graph of M markers and E edges, where each edge is labeled with its support. At first, we find the best map using the Carthagene software [12] by mapping the data containing the information on all individuals $J^{(0)}$. The markers then, are labeled sequentially according to their position in the best map. A marker with position i in the map is labeled as M_i.

The RH data is then resampled using the jackknife resampling method by removing one individual at a time. $J^{(l)}$ is the resampled data set that contains all individual information except for individual l. Notice that by using this resampling method, we will have exactly v resampled data sets, where v is the number of individuals.

Definition 5 (Similarity network)**.** *Let $G^{(l)} = (M, E^{(l)})$ be the undirected graph constructed using the resampled data l. $M = \{M_1, M_2, \ldots, M_n\}$ is the list of markers labeled sequentially according to their position on the map using data from $J^{(0)}$. $(M_i, M_j) \in E^{(l)}$ only if M_i and M_j are mutually connected to each other based on their k nearest neighbors*

with the highest LOD scores:

$$\forall_{l=0}^{v}\forall_{i=1}^{n-1}\forall_{j=i+1}^{n}(M_i, M_j) \in E^{(l)} \quad \text{if } M_i \in KNN^{(l)}(M_j, k) \bigwedge M_j \in KNN^{(l)}(M_i, k) \tag{12}$$

Definition 6 (Support network)**.** *$G^{(0)} = (M, E^{(0)})$ is the support network created using Definition* 5 *using all individual information. For each edge $(M_i, M_j) \in E^{(0)}$ we calculate the edge support by finding the proportion of how many times (M_i, M_j) occurs on all graphs built on resampled data:*

$$\forall (M_i, M_j) \in E^{(0)}, \ \text{Supp}\ (M_i, M_j) = \frac{\sum_{l=1}^{v} f(l)}{v}$$
$$\text{Where, } f(l) = \begin{cases} 1, & \text{if } (M_i, M_j) \in E^{(l)} \\ 0, & otherwise \end{cases} \tag{13}$$

The details of the support network construction can be seen in Algorithm 4. The input is the RH binary data matrix of n markers and v individuals, where entries are binary values indicating the presence or absence of the marker in an individual. The output is an undirected graph, where nodes are marker labels and edges are labeled with their support. In line 1, the full data set is mapped using the Carthagene software [12]. Jackknife resampling is carried out in line 3. In lines 4 and 5, we label markers for all the resampled data according to the marker positions in the best map found in line 1. In lines 6 to 9, we find the $k-$nearest neighbors for each marker in each resampled data. In lines 10 through 13, we create an undirected graph for each resampled data $J^{(l)}$. Finally, we calculate the support for each edge on the graph created using all individual information $(G^{(0)})$ in lines 14 and 15.

Algorithm 4: Support Network Construction.

Data: D, n, v; /* D: RH data matrix of n markers and v individuals */

Result: $G^{(0)} = (M, E^{(0)})$; /* graph of connected markers. Edges are labeled with support */

```
J^(0) = Map(D) ;                    /* find best map using carthagene (reference map) */
for i=0 to v do
    J^(i) = Resampling(D) ;         /* Jackknife Re-sampling */
    foreach m ∈ M do
        Label(m) = M&pos(m) ;       /* label markers according to position */
    KNN^(i)(n, k) = zeros(n, k);    /* initialize */
    foreach m_p ∈ M do
        for j=1 to k do
            KNN^(i)(p, k)=KNN(m_p);
    for j=1 to n-1 do
        for l=j+1 to n do
            if m_j ∈ KNN^(i)(l, k) AND m_l ∈ KNN^(i)(j, k) then
                (m_j, m_l) ∈ E^(i) ;   /* determine edges in graph */
foreach E^(0) ∈ G^(0) do
    Support(E^(0));
return G^(0);
```

4.3.2. Edge Breaking

After calculating the support for each edge in $G^{(0)}$, all edges that fail to meet specific predefined support threshold are broken. The process of breaking edges from the graph decreases the number of cycles and increases number of markers filtered out by removing connections. Edge breaking is performed according to the formula below:

$$\forall (M_i, M_j) \in E^{(0)}, (M_i, M_j) \notin E^{(0)} \quad (14)$$

$$\text{if Supp}(M_i, M_j) \leq s$$

4.3.3. Marker Filtering

We filter unreliable markers from $G^{(0)}$ based on the same criteria proposed in [4]. Makers that fail to meet a specific linkage range r, i.e., which do not have an edge with respect to a node with an index that differs by no more than r, are added to the set F of unreliable markers:

$$\forall_{i=1}^{n} M_i : M_i \in F \quad \text{(15)}$$
$$\text{if } \nexists_{j=i-r}^{i+r} (M_i, M_j) \in E^{(0)}$$

Algorithm 5: Marker Filtering from Support Network.

Data: $G^{(0)} = (M, E^{(0)})$; /* graph created using all individual information */
Data: r, s; /* r:range threshold, s:support threshold */
Result: $F \subset M$; /* list of potential markers to be filtered */

1 **foreach** $(m_i, m_j) \in E^{(0)}$ **do**
2 **if** $Supp((m_i, m_j)) < s$ **then**
3 $Remove((m_i, m_j))$; /* edge breaking */
4 **foreach** $m_i \in M$ **do**
5 $Flag = False$;
6 **foreach** $(m_i, m_j) \in E^{(0)}$ **do**
7 **if** $m_j \in [m_{i-r}, m_{i+r}]$ **then**
8 $Flag = True$;
9 $exitFor$;
10 **if** $Flag = False$ **then**
11 $m_i \in F$;
12 **return** F;

The details of edge breaking and identifying unreliable markers can be seen in Algorithm 5. The inputs are the undirected graph $G^{(0)}$ created using all individual information $J^{(0)}$ and the two parameters r and s indicating range threshold and support threshold respectively. The output is the set F of unreliable markers. In lines 1 to 3, we break

all edges that fail to meet a specific support threshold s. In lines 4 to 11 we find the set F. Each marker that fails to meet a linkage range r is defined as unreliable marker.

Notice that we carry out the process of support network construction and marker filtering iteratively until no other marker can be filtered out.

4.3.4. Baseline Model - Mapped Neighborhood Matrix

Since there is no ground truth available to evaluate our result, we assume that the markers filtered using the neighborhood matrix [39, 40, 41, 51] are the true answer. To evaluate our approach we implemented the neighborhood matrix approach by iteratively filtering exactly one marker each time. The criterion for filtering is that the marker has the lowest neighborhood value from the neighborhood matrix created by mapping all the resampled data. The remaining data are resampled again and the filtering process is carried out until all markers have a neighborhood value of at least 99% with regard to their immediate neighbor. The pseudocode for the baseline model can be seen in Algorithm 6.

4.4. Results and Discussion

4.4.1. Data Sets

The evaluation is done on two data sets from radiation hybrids and genetic mapping data. The first data set is from the radiation hybrids of the wheat 1D and 2D chromosomes. This data set was generated in the laboratory on a mapping population of 1542 radiation hybrid individuals. Individuals showing maximum marker loss where selected resulting in a mapping population of 178 individuals. The number of markers for chromosomes 1D and 2D analyzed using Diversity Array Technology (DArT) [1] were 59 and 51 markers respectively.

The second evaluation we carried out is on a genetic mapping data set of wheat chromosome 1B (unpublished data). The data set represents the doubled haploid mapping population developed from a cross of two durum wheat cultivars Rugby [48] and Maier [16]. The mapping population consists of 105 individuals on 36 markers.

Algorithm 6: Iterative Filtering of unreliable markers from Neighborhood Matrix.

```
Data: D, n, v;  /* D: RH data matrix of n markers and v individuals */
Data: t;  /* normalized neighborhood frequency threshold */
Result: RefMap^(R);  /* Framework map after filtering the unreliable
      markers */
Result: F ⊂ M;  /* list of potential markers to be filtered */
m = n;
F = {};
MapStable = False;
while MapStable = False do
    RefMap^(R) = Map(D) ;  /* find best map using carthagene */
    foreach i ∈ v do
        J^(i) = Resampling(D) ;  /* Jackknife Re-sampling */
        Map^(i) = Map(J^(i));
    N(m, m) = zeros(m, m);  /* initialize */
    foreach Map^(i) do
        for j=1 to m-1 do
            mrk_1 = Map^(i)(j);
            mrk_2 = Map^(i)(j + 1);
            pos_1 = FindMarkerPosition(mrk_1) ;  /* position in RefMap */
            pos_2 = FindMarkerPosition(mrk_2);
            N(pos_1, pos_2) + +;
            N(pos_2, pos_1) + +;
    N = Normalize(N);
    min = 0;
    mrk = "";
    foreach m_i ∈ M do
        if N(i, i + 1) < min then
            mrk = m_i;
            index = i;
    if min ≥ t then
        MapStable = True;
    else
        m = n − 1;
        F = F ∪ mrk;
        D(index, :) = [ ];
return RefMap^(R);
return F;
```

4.4.2. Radiation Hybrids Results

Marker Alignment on Mapping Results: In order to test the stability of the mapping results, we created several maps for the same chromosomes based on the filtering strategies

discussed in Section 4.3. Figure 17 shows the mapping results for wheat chromosome 1D visualized using [14]. The middle part of Figure 17 represents the mapping result for all markers without any filtering. The left part of the figure is the mapping result using the baseline approach for filtering of markers. Using this approach one marker is filtered out in each iteration, as discussed in Section 4.3.4. The right part of Figure 17 represents the mapping result when using our algorithm for filtering the unreliable markers as discussed in Algorithm 5. As can be seen from the figure, those three maps are not consistent. The original map (middle part) is not consistent with either framework maps. Creating such a stability requires detecting and eliminating unreliable markers.

Similar inconsistencies can be seen in Figure 18. This figure represents the mapping results for wheat chromosome 2D. We carried out the same test of how markers align if all markers included (middle part of Figure 18) with our algorithm (left part of the figure) and with the baseline algorithm (right part). Once again, those three maps where found inconsistent.

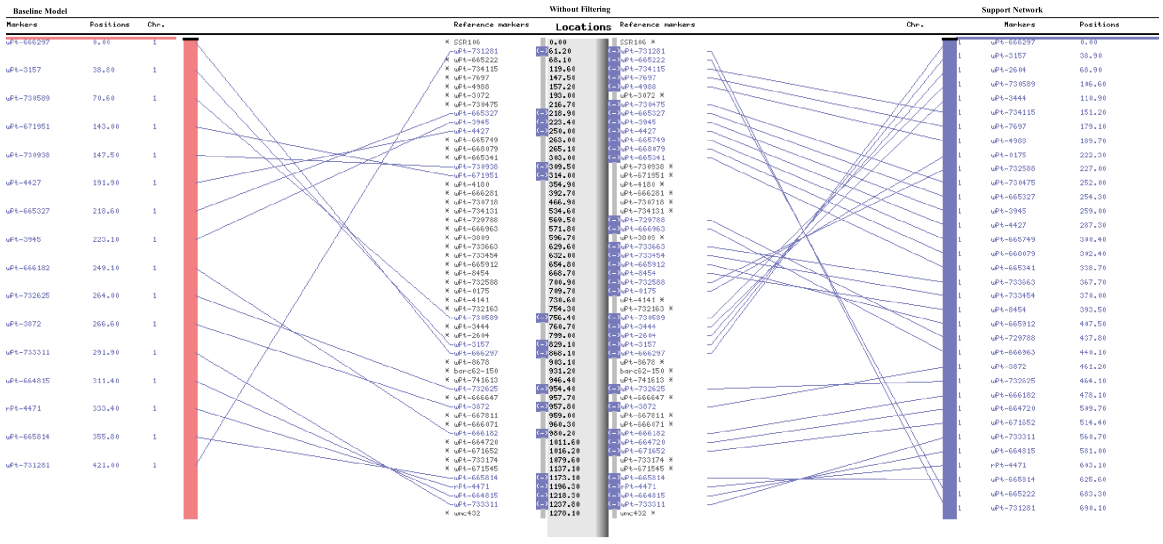

Figure 17: Comparison between three maps created for the same 1D chromosome of wheat. Left: map created after removing unreliable markers detected using the neighborhood matrix approach. Middle: map created using all markers without any filtering. Right: map created after removing unreliable markers detected using support network algorithm.

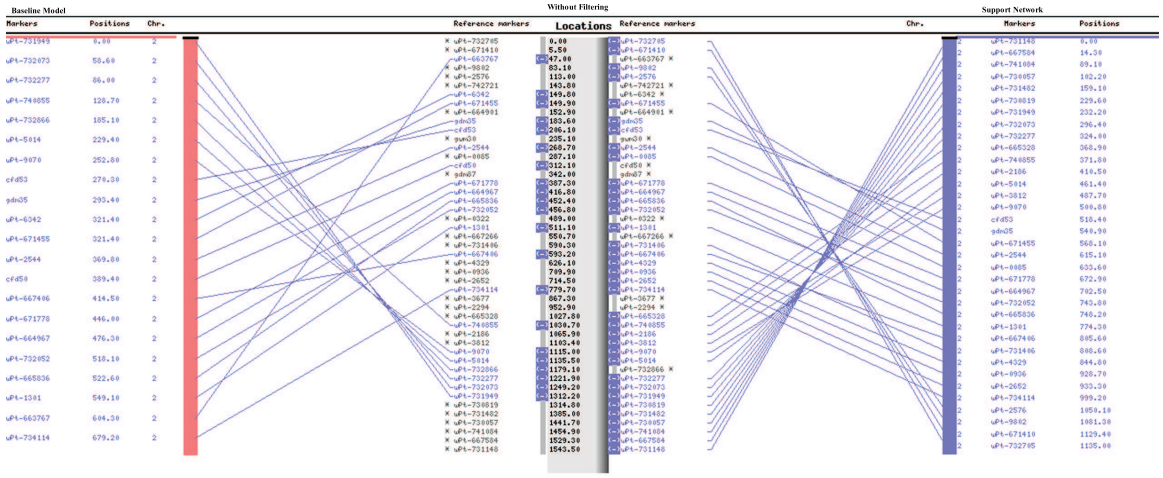

Figure 18: Comparison between three maps created for the same 2D chromosome of wheat. Left: map created after removing unreliable markers detected using the neighborhood matrix approach. Middle: map created using all markers without any filtering. Right: map created after removing unreliable markers detected using support network algorithm.

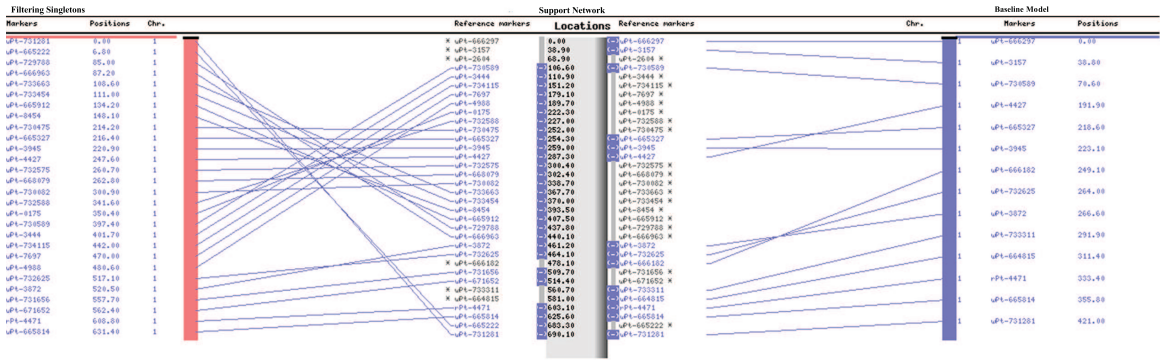

Figure 19: Comparison between three approaches for finding solid framework maps by filtering out unreliable markers created for the same 1D chromosome of wheat. Middle: the neighborhood matrix used as a baseline approach (framework map created after filtering out loose markers iteratively based on their neighborhood values). Left: framework map created after removing singleton markers detected using the clustering algorithm provided by Carthagene software. Right: framework map created after removing unreliable markers detected using support network algorithm.

We compared the framework maps created after filtering the unreliable markers using our support network algorithm with the computational expensive baseline algorithm (neighborhood matrix algorithm proposed in [39, 40, 41, 51]) and with filtering singletons from standard clustering algorithm used in Carthagene software [12]. Figure 19 shows this

comparison for wheat chromosome 1D visualized using [14]. The middle part of Figure 19 is used as a reference (framework map created after filtering the unreliable markers using the support network algorithm). The right part of Figure 19 corresponds to the framework map created after filtering unreliable markers using the baseline model of neighborhood matrix. As can be seen from the figure, the layout of most markers in our framework map align very well with the computational expensive baseline model. The first and the last markers in the two frameworks are the same. The order of other markers matches the baseline framework map order with the exception of some local flipping. Note that the baseline framework map does not align well with the map created by filtering out only singletons from the clustering algorithm provided Carthagene software [12]. This comparison can be seen in Figure 19. The left part of the figure correspond to the map after filtering out singletons. As can be seen from the figure the location of most markers are inconsistent with the baseline model along the whole chromosome.

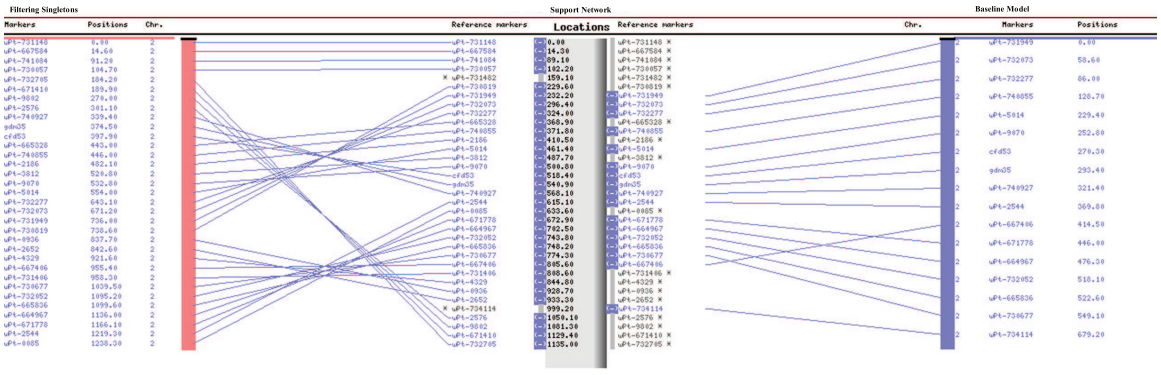

Figure 20: Comparison between three approaches for finding solid framework maps by filtering out unreliable markers created for the same 2D chromosome of wheat. Middle: the neighborhood matrix used as a baseline approach (framework map created after filtering out loose markers iteratively based on their neighborhood values). Left: framework map created after removing singleton markers detected using the clustering algorithm provided by Carthagene software. Right: framework map created after removing unreliable markers detected using support network algorithm.

We conducted the same alignment test of the three framework maps on wheat chromosome 2D. Figure 20 shows this comparison. The framework map created using the

baseline model (right part of Figure 20) is consistent with the framework map created using our proposed support network algorithm (middle part of Figure 20). The locations for all markers align very well in both framework maps with the exception of only one marker ($wPt-667406$). However, the inconsistency between the baseline framework map and the framework map created after filtering singletons (left part of Figure 20) is clear. It would be insufficient to depend only on filtering out singletons to remove noise from RH mapping data.

Map Cumulative Distance: The second criteria we considered for our algorithm evaluation is the cumulative physical map distance of the created framework maps. Table 8 shows this comparison. In general, the cumulative distance of the framework maps created using our proposed support network algorithm are comparable with the framework maps created using the baseline method. The cumulative distance for the framework maps of wheat chromosome 1D for the baseline method and the support network algorithm were 421 cR and 690.10 cR respectively (see Figure 19). One could question why the cumulative distance differ, considering that the first and the last markers on the two framework maps are the same. This can be explained considering that only 18 markers were mapped using the baseline model while 34 markers were mapped using the support network algorithm. It is well known that the measure cR (centi-Ray) depends on the number of breaks between markers. Mapping more markers means creating more breaks which results in a larger physical distance.

To make a fair comparison of the cumulative map distance between our algorithm and the baseline model, we fixed the first and the last markers of the framework map and resampled the data set by randomly choosing exactly 16 markers each time. The resampled data was mapped and the cumulative distance was recorded for all 30 resampled data sets. We averaged over all 30 cumulative distances and took the average cumulative distance of 485.92 cR with standard deviation of 37.45 cR. Mapping an exactly equal number of

markers as in the baseline model resulted in approximately the same cumulative distance.

The results for wheat chromosome 2D confirm that we can achieve approximately the same physical map distance using the support network algorithm achieved by the more computationally expensive approach. Figure 20 shows the cumulative distance for the framework map created using the baseline model (right part of the figure) and the cumulative distance of the framework map created using the support network algorithm (middle part of the figure). Mapping 25 markers using the baseline model resulted in a cumulative distance of 679.24 cR while mapping 35 markers resulted in a cumulative distance of 1135.00 cR. However, considering only the portion of the map that aligns with the baseline model framework map (the range [232.20 cR, 999.20 cR]) results in a cumulative distance of 767.00 cR mapping 25 markers. We used the same resampling strategy for wheat chromosome 2D. Fixing the first and the last markers of the framework and resampling from the remaining markers by randomly choosing 23 markers each time resulted in an average cumulative distance of 714.81 cR with 30.36 cR standard deviation. The results of the proposed support network are better than the results of our previous similarity network algorithm [4] for wheat chromosome 2D in terms of map cumulative distance when resampling equal size of markers. In addition, the alignment between support network maps and the baseline model is much better than the alignment between the similarity network [4] maps and the baseline model. Table 8 summarizes this section.

Computation Time and Overlap Percentage: We compared the computation time required for creating the framework maps by filtering out the unreliable markers for the three different approaches. The baseline model is computationally expensive since it requires mapping all the resampled data in each iteration. In addition, it requires many iterations to converge because it filters out exactly one marker in each iteration. To handle this computation intensive problem, we used the Linux version of Carthagene [12] for the actual mapping and installed it on cluster of 120 high performance machines. In contrast,

our proposed support network algorithm is computationally fast since we only resample the data to calculate the LOD scores needed to build the networks. Fewer iterations are required to converge since there is no restriction on the number of markers filtered in each iteration. Filtering out singletons from clustering is the fastest approach, however, we showed in Section 4.4.2 that this approach is not sufficient for filtering out unreliable markers.

The run time required for the baseline approach is ($R_{(b)} = i_{(b)} * t_{(b)} * (v + 1)$) where, $i_{(b)}$ is the number of iterations required to converge, $t_{(b)}$ is the run time for a single data set, and v is the number of individuals. On the other hand, the run time for the proposed support network algorithm is ($R_{(c)} = i_{(c)} * t_{(c)}$) where $i(c)$ is always much smaller than $i_{(b)}$ because we do not have a restriction on the number of markers filtered in each iteration and $t_{(c)}$ is always less than $t_{(b)}$ because we map less markers in each iteration. In the worst case scenario, if $i_{(b)}$=$i_{(c)}$ and $t_{(b)}$=$t_{(c)}$, we are eliminating mapping the resampled data sets. Table 9 shows a comparison of the number of iterations required to converge for the two algorithms for wheat chromosomes 1D and 2D. Using the baseline approach, the number of data sets mapped for wheat chromosomes 1D and 2D were 7697 and 5012 respectively, while using our algorithm, only 5 and 3 data sets were mapped. For this data, the run time was decreased by more than 3 orders of magnitude using our algorithm.

Table 9 shows the number of markers filtered using the three different algorithms and the number of iterations required for algorithm convergence. Notice that we only needed five and three iterations for our algorithm to converge for wheat chromosomes 1D and 2D respectively, while the baseline model algorithm needed 43 and 28 iterations respectively.

We also achieved a good overlap percentage of the filtered markers between our algorithm and the baseline algorithm. As can be seen in Table 9 the overlap percentage was 86% and 75% for chromosomes 1D and 2D respectively.

Table 8: Comparison of map cumulative distance between the baseline model and support network algorithm.

	(Cum Dist / No. of Mrk)			((Avg. / STDV) r=30)	
Chr.	Baseline Model [40]	Similarity Network [4]	Support network	Similarity Network [4] with resampling	Support Network with resampling
Chr. 1D	421.00 cR / 18 Mrk	891.80 cR / 51 Mrk	690.10 cR / 34 Mrk	464.94 cR / 51.09 cR	485.92 cR / 37.45 cR
Chr. 2D	679.24 cR / 25 Mrk	1406.40 cR / 47 Mrk	767.00 cR / 25 Mrk	944.09 cR / 62.63 cR	714.81 cR / 30.36 cR

Table 9: Comparison between different approaches for finding framework maps. The neighborhood matrix approach is used as a baseline model. The table shows the number of loose markers detected using each approach, the number of iterations required to converge, and the overlap percentage between the baseline model, the support network algorithm, and clustering algorithm respectively.

		No. of markers Filtered / No. iterations			Overlap Percentage	
Chr.	No. Markers	a) Neighborhood Matrix [40]	b) Supp. Network	c) Singletons	a) and b)	a) and c)
Chr. 1D	59	43 / 43	22 / 5	16 / 1	86%	75%
Chr. 2D	51	28 / 28	12 / 3	5 / 1	75%	80%

4.4.3. Genetic Mapping Results

Similar results have been achieved for wheat 1B chromosome genetic data. Figure 21 shows a comparison between the map created using map maker software [35] (left part) and the map created using our support network algorithm (right part). As can be seen from the figure, the two maps align very well. All markers are in positions with the exception of one local flip (markers wPt-0420 and wPt-1684). The map cumulative distance for the genetic map created using map maker software [35] was 110.5 cM while using our support network algorithm we achieved a map cumulative distance of 77.4 cM. Only two iterations where needed for algorithm convergence using our support network algorithm.

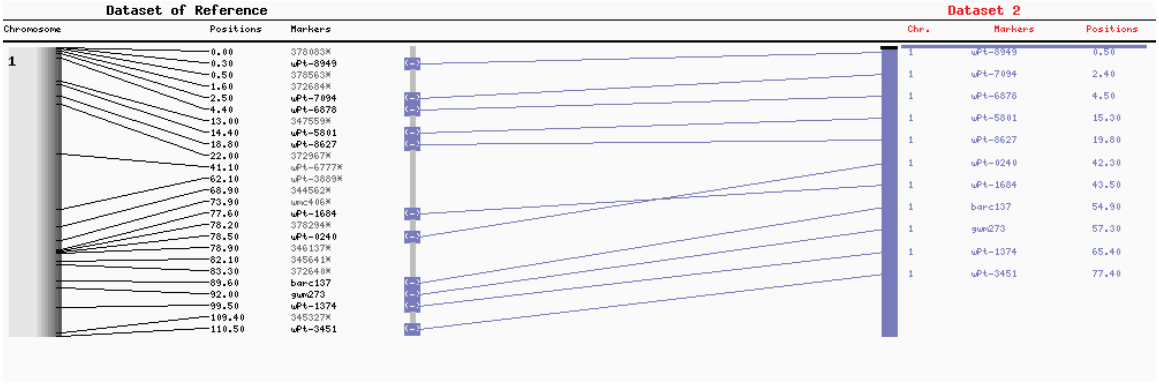

Figure 21: Comparison between two genetic maps of Wheat chromosome 1B. Left: Wheat chromosome 1B genetic map created using map maker software [35]. Right: chromosome 1B genetic map created using support network algorithm.

4.5. Conclusions

In this chapter, we proposed a fast algorithm for building solid framework maps by filtering out loose unreliable markers from support networks. Our algorithm depends on creating a support network of marker linkage by aggregating the information of all similarity networks built on resampled data. These similarity networks are built using 2-point LOD scores. We create a solid framework map by iteratively filtering out unreliable markers detected by the support network algorithm. Building support networks is computationally fast since creating these networks does not depend on the actual mapping

results of all resampled data. We have shown that our algorithm largely matches conventional approaches for detecting unreliable markers that build framework maps by creating a distribution of the neighborhood relationships from the mapping of resampled data, and have a prohibitively high computational complexity. While our approach matches standard approaches in terms of marker alignment on the framework map as well as map cumulative distance, our algorithm outperforms standard approaches in terms of computation time. In addition, we have shown that other approaches that depend on filtering singletons from standard clustering algorithms are insufficient for filtering markers from RH data. Evaluation is carried out on radiation hybrid data for wheat chromosomes 1D and 2D.

CHAPTER 5. GENERAL CONCLUSIONS

In this dissertation, several algorithms for mining significant information from both structured and unstructured data formats have been proposed. The general theme of this dissertation is integrating data mining techniques with standard statistical methods. This integration made it possible to address statistical significance when solving complex problems in sciences.

Several applications of mining significant information have been introduced in this dissertation. In Chapter 2 an algorithm for identifying significant patterns between standardized items of information and textual representations of genomic data have been introduced. A re-weighting model is integrated with a density-based algorithm for finding the usefulness of textual representations for predicting biological class labels. The re-weighting model addressed problems due to the multi-relational nature of the data that would be responsible for spurious predictions when using standard classification techniques. According to the biological expectations, most protein domains are expected to be non-significant while gene ontology information is expected to be significant because most publications address gene functions rather than domain knowledge. The results shown in the evaluation confirm the biological expectations. Based on domain knowledge, more than half of the protein domains are found to be non-significant, while the Naive Bayes classifier predicted most of them as significant relationships. Most ontology functions were confirmed to be significantly related to the textual representations. In addition, two highest level gene ontology functions were predicted as non-significant using our algorithm but one of them was not predicted as non-significant by the comparison algorithm.

Several applications have been introduced in the ordering problem in radiation hybrid mapping. Detecting unreliable markers in genome mapping is addressed in Chapters 3 and 4. In Chapter 3, a network-based algorithm has been introduced for finding unreliable markers. The networks are created using the 2-piont LOD similarity measure. Standard

conventional approaches that rely on mapping data as part of a resampling analysis are used as a baseline model. The proposed network-based algorithm can detect the unreliable markers much faster than the baseline model, decreasing the run time by more than two orders of magnitude and outperforming a hierarchical clustering algorithm in terms of accuracy.

In Chapter 4, a modified network-based algorithm is introduced. The proposed support network is an iterative approach for finding the unreliable markers in genome mapping and building solid framework maps. While creating support networks is computationally fast in comparison to the baseline model, it largely matches the baseline model in terms of marker alignment on the framework map as well as map cumulative distance. Simple approaches that rely on filtering singletons from clustering algorithm for creating framework maps were proven to be insufficient for filtering noise from RH data.

In summary, several algorithms have been proposed for mining significant information from different data sources. Applications shown proved that the presented algorithms are efficient, effective, can be applied for high-dimensional data, and reduce computational cost.

REFERENCES

[1] M. Akbari, P. Wenzl, V. Caig, J. Carling, L. Xia, S. Yang, G. Uszynski, V. Mohler, A. Lehmensiek, and H. Kuchel, *Diversity arrays technology (dart) for high-throughput profiling of the hexaploid wheat genome*, TAG THEORETICAL AND APPLIED GENETICS **113 Number 8** (2006), 1409–1420.

[2] E. Akhunov, C. Nicolet, and J. Dvorak, *Single nucleotide polymorphism genotyping in polyploid wheat with the illumina goldengate assay*, Theor. Appl. Genet. **119** (2009), 507 517.

[3] O. Al-Azzam, J. Wu, L. Al-Nimer, C. Chitraranjan, and A. Denton, *A weighted density-based approach for identifying standardized items that are significantly related to the biological literature*, Text Mining Workshop in conjunction with the Eleventh SIAM International Conference on Data Mining, Mesa, AZ, USA, ACM, 2011.

[4] O. Al Azzam, L. Al Nimer, C. Chitraranjan, A. M. Denton, A.Kumar, F., M. J. Iqbal, and S. F. Kianian, *Network-based filtering of unreliable markers in genome mapping*, The Tenth International Conference on Machine Learning and Applications, ICMLA 2011, Honolulu, HI, USA, IEEE Computer Society, 2011.

[5] B. Birren, E. D. Green, P. Hieter, S.Klapholz, R. M. Myers, H. Riethman, and J. Roskams, *Genome analysis a library manual*, Cold Spring Harbor Laboratory Press, 1999.

[6] T. Brants, *Natural language processing in information retrieval*, CLIN, Antwerp papers in linguistics, vol. 111, University of Antwerp, 2003.

[7] A. Brard, MC. Le Paslier, M. Dardevet, F. Exbrayat-Vinson, I. Bonnin, A. Cenci, A. Haudry, D. Brunel, and C. Ravel, *High-throughput single nucleotide polymorphism genotyping in wheat (triticum spp.).*, Plant Biotechnol J. **7** (2009), 364–374.

[8] G. Carvalho, D. Martins de Matos, and V. Rocio, *Document retrieval for question answering: a quantitative evaluation of text preprocessing*, PIKM '07: Proceedings of the ACM first Ph.D. workshop in CIKM (New York, NY, USA), ACM, 2007, pp. 125–130.

[9] J. Chiang and H. Yu, *Meke: Discovering the functions of gene products from biomedical literature via sentence alignment*, Bioinformatics **19** (2003), no. 11, 1417–1422.

[10] L. E. Clifford, *Data analysis by resampling: Concepts and applications*, Duxbury (Pacific Grove, CA), 2000.

[11] D. Cox, M. Burmesiter, E. Price, S. kim, and R. Myers, *Radiation hybrid mapping: a somatic cell genetic method for constructing high-resolution maps of mammalian chromosomes*, Science **12 Vol. 250no 4978** (1990), 245–250.

[12] S. de Givry, M. Bouchez, P. Chabrier, D. Milan, and T. Schiex, *Carthagene: multipopulation integrated genetic and radiation hybrid mapping*, Genome analysis **21 no. 8** (2005), 1703–1704.

[13] A. M. Denton and J. Wu, *Data mining of vector-item patterns using neighborhood histograms*, Knowl. Inf. Syst. **21** (2009), no. 2, 173–199.

[14] T. Derrien, C. André, F. Galibert, and C. Hitte, *Autograph: an interactive web server for automating and visualizing comparative genome maps*, Bioinformatics **23** (2007), no. 4, 498–499.

[15] M. H. Dunham, *Data mining introductory and advanced topics*, Prentice Hall, 2003.

[16] E.M Elias and J.D Miller, *Registration of maier durum wheat*, Crop Sci. **40** (2000), 1498–1499.

[17] C. Elkan, *Deriving tf-idf as a fisher kernel*, SPIRE, Lecture Notes in Computer Science, vol. 3772, Springer, 2005, pp. 295–300.

[18] J. Ellson, E. Gansner, L. Koutsofios, S. North, G. Woodhull, S. Description, and L. Technologies, *Graphviz open source graph drawing tools*, Lecture Notes in Computer Science, Springer-Verlag, 2001, pp. 483–484.

[19] R. Elmasri and S. B. Navathe, *Fundamentals of database systems*, Pearson Addison Wesley, 2011.

[20] B.S Everitt, *The analysis of contingency tables*, CHAPMAN and HALL/CRC, London, 1992.

[21] W. Fan, L. Wallace, S.e Rich, and Z. Zhang, *Tapping the power of text mining*, Commun. ACM **49** (2006), no. 9, 76–82.

[22] W. Gao, Z. J. Chen, J. Z. Yu, R. J. Kohel, J. E. Womack, and D. M. Stelly, *Wide-cross whole-genome radiation hybrid mapping of the cotton (gossypium barbadense l.) genome*, Mol Gen Genomics **275** (2006), 105113.

[23] P. I. Good, *Resampling methods: a practical guide to data analysis*, 2 ed., Birkhduser, 2001.

[24] S. J. Goss and H. Harris, *New method for mapping genes in human chromosomes*, Nature **255** (1975).

[25] G.Shantanu and R. Shourya, *Text classification, business intelligence, and interactivity: automating c-sat analysis for services industry*, KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (New York, NY, USA), ACM, 2008, pp. 911–919.

[26] K. G. Hossain, O. Riera-Lizarazu, V. Kalavacharla, M. Isabel Vales, S. S. Maan, and S. F. Kianian, *Radiation hybrid mapping of the species cytoplasm-specific (scsae) gene in wheat*, Genetics Society of America **168** (2004), 415423.

[27] E. Frank I. H. Witten, *Data mining practical machine learning tools and techniques*, Morgan Kaufmann Publishers, 2005.

[28] T. R. Inniss, J. R. Lee, M. Light, M. A. Grassi, G. Thomas, and A. B. Williams, *Towards applying text mining and natural language processing for biomedical ontology acquisition*, TMBIO '06: Proceedings of the 1st international workshop on Text mining in bioinformatics (New York, NY, USA), ACM, 2006, pp. 7–14.

[29] T. Joachims, *A probabilistic analysis of the rocchio algorithm with tfidf for text categorization*, ICML (D. H. Fisher, ed.), 1997, pp. 143–151.

[30] H. L. Johnson, K. B. Cohen, and L. Hunter, *A fault model for ontology mapping, alignment, and linking systems*, Pacific Symposium on Biocomputing, World Scientific, 2007, pp. 233–268.

[31] V. Kalavacharla, K. Hossain, Y. Gu, O.Riera-Lizarazu, M. I. Vales, S. Bhamidimarri, J. L. Gonzalez-Hernandez, S. S. Maan, and S. F. Kianian, *High-resolution radiation hybrid map of wheat chromosome 1d*, Genetics Society of America **173** (2006), 1089–1099.

[32] V. Kalavacharla, K.Hossain, O.Riera-Lizarazu, Y. Gu, S. S. Maan, and S. F. Kianian, *Advances in agronomy*, vol. 102, pp. 201–222, Elsevier Inc., 2009.

[33] D. Koller, *Probabilistic relational models*, ILP, Lecture Notes in Computer Science, vol. 1634, Springer, 1999, pp. 3–13.

[34] R. G. Kynast, R. J. Okagaki, M. W. Galatowitsch, S. R. Granath, M. S. Jacobs, A. O. Stec, H. W. Rines, and R. L. Phillips, *Dissecting the maize genome by using chromosome addition and radiation hybrid lines*, PNAS **101 no. 26** (2004), 99219926.

[35] E. S. Lander, P. Green, J. Abrahamson, A. Barlow, M. J. Daly, S. E. Lincoln, and L. Newburg, *Mapmaker: an interactive computer package for constructing primary genetic linkage maps of experimental and natural populations*, Genomics **1 No. 2** (1987), 174–181.

[36] W. Li, P. Zhang, J.P. Fellers, B. Friebe, and B.S. Gill, *Sequence composition, organization, and evolution of the core triticeae genome.*, Plant J. **40** (2004), 500511.

[37] Y. A. Lussier, T. Borlawsky, D. Rappaport, Y. Liu, and C. Friedman, *Phenogo: Assigning phenotypic context to gene ontology annotations with natural language processing*, Pacific Symposium on Biocomputing, World Scientific, 2006, pp. 64–75.

[38] L. C. McCarthy, J. Terrett, and M. E. Davis, *A first-generation whole genome-radiation hybrid map spanning the mouse?genome*, Genome Research **7** (1997), 1153–1161.

[39] D. I. Mester, Y. I. Ronin, M. A. Korostishevsky, V. L. Pikus, A. E. Glazman, and A. B. Korol, *Multilocus consensus genetic maps (mcgm): Formulation, algorithms, and results*, Computational Biology and Chemistry **30** (2006), no. 1, 12–20.

[40] D.I. Mester, Y. I. Ronin, Y. Hu., J. Peng, E. Nevo, and A. B. Korol, *Efficient multipoint mapping: making use of dominant repulsion-phase markers.*, Theo Appl Genet **107** (2003), 1102–1112.

[41] D.I. Mester, Y. I. Ronin, D. Minkov, E. Nevo, and A. B. Korol, *Constructing large-scale genetic maps using an evolutionary strategy algorithm*, Genetics Society of America **165** (2003), 2269–2282.

[42] K. A. Mieczyslaw, *Very large bayesian multinets for text classification*, Future Gener. Comput. Syst. **21** (2005), no. 7, 1068–1082.

[43] H. Mima, S. Ananiadou, and K. Matsushima, *Terminology-based knowledge mining for new knowledge discovery*, ACM Trans. Asian Lang. Inf. Process. **5** (2006), no. 1, 74–88.

[44] R. J. Mooney and R. Bunescu, *Mining knowledge from text using information extraction*, SIGKDD Explor. Newsl. **7** (2005), no. 1, 3–10.

[45] R. Nielsen, J. S. Paul, A. Albrechtsen, and Y. S. Song, *Genotype and snp calling from next-generation sequencing data*, Nature Reviews Genetics **12** (2011), 443–451.

[46] M. Olivier, A. Aggarwal, and J. Allen, *A high-resolution radiation hybrid map of the human genome draft sequence.*, Science **291** (2001), 1298–1302.

[47] M. Porter, *Porter stemming algorithm http://tartarus.org/ martin/PorterStemmer/*, Nov. 2009.

[48] J.S Quick, D.E. Walsh, K.L Lebsock, and J.D Miller, *Registration to rugby durum wheat*, Crop Sci. **15** (1975), 604.

[49] M. W. Geatz R. J. Roiger, *Data mining a tutorial-based primer*, Pearson Addison Wesley, 2003.

[50] O. Riera-Lizarazu, M. I. Vales, E. V. Ananiev, H. W. Rines, and R. L. Phillips, *Production and characterization of maize chromosome 9 radiation hybrids derived from an oat-maize addition line*, Genetics Society of America **156** (2000), 327339.

[51] Y. I. Ronin, D. I. Mester, D. Minkov, and A. B. Korol, *Building reliable genetic maps: different mapping strategies may result in different maps*, Natural Sciences **2 No.6** (2010), 576–589.

[52] A. A. Schaffer, E. S. Rice, W. Cook, and R. Agarwala, *rh tsp map 3.0: end-to-end radiation hybrid mapping with improved speed and quality control*, Genome analysis **23 no. 9** (2007), 1156–1158.

[53] M. J. Schervish, *Theory of statistics*, Springer, 1995.

[54] H. J. Seltman, *Experimental design and analysis*, 2011.

[55] M. D. Smucker, J. Allan, and B. Carterette, *Agreement among statistical significance tests for information retrieval evaluation at varying sample sizes*, SIGIR, ACM, 2009, pp. 630–631.

[56] M. D. Smucker, A. James, and B. Carterette, *A comparison of statistical significance tests for information retrieval evaluation*, CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management (New York, NY, USA), ACM, 2007, pp. 623–632.

[57] Y. Song, J. Huang, D. Zhou, H. Zha, and C. L. Giles, *Iknn: Informative k-nearest neighbor pattern classification*, PKDD, Lecture Notes in Computer Science, vol. 4702, Springer, 2007, pp. 248–264.

[58] I. Spasic and S. Ananiadou, *Using automatically learnt verb selectional preferences for classification of biomedical terms*, Journal of Biomedical Informatics **37** (2004), no. 6, 483–497.

[59] E. Stefan, *Significance tests for the evaluation of ranking methods*, COLING '04: Proceedings of the 20th international conference on Computational Linguistics (Morristown, NJ, USA), Association for Computational Linguistics, 2004, p. 945.

[60] P. Tan, M. Steinbach, and V. Kumar, *Introduction to data mining*, Pearson Addison Wesley, 2006.

[61] D. Trebbi, M. Maccaferri, P. de Heer, A. Srensen, S. Giuliani, S. Salvi, MC. Sanguineti, A. Massi, EA. van der Vossen, and R. Tuberosa, *High-throughput snp discovery and genotyping in durum wheat*, (Triticum durum Desf.). **123** (2011), 555–569.

[62] A. Valencia, *Text mining in genomics and systems biology*, DTMBIO '08: Proceeding of the 2nd international workshop on Data and text mining in bioinformatics (New York, NY, USA), ACM, 2008, pp. 3–4.

[63] M. A. Walter, D. J. Spillett, P. Thomas, J. Weissenbach, and P. N. Goodfellow, *A method for constructing radiation hybrid maps of whole genomes*, Nature Genetics **7** (1994), 22–28.

[64] Q. Xiaoguang and D. D. Brian, *Web page classification: Features and algorithms*, ACM Comput. Surv. **41** (2009), no. 2, 1–31.

[65] L. Xiong, S. Chitti, and L. Liu, *k nearest neighbor classification across multiple private databases*, CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management (New York, NY, USA), ACM, 2006, pp. 840–841.

[66] A. S. Yeh, L. Hirschman, and A. A. Morgan, *Evaluation of text data mining for database curation: lessons learned from the kdd challenge cup*, ISMB (Supplement of Bioinformatics), 2003, pp. 331–339.

[67] C. Zhang, X. Lu, and X. Zhang, *Significance of gene ranking for classification of microarray samples*, IEEE/ACM Trans. Comput. Biol. Bioinformatics **3** (2006), no. 3, 312–320.

[68] L. Zhang, D. Zhang, S. J. Simoff, and J. Debenham, *Weighted kernel model for text categorization*, AusDM '06: Proceedings of the fifth Australasian conference on Data mining and analystics (Darlinghurst, Australia, Australia), Australian Computer Society, Inc., 2006, pp. 111–114.

[69] P. Zweigenbaum, D. Demner-Fushman, H. Yu, and K. B. Cohen, *Frontiers of biomedical text mining: current progress*, Briefings in Bioinformatics **8** (2007), no. 5, 358–375.

APPENDIX A. SIGNIFICANCE CALCULATION

Hypotheses testing is a statistical way for determining if a set of observations can occur by random chance. There are two parts of hypotheses testing:

- H_0 : the observed and the expected data do not differ
- H_1 : the observed and the expected data differ significantly

Under a specific significance level α the null hypotheses H_0 is either accepted or rejected. The null hypothesis is accepted only if the derived $P-$value is less than the significance level α. These $P-$values are derived based on the distribution that the data follow. In this study the χ^2 distributions is used:

- χ^2 distribution: used to determine if the distribution of two sets of random variables (categorical data) differ. Using Table A. 1 the $P-$values can be calculated using the equation: $\sum(Observed-Expected)^2/Expected$. The observed data in a classification style results is the (2 X 2) confusion matrix. The matrix is treated as a contingency table and the $P-$values is derived using the above equation with one degree of freedom ((number of rows -1) * (number of columns -1))

Table A. 1: Chi-Square Probabilities

df/α	0.995	0.99	0.975	0.95	0.90	0.10	0.05	0.025	0.01	0.005
1	—	—	0.001	0.004	0.016	2.706	3.841	5.024	6.635	7.879
2	0.010	0.020	0.051	0.103	0.211	4.605	5.991	7.378	9.210	10.597
3	0.072	0.115	0.216	0.352	0.584	6.251	7.815	9.348	11.345	12.838
4	0.207	0.297	0.484	0.711	1.064	7.779	9.488	11.143	13.277	14.860
5	0.412	0.554	0.831	1.145	1.610	9.236	11.070	12.833	15.086	16.750
6	0.676	0.872	1.237	1.635	2.204	10.645	12.592	14.449	16.812	18.548
7	0.989	1.239	1.690	2.167	2.833	12.017	14.067	16.013	18.475	20.278
8	1.344	1.646	2.180	2.733	3.490	13.362	15.507	17.535	20.090	21.955
9	1.735	2.088	2.700	3.325	4.168	14.684	16.919	19.023	21.666	23.589
10	2.156	2.558	3.247	3.940	4.865	15.987	18.307	20.483	23.209	25.188
11	2.603	3.053	3.816	4.575	5.578	17.275	19.675	21.920	24.725	26.757
12	3.074	3.571	4.404	5.226	6.304	18.549	21.026	23.337	26.217	28.300
13	3.565	4.107	5.009	5.892	7.042	19.812	22.362	24.736	27.688	29.819
14	4.075	4.660	5.629	6.571	7.790	21.064	23.685	26.119	29.141	31.319
15	4.601	5.229	6.262	7.261	8.547	22.307	24.996	27.488	30.578	32.801
16	5.142	5.812	6.908	7.962	9.312	23.542	26.296	28.845	32.000	34.267
17	5.697	6.408	7.564	8.672	10.085	24.769	27.587	30.191	33.409	35.718
18	6.265	7.015	8.231	9.390	10.865	25.989	28.869	31.526	34.805	37.156
19	6.844	7.633	8.907	10.117	11.651	27.204	30.144	32.852	36.191	38.582
20	7.434	8.260	9.591	10.851	12.443	28.412	31.410	34.170	37.566	39.997
21	8.034	8.897	10.283	11.591	13.240	29.615	32.671	35.479	38.932	41.401
22	8.643	9.542	10.982	12.338	14.041	30.813	33.924	36.781	40.289	42.796
23	9.260	10.196	11.689	13.091	14.848	32.007	35.172	38.076	41.638	44.181
24	9.886	10.856	12.401	13.848	15.659	33.196	36.415	39.364	42.980	45.559
25	10.520	11.524	13.120	14.611	16.473	34.382	37.652	40.646	44.314	46.928
26	11.160	12.198	13.844	15.379	17.292	35.563	38.885	41.923	45.642	48.290
27	11.808	12.879	14.573	16.151	18.114	36.741	40.113	43.195	46.963	49.645
28	12.461	13.565	15.308	16.928	18.939	37.916	41.337	44.461	48.278	50.993
29	13.121	14.256	16.047	17.708	19.768	39.087	42.557	45.722	49.588	52.336
30	13.787	14.953	16.791	18.493	20.599	40.256	43.773	46.979	50.892	53.672
40	20.707	22.164	24.433	26.509	29.051	51.805	55.758	59.342	63.691	66.766
50	27.991	29.707	32.357	34.764	37.689	63.167	67.505	71.420	76.154	79.490
60	35.534	37.485	40.482	43.188	46.459	74.397	79.082	83.298	88.379	91.952
70	43.275	45.442	48.758	51.739	55.329	85.527	90.531	95.023	100.425	104.215
80	51.172	53.540	57.153	60.391	64.278	96.578	101.879	106.629	112.329	116.321
90	59.196	61.754	65.647	69.126	73.291	107.565	113.145	118.136	124.116	128.299
100	67.328	70.065	74.222	77.929	82.358	118.498	124.342	129.561	135.807	140.169

APPENDIX B. CLASSIFICATION STYLE MEASURES

The results of a classification style data is represented in (2 X 2) confusion matrix as can be seen in the below table:

Table B. 1: Classification Style Confusion Matrix

TP	FN
FP	TN

Where,

- TP: are the true positives. Data with class label 1 and predicted as 1.
- FN: are the false negatives. Data with class label 1 and predicted as 0 (type I error).
- FP: are the false positives. Data with class label 0 and predicted as 1 (type II error).
- TN: are the true negatives. Data with class label 0 and predicted as 0.

The below measures are used across this study:

$$Sensitivity = \frac{TP}{TP + FN} \tag{8}$$

$$Specificity = \frac{TN}{TN + FP} \tag{9}$$

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \tag{10}$$