Preface

This Master thesis is written as a part of the Master of Science Program in Bioinformatics at the Norwegian University of Science and Technology (NTNU) at the Department of Computer and Information Science.

Acknowledgements

I would like to express my special thanks to my supervisor Pål Sætrom at NTNU who patiently walked with me throughout the entire work. His wide knowledge and brilliant ideas have been of a great value to me.

Many thanks to my parents, Zahra & sadegh, and brothers, Pooya & Mohsen, for their love and support, especially Pooya for his kind advices.

I also thank my friends and members of Bioinformatics group for making my time at NTNU enjoyable.

Abstract

The cell cycle is an important biological process in which a set of events occurs in a sequential manner progressing to the cell division. Cell cycle is regulated by periodic fluctuations in the expression levels of several genes referred to as cell cycle regulated genes. In this study, we apply machine learning techniques to prioritize a list of candidate genes with respect to being involved in the cell cycle regulation process. We focus on the data obtained from different expression experiments on which partial least squares regression (PLS) models have been previously developed to identify genes with cell cycle dependent expression profiles. The different expression experiments used different synchronization methods to halt the cell cultures, so that each experiment started to measure gene expression values at different cell cycle phases after synchronization. We are mainly interested in genes having cyclic expression profile which is consistent with respect to cell cycle phases within all experiments. Our goal is therefore to develop a method that can identify gens that have consistent cyclic expression profiles across multiple synchronization experiments.

We solve the cell cycle related gene prioritization problem through a novelty detection approach using one-class support vector machine. The candidate genes are ranked according to their similarity to the genes with known cell cycle function. It is found that some of the top ranked genes are involved in biological processes related to the cell cycle, that is a good indication that our approach is able to prioritize genes with cell cycle function.

Keywords: Cell cycle, Partial least squares (PLS), Gene prioritization, one-class support vector machine.

Contents

List of Figures

List of Tables

Acronyms

A	Adenine
AUC	Area under the curve
C	Cytosine
CCNE2	Cyclin E2
Cdk	Cyclin-dependent kinase
cDNA	complementary DNA
CV	Cross validation
DAVID	Database for annotation, visualization and integrated discovery
DNA	Deoxyribonucleic acid
FN	False negative
FP	False positive
FPR	False positive rate
G	Guanine
G0	quiescence phase
G1	First gap
G2	Second gap
GO	Gene ontology
LOOCV	Leave one out cross validation
M	Mitotic
MKL	Multiple kernel learning
ML	Machine learning
MPF	M- phase-promoting factor
mRNA	messenger RNA
NCBI	National center for biotechnology information
OCSVM	One-class support vector machine
PLS	Partial least squares
QP	Quadratic programming

RBF	Radial basis function
RNA	Ribonucleic acid
ROC	Receiver operating characteristic
S	Synthesis
SN	Sensitivity
SP	Specificity
SRM	Structural risk minimization
SVM	Support vector machine
T	Thymine
TN	True negative
TP	True positive
TPR	True positive rate
U	Uracil
W	Weight vector

1. Introduction

This introductory chapter presents the biological context and several basic background topics for understanding the development of this thesis. Section 1.1 introduces the concept of the cell cycle. Section 1.2 focuses on the cell cycle control system and regulation of the cell cycle. Section 1.3 is dedicated to cell synchronization and presents a summary of several synchronization methods. Section 1.4 is a brief overview of the basic concepts involved in microarray experiments. Section 1.5 outlines the approach of partial least squares and describes how it is being applied to identify genes with cyclic expression profiles. Section 1.6 presents the theory of support vector machine and its mathematical foundations. Section 1.7 gives a brief summary of multiple kernel learning approach. Section 1.8 introduces three different methods used for integration of datasets. Section 1.9 describes the gene prioritization problem that is the focus of this thesis.

1.1. Cell cycle

The cell cycle is a complex biological process in which a set of cellular stages occurs in a sequential manner progressing to the cell division. The cell cycle is a period of time in which a cell is formed from its dividing parent cell until its own division into two cells. Cell division is the main component of the cell cycle (Campbell et al. 2008).

In prokaryotic (cells without a nucleus), the cell cycle is done through a process called binary fission. In eukaryotes (cells with a nucleus), the cell cycle is divided in two major phases: a growing phase (inter phase) and mitotic phase (M phase). The interphase, in which the cell grows and its chromosomes are copied to be prepared for the cell division, often consists of 90 percent of the cell cycle.

The interphase proceeds in three sub phases: the first gap (G1 phase), the synthesis phase (S phase), and the second gap (G2 phase). The cell continues growing during all three sub phases. G1 is the starting phase of the cell cycle and it is where a cell grows and increases in size. DNA replication happens during S phase. Each chromosome is single in the start of S phase. After DNA replication, at the end of S phase, the chromosomes are double. In G2 the cell grows more to complete preparation for the cell division.

The M phase is the shortest part during the cell cycle and consists of only about 10 percent of the cell cycle. M phase is divided into two processes: mitosis and cytokinesis. The division of the nucleus happens in mitosis and is followed by cytokinesis, the division of the cytoplasm. Growth of the cell stops at the M phase and cellular energy is used for the division of the cell into two daughter cells. The cell cycle may then be repeated by the daughter cells through G1 phase (Alberts et al. 2008; Campbell et al. 2008).

Cells that have temporarily left the cell cycle and stopped dividing are in a state of quiescence called G_0 phase. In the human body, most of the cells are actually in the G_0 phase. For instance, mature, fully formed nerve cells and muscle cells do not divide at all while skin cells divide frequently.

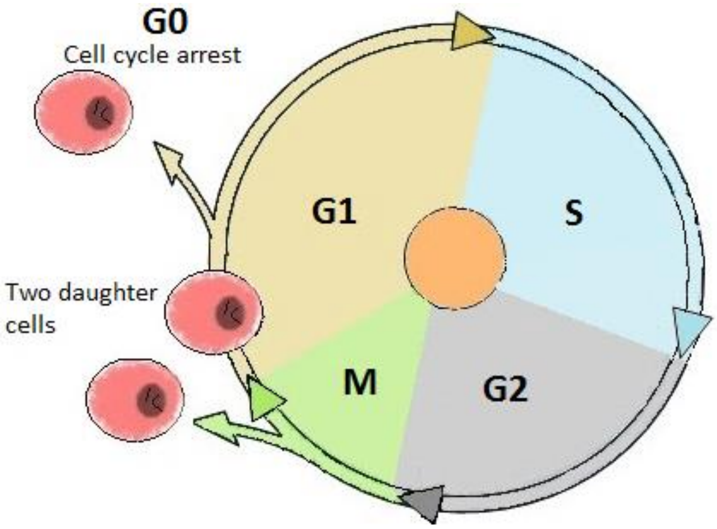

Figure 1: Phases of the eukaryotic cell cycle
(Adopted from Campbell 2008)

1.2. Regulation of the cell cycle

The sequential events of the cell cycle are controlled by a cell cycle control system. The cell cycle control system contains a set of molecules in the cell that operates cyclically to trigger and coordinate what happens in the cell cycle. The cell cycle is regulated at certain check points in all of its phases. Checkpoints are used by the cell to control the progress of the cell cycle and ensure that the received amount of genetic information in the resulting daughter cells is in the appropriate amount. If any damages are distinguished by checkpoints, they respond to it by halting the cell cycle to provide time for repair.

Three main cell cycle checkpoints are: the G_1 checkpoint also known as the restriction point, the G_2 checkpoint, and the M checkpoint also known as the spindle checkpoint. The G1 checkpoint is placed at the end of the cell cycle's G_1 phase, before entry into the S phase. It makes sure that the cell is large enough to enter the S phase. If a go ahead signal is received by a cell at the G1 checkpoint, the cell proceeds to the S phase. In the case of not receiving a go ahead signal, the cell leaves the cell cycle and switches into the quiescent G0 phase. The G2 checkpoint is placed at the end of the G_2 phase. It makes the decision that DNA replication in S phase has been completed, replication errors have been fixed, and the cell is ready for mitosis. The M checkpoint is placed at the end of the M phase. It ensures whether all the chromosomes are correctly attached to the spindle microtubules (Elledge 1996).

In addition, to avoid abnormal cell growth that could cause tumor development, the cell cycle control system regulates when and how much the cells of a given tissue proliferate. Tumors are caused by abnormal proliferation resulting from alterations in cell cycle regulatory system. In the cases where one or more of such controls are disrupted, abnormal excessive growth will happen leading to defects and diseases (Whitfield et al. 2006).

The transitions between the cell cycle stages are regulated via a complex network of protein interactions. Then, understanding the interaction and modification of such proteins are crucial to find out the dynamics of the cell cycle. The regulatory proteins are mainly protein kinases and cyclins. Most of the time, kinases that derive the cell cycle are in an inactive form. A kinase is activated when it binds to a cyclin, and changes in the concentration of a cyclin partner leads to the changes in the activity of a kinase. For this requirement, such a kinase is called a cyclin-dependent kinase (Cdk). When the Cdks are activated via cyclins, they can perform a common biochemical reaction termed phosphorylation that activates or inactivates target proteins. This modification serves as a signal for entry to the next phases during the cell cycle progression. Mitosis promoting factor or M- phase-promoting factor (MPF) is also a regulatory protein composed of Cdk and cyclins and is activated at the end of G_2. After the accumulated cyclin during the G2 binds to Cdk molecules, the resulting MPF triggers the entrance into the M phase from the G2 phase through phosphorylating a variety of proteins required during the M phase. During anaphase (one of the mitosis stages), MPF initiates a process resulting in the degradation of its cyclin component and termination of the M phase. The Cdk part of MPF remains in the cell inactively until it binds to new molecules synthesized during the S and G2 phases of the next cell cycle (Campbell et al. 2008).

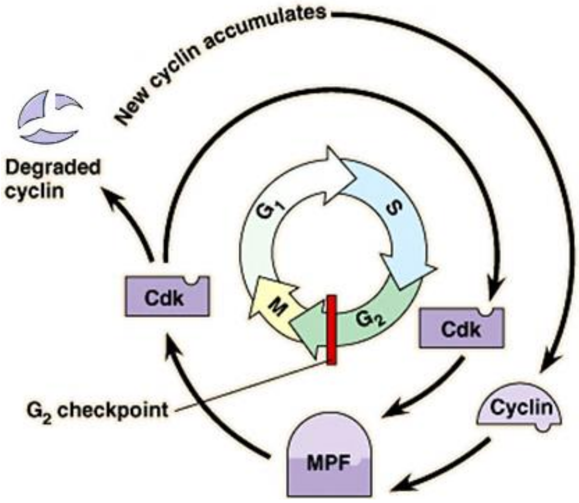

Figure 2: The molecular mechanisms that help regulate the cell cycle (Adopted from Campbel 2008)

1.3. Cell synchronization

Cells growing in the culture are asynchronous with respect to phases of the cell cycle. The study of different stages of the cycle**,** the cell cycle control system and identifying the cell cycle regulated genes require synchronizing the cells. Cell synchronization is the process of bringing cells at different stages of the cell cycle to one particular stage of the cell division (Chou and Langan 2003). There is a number of methods for synchronizing the cell cultures by halting the cell cycle at a particular phase (Merrill 1998). A brief summary of these methods is presented in the following.

1.3.1. Serum starvation

To synchronize cells, a common and often-described technique is depriving cells of nutrients needed to proliferate, by placing them in a low concentration of serum which produces growth arrest. This puts the cells into quiescence phase (G0) or G0/G1 phase (shown in Figure 3) (Khammanit et al. 2008). For synchronizing the cells using this method, it is crucial to be sure that the cells are not at confluence when the serum is eliminated. The cells can be released from quiescence by resumption of growth, addition of normal serum concentrations, or treating with certain growth factors that induce cells to re-enter the cell cycle (Whitfield et al. 2006).

1.3.2. Using chemical inhibitors to synchronize cells

Other techniques use chemical inhibitors which arrest cells in specific points of the cell cycle. The inhibitor is then eliminated from the media and the cells can progress in the cell cycle synchronously. There are various inhibitors available to halt cells in different points of the cell cycle. When the purpose is to synchronize the cells and then keep them going afterwards, it is important to use reversible inhibitors (Whitfield et al. 2006). Treatment with drugs such as aphidico, thymidine and hydroxyurea halts cells in S phase by preventing G1 cells from entering the DNA synthesis stage. Consequently cells is G2, M and G1 phases will continue the cell cycle and accumulate at the G1/S border (Pedrali-Noy et al. 1980; Morgan D. O. 2007). Alternatively treatment with nocodazole halts cells in mitosis (M) phase (Figure 3) (Zieve et al. 1980).

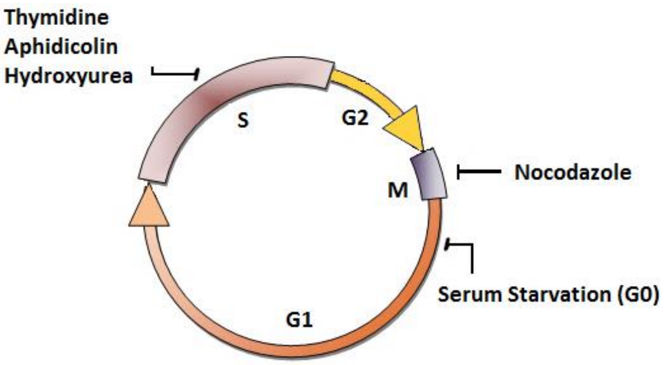

Figure 3: Identification of cell-cycle-regulated genes by synchronizing the cell cultures. (Adopted from Whitfield et al. 2006)

1.4. Microarray technology

Microarray technology is a high throughput technology used in molecular biology. It enables researchers to monitor and measure changes in the expression levels of thousands of genes in parallel under different biological conditions (Zvelebil and Baum 2008). Previous analysis of expression data has shown that the expression patterns in microarray experiments are similar in the genes with similar functions (Brown et al. 2000). Large-scale genotyping, comparison of genomic hybridization, and gene expression profiling are the most common applications that use microarray technology (Dufva 2009).

The following section is a brief overview of the main concepts involved in a microarray experiment.

1.4.1. What are microarrays?

A microarray is a solid support (surface) on which DNA fragments are immobilized at specific positions called spots or probes in a predetermined arrangement. A microarray may consist of thousands of spots, and the DNA fragments fixed in spots are usually cellular mRNAs that have been converted to cDNAs by reverse transcription or a short stretch of oligo-nucleotides corresponding to a single gene (Stears et al. 2003, Selvaraj and Natarajan 2011). The solid surface to which these DNA molecules are attached can be a glass, plastic or silicon chip, commonly known as the genome chip or the gene array. Also an Affymetrix chip can be used that is known as an Affy chip. Another microarray platform, such as Illumina, consists of a collection of technical replicates (microscopic beads), instead of the solid support. On Illumina arrays the oligonucleotides are attached to beads which are then printed on to the solid surface by different technologies such as photolithography or robot spotting (Barnes et al. 2005). In the common form of a microarray experiment, the mRNAs in the target sequence (target molecules) are labeled using fluorescent tags and mixed with the array so that the probe-target hybridization can be detected via emission of a fluorescent light (Madan Babu 2004).

The main element in a microarray technology is hybridization between two DNA strands through forming hydrogen bonds between complementary nucleotide base pairs. Hybridization, also called base-pairing, is the process of joining one single stranded DNA fragment to the DNA fragment with the complementary sequence. Hybridization takes place between single stranded fragments of DNA, mRNA and cDNA and the result is double stranded fragments. In both DNA and RNA the base adenine (A) attaches to base thymine (T) in DNA and base uracil (U) in RNA, and base guanine (G) attaches to base cytosine (C) (Campbell 2008).

The most common application of gene expression microarray experiments is to detect the genes being expressed at a given time or compare gene expression under different conditions. To compare gene expression in two different conditions, a common method is to label two samples with different fluorescent dyes. For example, the cDNA from one sample is labeled with a red dye and that of another sample with a green dye. The two differentially labeled samples are then allowed to hybridize to particular probes on the array containing its complementary sequence. After washing away unbounded sequences, only strongly paired strands will remain hybridized. Then, the microarray is scanned with a florescence imager to detect the probes that have become fluorescently labeled. The florescence intensity emitted upon excitation reflects the amount of target sample binding to the probes (Madan Babu 2004; Zvelebil and Baum 2008). A microarray platform and the experimental steps involved are illustrated in Figure 4.

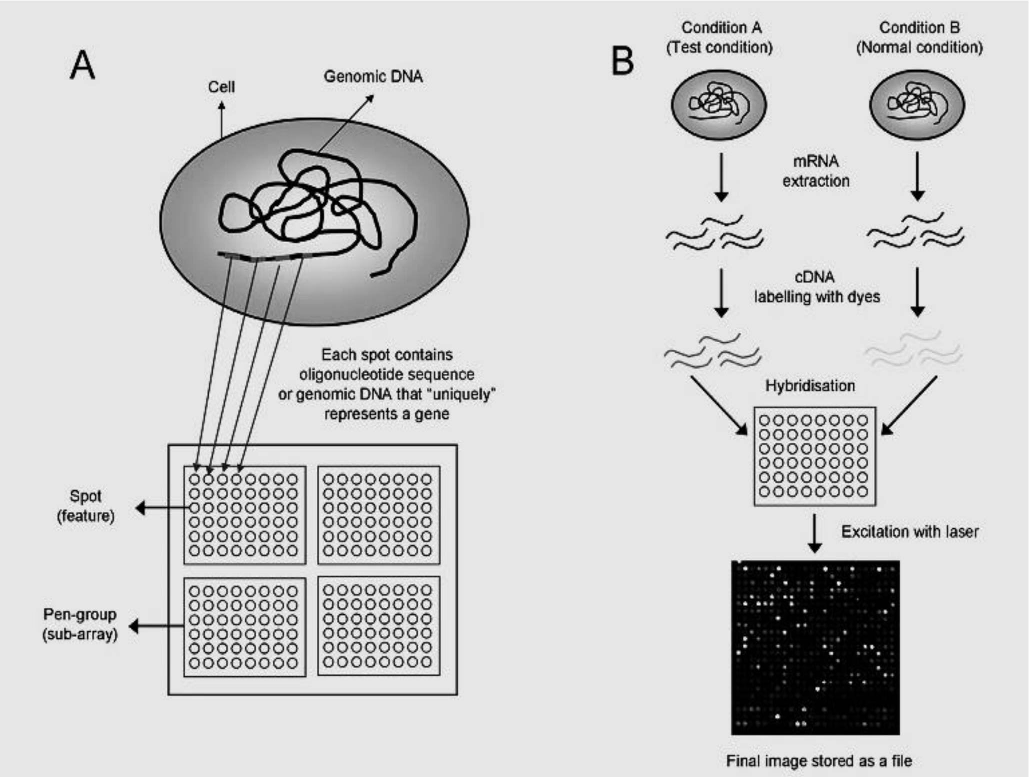
A
Cell
Genomic DNA
Each spot contains oligonucleotide sequence or genomic DNA that "uniquely" represents a gene
Spot (feature)
Pen-group (sub-array)
B
Condition A (Test condition)
Condition B (Normal condition)
mRNA extraction
cDNA labelling with dyes
Hybridisation
Excitation with laser
Final image stored as a file

Figure 4: (A) A typical microarray platform. (B) An overview of a microarray experiment and experimental steps involved to study differential expression of genes grown in two different conditions (a reference condition and a test condition), (From Madan Babu 2004).

1.5. Partial least squares

Partial least squares (PLS) method is a multivariate regression approach that is suited for the analysis of highly dimensional data in bioinformatics and genomics. The PLS regression method has gained popularity in monitoring multivariate processes in the last decade, see, e.g., (Stone and Brook 1990; Frank and Friedman 1993; Garthwaite 1994). This was mostly because of the ability of multivariate statistical PLS regression to handle data with a large number of parameters and small sample sizes (Boulesteix and Strimmer 2006). It is valuable to apply the PLS method to analyze microarray data. Gene expression data from, since DNA microarray hybridization experiments have many measured variables (genes) and only a few observations (experiments). Some issues like noisy and missing data make it difficult to analyze the results of microarray experiments (Johansson et al. 2003). In the following section, we provide a short overview of PLS model.

PLS is a dimension reduction method which is joined with a regression model. In the PLS regression method, the dataset (data matrix) X is associated to a response variable y. To minimize the dimensionality of a data matrix X by using PLS, principal components are constructed. To acquire the principal components, the covariance between a linear combination of the original variables $t = Xw$, (X is the data matrix, w is the weight vector and t is the score vector) and the y response is optimized by PLS regression. To study more about PL, see, e.g, (Martens and Naes 1989; Hoskuldsson 1996; Burnham et al. 1999; Boulesteix and Strimmer 2006).

PLS is usually applied to construct a model which predicts a y-response from X. However, in this study the PLS regression has been used based on Johansson et al.'s modeling approach for identification of genes that are expressed periodically. They made a virtual response Y that shows cyclic behavior containing the same periodicity as the cell cycle. According to Johansson et al.'s: "The response is constructed to represent cyclic behavior with the same periodicity as the cell cycle. Thus, the variables/genes that contribute significantly to the models have expression patterns that appear to be coupled to the cell cycle (cyclically expressed). These genes include those that regulate cell cycling as well as those that are regulated by it" (Johansson et al. 2003). The following describes how the PLS model is being applied to identify genes with cyclic expression profiles.

Visual examination of the expression profiles of genes has illustrated that genes that are cyclically expressed have expression profiles that are similar to sine curves. Moreover, it has been shown that sine functions model cell cycling patterns in a precise manner (Alter et al. 2000). Since the genes are expressed in different phases of the cell cycle, there are different phase angles in sine curves. If it is considered that all cyclically expressed genes are distinguished by periodic expression data, in the form of $sin(\omega t + \varphi)$, with different phases, φ, and a constant frequency ω, then all cyclic gene expression profiles are modeled by applying two models, $sin(\omega t)$ and $cos(\omega t)$ (Johansson et al. 2003).

To interpret the importance of the variables, the weight vector parameter of the first principal component, w_1, is applied and it is calculated for increasing the estimated covariance of Xw_1 and y. Genes with large w-value will vary significantly with y (Johansson et al. 2003).

Two models are required for a y-response, y_1 for the sine curve and y_2 for the cosine curve. A cyclically expressed gene whose expression profile resembles the sine curve, will score significantly in the first model, and will score slightly in the second one. For identifying such genes from all stages in the cell cycle, we need to employ the weight vector (w_1) of each model. For doing this, the weight vector, w_1, for the two models is plotted against each other. Each point in the plane is the representation of a particular gene. Distance d , is the length from the centre of the coordinates to the point and shows the significance of a candidate gene to be coupled to the cell cycle (Figure 5). Non-cyclic expression genes are located close to the origin, while cyclically expressed genes are lied in a greater distance from the origin (Johansson et al. 2003). In addition, the direction from the origin is identified by the phase angle of the cyclic variation and this direction is same for genes with the same temporal expression patterns.

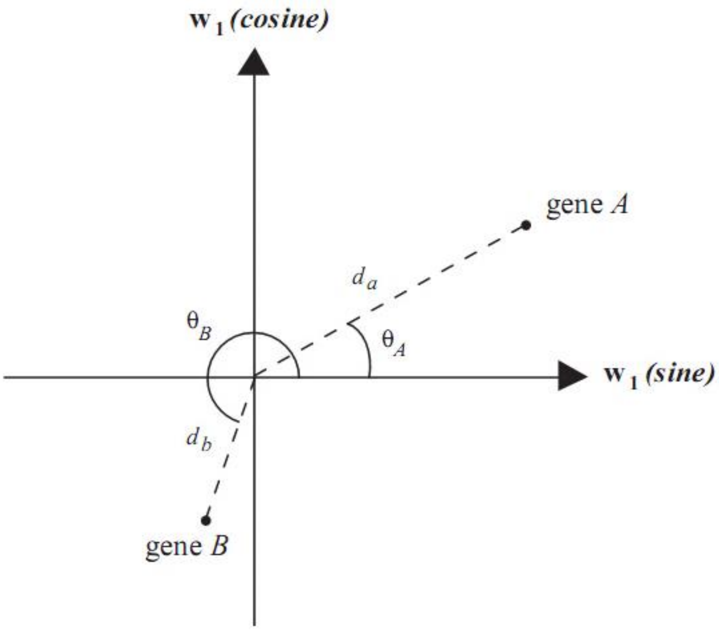

Figure 5: Ranking and assigning cell cycle phase. Cyclic expression genes are far away from the center. Genes A and B are ranked according to the distances da and db, respectively, and the cell cycle phase assignments are determined by the angles ϑA and ϑB, respectively.(From Johansson et al. 2003)

1.6. Support vector machine

In this part we first present a summary about the theory of support vector machine (SVM) and its application. Then, we briefly introduce the mathematical foundations of two-class and one-class SVM.

1.6.1. Support vector machine: Theory

The support vector machine (SVM) is a state of the art machine learning tool for classification and regression. SVM was first introduced by by Boser, Guyon and Vapnik in 1992 (Boser et al. 1992). SVM is based on the principle of structural risk minimization (SRM) from Vapnik's statistical learning theory (1982). Statistical learning theory is a method for theoretically analyzing the problem of function estimation from a given data collection and also a method for making practical algorithms to estimate multidimensional functions. SRM's purpose is to find a hypothesis which has the lowest probability of error. In Vapnik's theory (1998), the bounds on the error are associated with the margin of separating hyperplanes. SVM's goal in their basic form is to maximize the margin of a hyper plane that separates the training data.

SVMs have been successfully applied in many applications ranging from text categorization, face detection, hand writing recognition, and information extraction to bioinformatics. (Cristianini and Shawe-Taylor 2000). The properties that make the SVMs system of choice for these applications are flexibility in choosing a similarity function, sparseness of solution when there are large datasets, the ability to handle large feature spaces and identifying outliers (Brown et al. 2000). However, SVMs suffer from slow training especially with large input data size and non-linear kernels. Using SVMs efficiently needs an understanding of how they work. The mathematical foundation of SVMs is described in detail in (Vapnik 1998; Kecman 2001; Müller et al. 2001; Alpaydin 2004). Section 1.6.2-1.6.4 gives a brief introduction about the mathematical background of SVMs in the linearly separable and non-linearly separable cases.

1.6.2. Linear support vector machines

Let us consider that the data is linearly separable meaning that there is a hyperplane that classifies all data points in two classes. We are given training data points $x_i \in R^d \ (i = 1,...,l)$ with corresponding labels y_i that is either +1 or −1, representing the class to which the data point x_i belongs, and the goal is to construct a binary classification. Figure 6(A) shows that there are many possible hyperplanes which correctly separate the positives examples from the negative ones, but the support vector algorithm chooses the separating hyperplane which has the maximum margin. This linear classifier is also known as a maximum margin classifier, as shown in Figure 6(B).

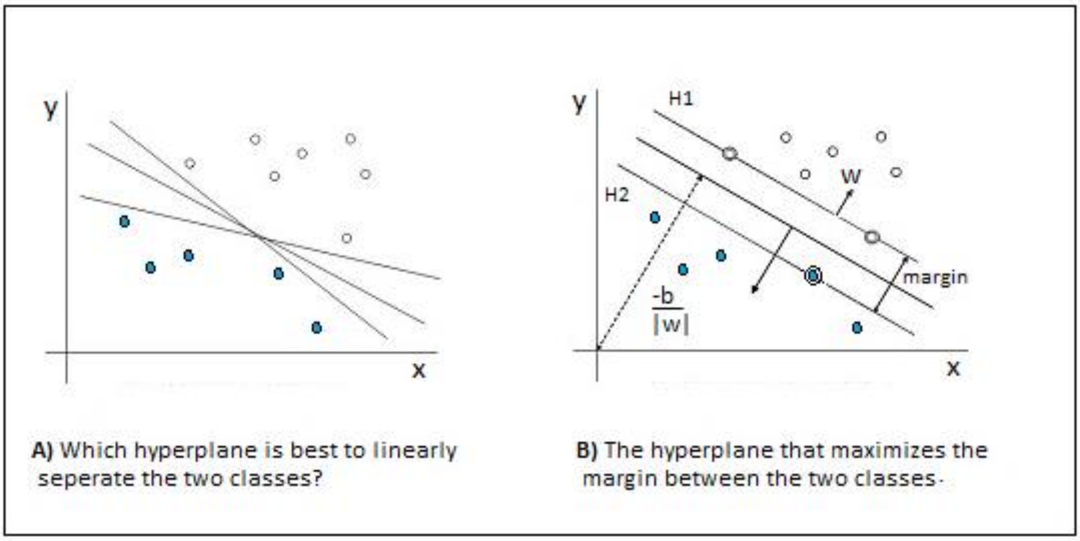
y
x
A) Which hyperplane is best to linearly seperate the two classes?
y
H1
H2
W
margin
-b
|w|
x
B) The hyperplane that maximizes the margin between the two classes.

Figure 6: SVM Linearly separable case and margins: A linear classifier is determined by a vector w and an offset b (Adopted from Burges 1998).

The data points x which fall on the hyperplane, meet the following equation:

$$x \cdot w - b = 0 \tag{1}$$

where:

Vector w is normal to the hyperplane and shows the orientation of a separating hyperplane.

Scalar *b* is the offset of the hyperplane from the origin.

$b / \| w \|$ is the perpendicular distance from the hyperplane to the origin.

$\| w \|$ is the Euclidean norm of *w*.

The goal is choosing *w* and *b* to maximize the distance between parallel hyperplanes.

In Figure 6(B), the data points that lie closest to the separating hyperplane are called *support vectors* (shown in circles in the Figure 6(B)). The two planes H_1 and H_2 on which these points fall can be formulated by:

$$x \cdot w - b = +1 \qquad \text{for } H_1 \tag{2}$$

$$x \cdot w - b = -1 \qquad \text{for } H_2 \tag{3}$$

We suppose that all data points can be described as follows:

$$x_i \cdot w + b \geq +1 \qquad \text{for } y_i = +1 \tag{4}$$

$$x_i \cdot w + b \leq -1 \qquad \text{for } y_i = -1 \tag{5}$$

These can be written as one set of inequalities:

$$y_i(x_i \, . \, w + b) \geq 1, \quad \forall i \tag{6}$$

Simple vector geometry shows that the distance or margin is equal to $1/\| w \|$ and maximizing the margin corresponds to finding:

$$\min \| w \| \tag{7}$$

subject to:

$$y_i(x_i \, . \, w + b) \geq 1, \quad \forall i$$

The presented optimization problem depends on $\| w \|$. It is possible to substitute $\| w \|$ with $\frac{1}{2}\| w \|^2$, the use of this term makes it possible to perform quadratic programming (QP) optimization later on. So we need to find

$$\min \frac{1}{2}\| w \|^2 \tag{8}$$

subject to:

$$y_i(x_i \, . \, w + b) \geq 1, \quad \forall i$$

To solve the above equation, a *Lagrangian formulation* of the problem (Eq.8) is considered and there are two reasons for this consideration. The first reason is that the Lagrangian constraints replace the constraint in (8), which is much softer to solve. The other reason is that we will have only the training data in new formulation where there are dot products between vectors. This important feature enables generalizing the procedure in the non linear case. Therefore, non-negative Lagrange multipliers α_i, $i = 1, \ldots, l$ are used.

Thus, the previous problem (8) is formulated as follows:

$$L = \frac{1}{2}\| w \|^2 - \sum_{i=1}^{l} \alpha_i y_i (x_i \cdot w + b) + \sum_{i=1}^{l} \alpha_i \qquad (9)$$

We must minimize (9) with respect to *w, b*, and maximize it with respect to α_i, for all constraints $\alpha_i \geq 0$. This can be done by differentiating L with respect to *w* and *b* and setting the derivatives to zero:

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{l} \alpha_i y_i x_i \qquad (10)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{l} \alpha_i y_i = 0 \qquad (11)$$

Substituting (10, 11) into (9) leads to the following formulation which is referred to as dual quadratic optimization problem:

$$L = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j x_i \cdot x_j \qquad (12)$$

subject to:

$$\sum_{i=1}^{l} \alpha_i y_i = 0, \quad \alpha_i \geq 0, \quad \forall i$$

The goal is to maximize *L*. In the solution each α_i with condition $\alpha_i \geq 0$, indicates that the corresponding x_i is a support vector. The support vectors fall on the hyperplane H_1 or H_2, or on that side of H_1 or H_2, (according to Eq.6) (see Figure 6).

Then the decision function will be expressed as:

$$f(x) = \sum_{i=1}^{l} \alpha_i y_i x x_i + b \qquad (13)$$

1.6.3. Linearly non-separable case

In the linearly non-separable case, the standard maximum margin algorithm described in the previous section is not able to find any separating hyperplanes to split positive and negative examples. Figure 7 illustrates the linearly non-separable case. This case happens because of the existence of noise and mislabeled examples, or because the kernel function is not appropriate for the training data. So more sophisticated techniques are necessary. This problem can be solved by using a *soft margin* method that is able to take some misclassifications of the training examples. Soft margin relaxes the margin constraints by pushing some data points into another side of the hyperplane without affecting the final result. This method include slack variables, ξ_i , which make having misclassification or noisy examples possible and estimate the degree of misclassification of the data x_i .

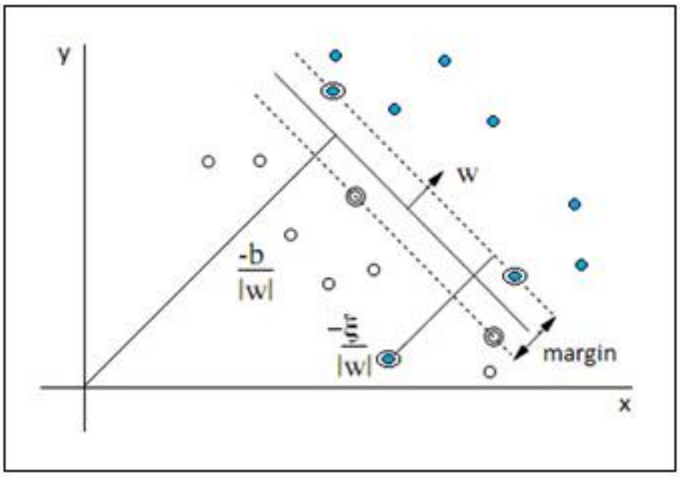

Figure 7: SVM linearly non-separable case and margins (Adopted from Burges 1998).

The new formulation including slack variables will be:

$$x_i \, . \, w + b \geq +1 - \xi_i \qquad \text{for } y_i = +1 \tag{14}$$

$$x_i \, . \, w + b \leq -1 + \xi_i \qquad \text{for } y_i = -1 \tag{15}$$

$$\xi_i \geq 0 \qquad \forall i \tag{16}$$

These can be written as one set of inequalities:

$$y_i (x_i \cdot w + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i \tag{17}$$

If the error function is linear, the optimization problem becomes:

$$\min_{w,b,\xi} \qquad L = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \qquad (18)$$

subject to:

$$y_i(x_i \cdot w + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i$$

The regularization parameter $C > 0$ is used for determining the tradeoff between the complexity term and training error (Burges 1998). As in the linearly separable case, this optimization problem can be converted to its dual problem:

$$\max_{\alpha} \qquad L = \sum_{i=1}^{l}\alpha_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j y_i y_j x_i \cdot x_j \qquad (19)$$

subject to:

$$0 \leq \alpha_i \leq C, \quad \text{for all } \alpha_i$$

$$\sum_{i=1}^{l}\alpha_i y_i = 0$$

where *l* is the number of support vectors. The only difference between this case and the case discussed before is that in this case the size of the Lagrange multiplier is limited by *C*.

Then the decision function will be expressed as:

$$f(x) = \sum_{i=1}^{l}\alpha_i y_i x x_i + b \qquad (20)$$

This situation is summarized in Figure 7.

1.6.4. Nonlinear support vector machines

There are cases where training datasets are nonlinearly separable, and decision function is not a linear function of the data. In 1992, B.E. Boser, I. Guyon, and V.N. Vapnik introduced a way to create nonlinear classifiers. For this purpose they used kernel trick to maximum margin hyperplanes (Boser et al. 1992). By applying kernel trick, the feature vector of all training samples are first transferred to high dimensional feature space. An example of the effect of mapping the two-dimensional nonlinear input space into a three-dimensional linear feature space is demonstrated in Figure 8. The linear hyperplane is searched in this new feature space. If the selected nonlinear mapping is appropriate enough, then a hyperplane separate the data in the feature space. The kernel trick just chooses an appropriate function corresponding to dot product of some nonlinear mapping. Some of the most commonly chosen kernel functions will be shown in the last part of this section.

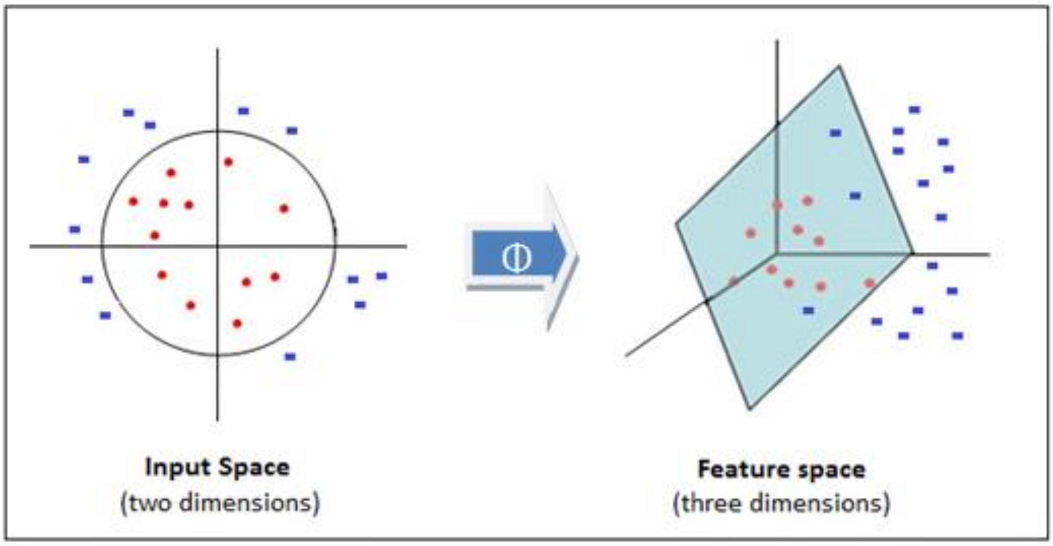

Figure 8: Mapping of SVM into higher dimension feature space
(Adopted from Muller et al. 2001)

The data can become visible in the training problem in the form of dot products, $x_i \cdot x_j$, Eqs. (19). By mapping each data point into high dimensional space through transformation Φ: x → φ(x), the dot product becomes:

$$K(x_i, x_j) = \varphi(x_i)\varphi(x_j) \tag{21}$$

The function K(x, y) is called the *kernel function*. A kernel function is some function that corresponds to an inner product in some expanded feature space. The use of a kernel function allows the SVM to operate efficiently in a nonlinear high-dimensional feature space without being adversely affected by the dimensionality of that space. Indeed, it is possible to work with feature spaces of infinite dimension.

The new formulation using kernel function K(x, y) will be:

Find $\alpha_1 ... \alpha_l$ to maximize

$$L = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \qquad (22)$$

subject to:

$$\sum_{i=1}^{l} \alpha_i y_i = 0$$

$$0 \le \alpha_i \le C, \quad \forall i$$

Then the decision function will be expressed as:

$$f(x) = \sum_{i=1}^{l} \alpha_i y_i K(x_i, x_j) + b \qquad (23)$$

Optimization techniques for finding α_i 's are the same as before.

A number of learning machines can be constructed by applying different kernel functions. A kernel function must fulfill some specific conditions (Mercer conditions). These conditions are explained in (Spiegelhalter et al. 1994). It is important to notice that the SVM's performance will be improved by selecting the suitable kernel function based on the specific problem. Some common kernel functions are listed in Table 1.

Table 1: Common kernel functions

Kernel type	**Formula**
Linear kernel (Identity kernel)	$K(x, x') = x^T x'$
Polynomial kernel function with degree *d*	$K(x, x') = (x^T x' + 1)^d$
Radial basis kernel function with width σ (RBF)	$K(x, x') = \exp(\frac{-\| x - x' \|}{2\sigma^2})$
Sigmoid kernel function with parameters β_0 and β_1	$K(x, x') = \tanh(\beta_0 x^T x' + \beta_1)$

1.6.5. One-class support vector machine

The SVM algorithm introduced by Vapnik is basically a two-class algorithm as there are both negative and positive examples. However, there are some cases that are difficult or impossible to obtain examples belonging to one specific class (Manevitz and Yousef 2001). The one-class support vector machine (OCSVM) was introduced by Schölkopf et al. (1999). They proposed a method of adapting the support vector classification methodology to the problem of one-class classification. OCSVM can be considered as a binary SVM where the first class contains all the training data and the origin is treated as the only member of the second class. Figure 9 illustrates the OCSVM classifier. We give a brief introduction to the basic concepts of OCSVM in this section. The description in detail can be found in (Schölkopf et al. 2001; De Bie et al. 2007).

Like the traditional (basic) SVM, the OCSVM algorithm maps input data into a high dimensional feature space through transformation Φ: x→ φ(x) and via an appropriate kernel function $k(x,x^{'})$.

$$K(x_i, x_j) = \varphi(x_i)\varphi(x_j) \tag{24}$$

In practice, the most widely used kernel function in OCSVM is the radial basis function (RBF) (Schölkopf and Smola 2002). OCSVM attempts to find the maximal margin hyperplane that best separates the mapped vectors (training data) from the origin. In the case that there is no such hyperplane, slack variables , ξ_i, make having misclassification examples possible and allow for some points to be lied within the margin.

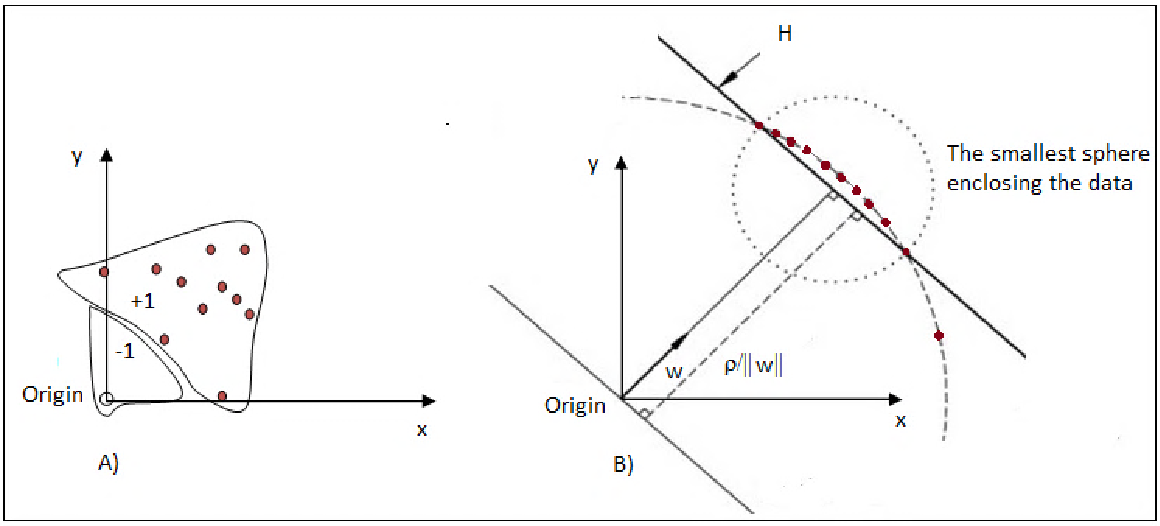

Figure 9: One-class SVM. A) Separating data from the origin. B) The data are mapped onto the hyperplane by RBF kernel (Adopted from Wu and Chung 2009)

As presented in (Schölkopf 1999), one needs to solve the following QP problem to separate input data from the origin:

$$\min_{w,\xi,\rho} \frac{1}{2}\|w\|^{2} + \frac{1}{\nu n}\sum_{i=1}^{l}\xi_i - \rho \tag{25}$$

subject to

$$(w,\varphi(x_i)) \geq \rho - \xi_i, \qquad \xi_i \geq 0, \quad \forall i$$

where w is a normal vector to the hyperplane and shows the orientation of a separating hyperplane, ξ_i is slack variable, ν is the regularization parameter used for controlling the tradeoff between the complexity term and training error and it varies from 0 to 1, $\frac{\rho}{\|w\|}$ is the margin (distance from the hyperplane), and n is the number of points in the training dataset.

The solution w and ρ of Eq.25 form the following decision function:

$$f(x) = sign((w.\varphi(x)) - \rho) \tag{26}$$

Applying Lagrange multipliers α_i for each vector x_i, the dual form of the primal OCSVM problem is formulated as follows:

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i\alpha_j k(x_i\ , x_j) \tag{27}$$

subject to:

$$0 \leq \alpha_i \leq \frac{1}{\nu n},$$

and

$$\sum_{i=1}^{l}\alpha_i = 1$$

1.7. Multiple Kernel Learning

The basic idea behind kernel methods is transforming the input feature space where the classes are linearly separable (Vapnik 1998). Classical learning algorithms are based on a single kernel, but in practice it is often desirable to apply multiple kernels when learning problems contain multiple, heterogeneous data sources. Thanks to recent progress on SVMs and other kernel methods, multiple kernel learning (MKL) methods have been proposed and successfully applied on real world datasets (Gönen and Alpaydin 2011) . MKL provides more flexibility by allowing to combine different kernels (representations) each corresponding to a different feature set, instead of selecting one specific kernel function.

There are several papers focusing on efficient methods to solve multiple kernel learning (Bennett et al., 2002; Bach et al., 2004; Bi et al., 2004; Lanckriet et al., 2004). A basic approach to combine different kernel matrices is to consider a convex linear combinations of m kernels, i.e.

$$K(x_i, x_j) = \sum_{k=1}^{m} w_k K_k(x_i, x_j), \qquad \forall i, j \tag{28}$$

subject to:

$w_k > 0$

and

$\sum_{k=1}^{m} w_k = 1$

where $w_k$is the weight of the kernel *k*. Different kernels can be assigned different weights according to their importance in the linear combination. There are different ways to tune the kernel weights, see, e.g., (Aerts et.al 2006).

1.8. Data integration

Several methods have been used for integrating multiple biological data sources with kernel methods. Pavlidis et al. (2002) used a combination of two fixed length data sources, microarray gene expression data and phylogenetic profiles, to train an SVM to recognize functional categories of yeast genes. To learn from combination of these two data sources, they applied three different methods. Their techniques differ in the stage of integration process and occur at three different levels. They are referred to as early, intermediate and late integration (Pavlidis et al. 2002). An overview of these three techniques for two datasets is shown in Figure 10. Early integration or full integration occurs at data base level. This means that before applying any algorithm and building an SVM model the two datasets are integrated to make a single dataset. This scheme has the advantage of being rather easy to implement when the underlying data structure allows such integration. Intermediate integration or the partial integration approach considers datasets as separate entities and datasets are combined during the learning process at the kernel level without referring back to the data. First, a kernel matrix is created for each dataset by applying an appropriate kernel function and relationships between variables within a dataset are taken into account while relationships between different types of variables are not considered. Then, different kernel matrices are added via a multiple kernel learning technique (describe in section 1.7), that is $K = \sum_{k=1}^{m} w_k K_k$. The last technique is integration at the knowledge level and is referred to late integration or the decision integration. In this approach, models are created on each dataset separately (one SVM is trained on each dataset), a decision function is derived from each training, then their obtained discriminant scores are summed. After comparing results, the best accuracy was achieved by using intermediate integration of datasets (Pavlidis et al. 2002, Schölkopf et al. 2004).

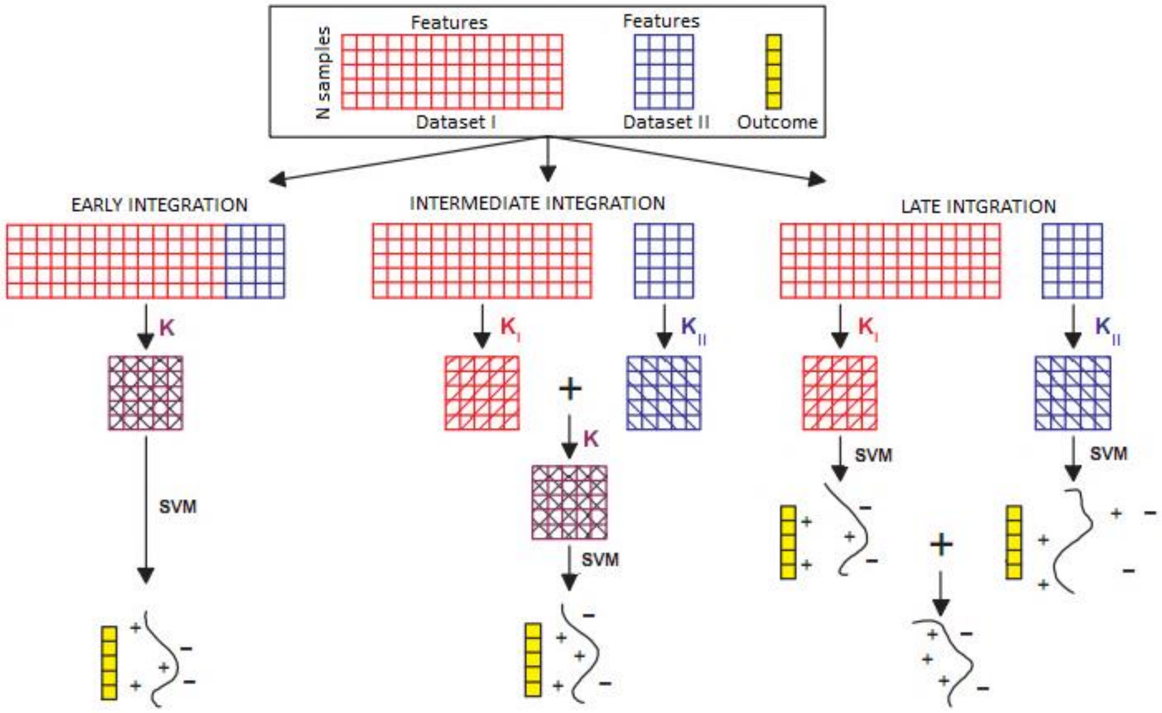

Figure 10: Three methods for learning from multiple datasets. In early integration, the two datasets are combined to make a single dataset. In intermediate integration, one kernel is computed for each dataset and then kernels are summed. In late integration, one SVM is trained on each dataset, and the obtained discriminant scores are added (From Daemen et al. 2009).

1.9. Gene prioritization

The identification of genes containing a particular biological function is an ongoing research issue in biology. There are often long lists of candidate genes and researchers must prioritize the genes from most to least promising. Gene prioritization is a predictive method based on existing knowledge regarding the genes known to be associated to a particular disease, biological function or expression pattern in different conditions and raw data. Gene prioritization identifies and ranks the large amount of candidate genes from most to least promising based on their similarity with respect to genes with known biological function.

In this dissertation gene prioritization can be defined as: Given cell cycle regulated genes (training set) and unlabeled candidate genes (test set), enter these data to the computational method and it will give a score for each of candidate genes. Higher scoring genes are supposed to be the genes that are most likely cell cycle regulated and least scoring genes are the ones that are less likely cell cycle regulated.

To date, several computational methods have focused on the prioritization of candidate genes. In the following, we describe two main categories: classification methods and novelty detection methods.

1.9.1. Classification methods

The first step in classification is training. Training instances (genes) are marked as belonging to one of the multiple distinct classes. In the next step, the candidate examples are assigned into the different classes based on their features. In binary classification, two classes are defined as the positive class containing genes known to be relevant to the process under investigation and the negative class containing genes known not to be relevant in that process. Some classification methods also give a score to each candidate gene that makes the method to be used appropriately for gene prioritization (Adie et al. 2005, López-Bigas and Ouzounis 2004). Applying these methods, we face the challenge that it is often difficult or impossible to obtain examples of one class since we do not have enough knowledge to ensure about our assessment (Calvo et al. 2007).

1.9.2. Novelty detection methods

Novelty detection methods recognize new or unknown examples on which a learning system has not been trained and do not have previous knowledge (Markou and Singh 2003). During the last decade, Novelty detection techniques have gained wide popularity and have attracted a great deal of attention (Aerts et al. 2006; Gardner 2006; Rossi et al. 2006; Zhang et al. 2006; Gaulton et al. 2007; Ma et al. 2007; Perez-Iratxeta et al. 2007). In bioinformatics, novelty detection is used for a learning system where it is very difficult to guarantee that a gene is not involved in a biological process (there is no reliable negative training set) and the learning system is trained by using a positive training set. This positive set is often composed of genes that are known to be related to a specific process or disease under investigation. This knowledge can also be derived from a vocabulary of terms describing gene characteristics and gene annotations, such as the GO consortium (Ashburner et al. 2000). In the next step, the rank of candidate genes is computed according to their similarities to the genes with known functions. For novelty detection, several strategies have been explained in the literatures, such as Hidden Markov model (Yeung et al. 2002), Parzen window density estimator (Yeung et al. 2002), KNN-based approach (Guttormsson et al.1999), one-class support vector machine

(OCSVM) (Schölkopf et al. 1999), and etc. Among them, OCSVM is the mostly used method in machine learning (Chen et al. 2011). For gene prioritization, this dissertation uses the application of one-class support vector machine novelty detection: finds a hyperplane that separates the positive data (genes that are known to be cell cycle regulated) from the origin with the largest possible margin and consider a gene more likely to be cell cycle regulated if is located farther in the direction of the hyperplane.

2. Problem description

The main idea of this project is to use machine learning techniques to identify cell cycle regulated genes and rank the set of candidate genes; the genes that are likely to be cell cycle regulated should end up in the top of the ranking list whereas the genes that are not cell cycle regulated should be ranked in the bottom of the list. To do this, we use gene expression profiles that have been produced by four gene expression experiments. Different methods have been used to synchronize the cell cultures in these experiments. For example double thymidine block halts cells in S phase whereas treatment with nocodazole halts cells in M phase. Then, it is known that the gene expression profiles have been measured at different phases of the cell cycle. It is predicted that genes with consistent expression profiles within the starting point of the cell cycle in all series are more likely to have cell cycle function compared to those with inconsistent expression profiles. A gene up regulated during S Phase, such as CCNE2, should therefore be up regulated at the start of a thymidine series, and down regulated at the start of a nocodazaole series. We are mainly interested in genes having consistent cyclic expression patterns across all experiments. A PLS analysis is available for all time series used in this thesis. As mentioned before, the PLS model can take a given time series dataset from the microarray hybridization experiment and identify whether the genes' expression profiles show cyclic behavior with the same periodicity as the cell cycle within the synchronized cells. According to PLS analysis, some genes are identified as having significant cyclic patterns in all four series but some are found to be significant in only 2 or 3 out of 4 series.

Our main strategy is using knowledge regarding how different time series have been created, PLS analysis information and other genes' attributes to derive features that lead to the identification of genes with consistent expression profiles across multiple synchronization experiments. Then, we use obtained features as input to rank the set of candidate genes with respect to having cell cycle function.

3. Materials and methods

In this part, first we describe the data sets that are to be ranked, and the techniques that are used to combine different datasets. Then, we present the strategy selected to solve the prioritization problem. Also the methods and feature sets that are selected and applied to reach the proposed approach are explained.

3.1. Dataset Used

In our classification model, four different gene expression experiments (datasets) are used, namely JCC, BJ, TT, and NS. Each experiment has measured gene expression values at different time points after synchronization. These are:

- JCC consists of HaCaT cells synchronized at S phase of the cell cycle by using thymidine.
- BJ consists of Foreskin Fibroblasts synchronized at G1 phase of the cell cycle by using serum starvation.
- TT consists of HeLa cells synchronized at S phase of the cell cycle by using thymidine.
- NS consists of HeLa cells synchronized at M phase of the cell cycle when cells start to divide by using nocodazole.

Three different microarray technologies have been used for the four different experiments. TT and NS (HeLa cells) are both based on the Illumina microarray platform, while JCC and BJ are based on two different versions of the Affymetrix microarray platform. A PLS model has been previously developed on each experiment. Thus, there are totally four different PLS models (datasets) namely PLS _JCC, PLS_BJ, PLS_TT, and PLS_NS.

3.2. Combine different datasets

To train the SVM, the datasets need to be combined across common genes in an appropriate way. In the following, we describe how common genes were identified, and how datasets were integrated into a reduced matrix for training the SVM.

As mentioned above, the datasets have been produced by applying different microarray technologies. Since each microarray technology uses specific probe ids, there is a set of probe ids from different microarray technologies corresponding to a particular gene. Probe ids are identical between the two HeLa series which both are based on the Illumina microarray platform, and differ between the Foreskin Fibroblasts and HaCaT series, which used the Affymetrix microarray platform. For instance, the gene named “HMMR” corresponds to probe id "4060064" in the Illumina and probe id "207165_at" in the Affymetrix microarray platform. Thus, a particular gene is represented by different symbols (probes). To integrate the common data among four different expression experiments and make a common matrix, what we need is to have a common representing symbol for each gene. Since the gene names are always the same, what we did is to perform the analysis at the gene level instead of the probe level.

Moreover, each gene can be measured by one or multiple probes in a particular microarray technology. This means that for some genes there exists only one probe and for some genes several probes. In these cases, we chose the most significant probe according to the following strategy:

First probe ids are translated to gene names in each PLS model. Second, repetitive genes are compared according to q-values. *Q*-value is the minimum false discovery rate at which the result may be called significant. Third, duplicates are eliminated based on the amount of *q*-values, that is, in different particular models only the most significant genes are selected and others are removed. The most significant genes are the ones whose q-values are below 0.05.

Another challenge we faced was the cases where one gene is measured by one microarray technology while not measured by other technologies, so there might be some mismatches in expression matrix size. To overcome this, we eliminated those genes and did not consider them in the expression matrix.

3.3. Identification of consistent expression profiles

It is already known that different methods have been used to synchronize the cell cultures by halting the cell cycle at a particular phase. Serum starvation puts cells into quiescence G0/G1 phase. Double thymidine block halts cells in S phase whereas treatment with nocodazole halts cells in M phase. So each experiment has started to measure gene expression values at different phases after synchronization. Consequently, genes that are starting as up regulated in a thymidin series should probably not be up regulated at the start of a nocodazole and serum starvation series. A truly cell cycle regulated gene would have an expression profile that is consistent with each series starting point of the cell cycle. Then, genes with consistent expression profiles in all series are very likely to be cell cycle regulated, whereas genes with inconsistent expression profiles in all series are not likely to have cell cycle related function.

To find to which part of the cell cycle the time points correspond, we used the information about the percentage of cells assigned to the G1, S, and G2/M phases of the cell cycle for each time point. These cell percentages have been estimated by measuring the DNA content of cells in all series (Bar-Joseph et al. 2008; Peña-Diaz et al. 2013). For instance, if the majority of cells are in the S phase at a specific time point, it follows that this time point is corresponding to S phase (See Figure 11-14).

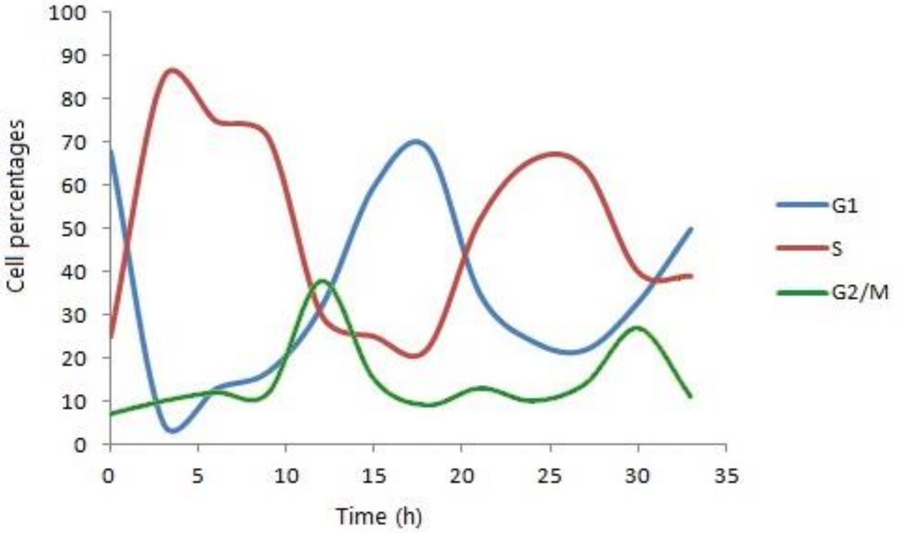

Figure 11: Percentage of cells assigned to the G1, S, and G2/M phases of the cell cycle for different time points in double thymidine blocked HaCaT cells (JCC).

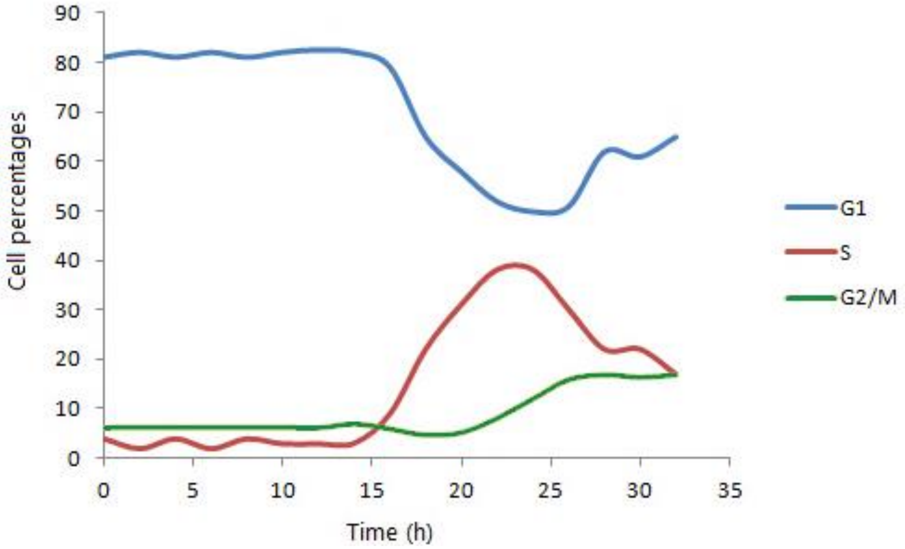

Figure 12: Percentage of cells assigned to the G1, S, and G2/M phases of the cell cycle for different time points in Foreskin Fibroblasts experiment synchronized by serum starvation (BJ).

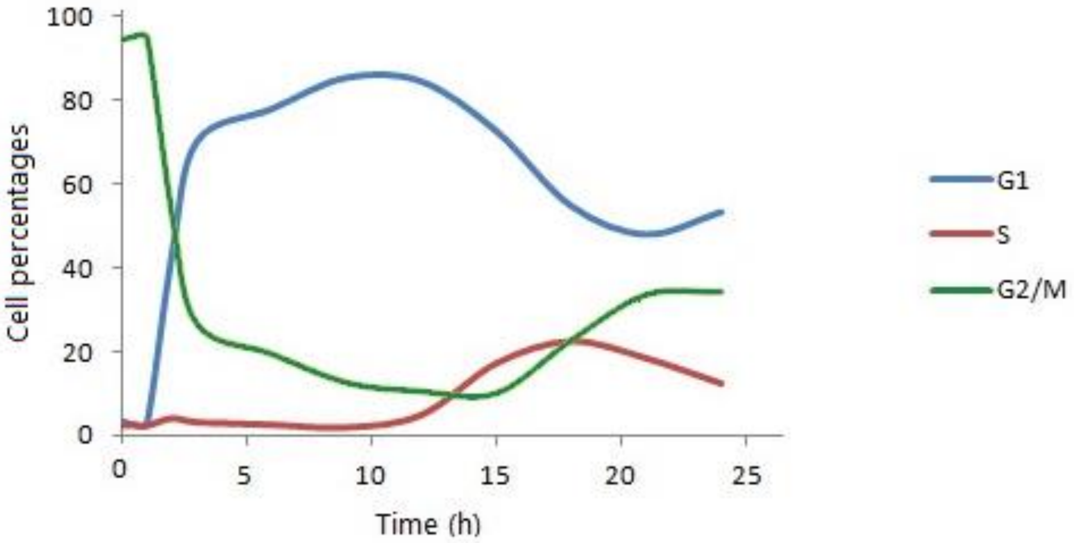

Figure 13: Percentage of cells assigned to the G1, S, and G2/M phases of the cell cycle for different time points in nocodazole blocked HeLa cells (NS).

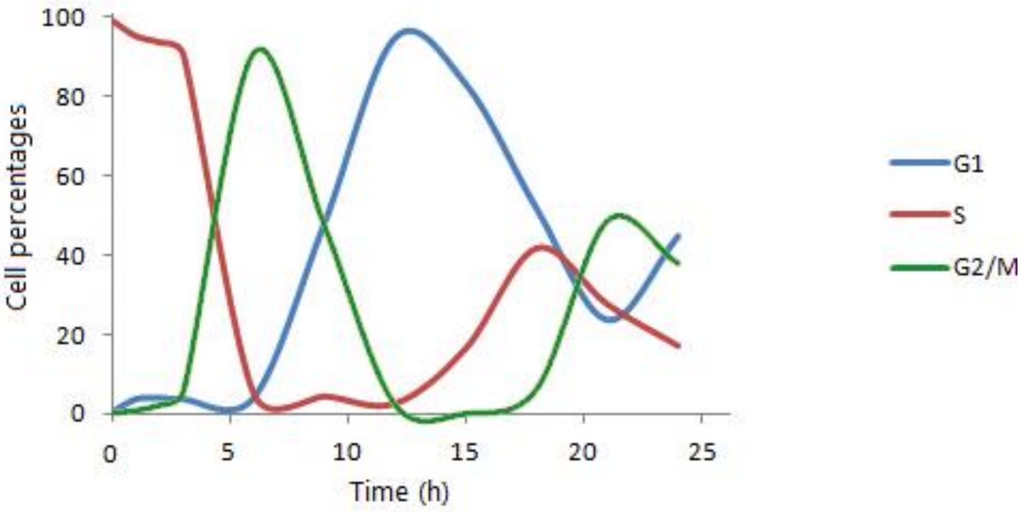

Figure 14: Percentage of cells assigned to the G1, S, and G2/M phases of the cell cycle for different time points in double thymidine blocked HeLa cells (TT).

An example of a gene with consistent expression profiles is described below:

Cyclin E2 (CCNE2) is a cell cycle regulated gene whose activity plays an important role for the cell cycle G1/S transition. The expression of CCNE2 peaks at the G1/S phase, so it is expected that CCNE2 is up regulated at the end of G1 phase and start of S phase.

We consider the expression profile of CCNE2 in four differentially synchronized experiments, HaCaT experiment synchronized by double thymidine block (JCC), HeLa experiment synchronized by double thymidine block (TT), HeLa experiment synchronized by nocodazole (NS) and Foreskin Fibroblasts experiment synchronized by serum starvation (BJ) (Figure 15a-15d respectively). In each experiment, we match the time points to corresponding G1, S, and G2/M cell cycle phases based on Figure 11-14. It is observed that in all four experiments gene CCNE2 is up regulated at the end of G1 phase and start of S phase and down regulated at the end of S phase and start of G2/M phase. So it is concluded that for some genes like CCNE2, there is high consistency between four differentially synchronized experiments.

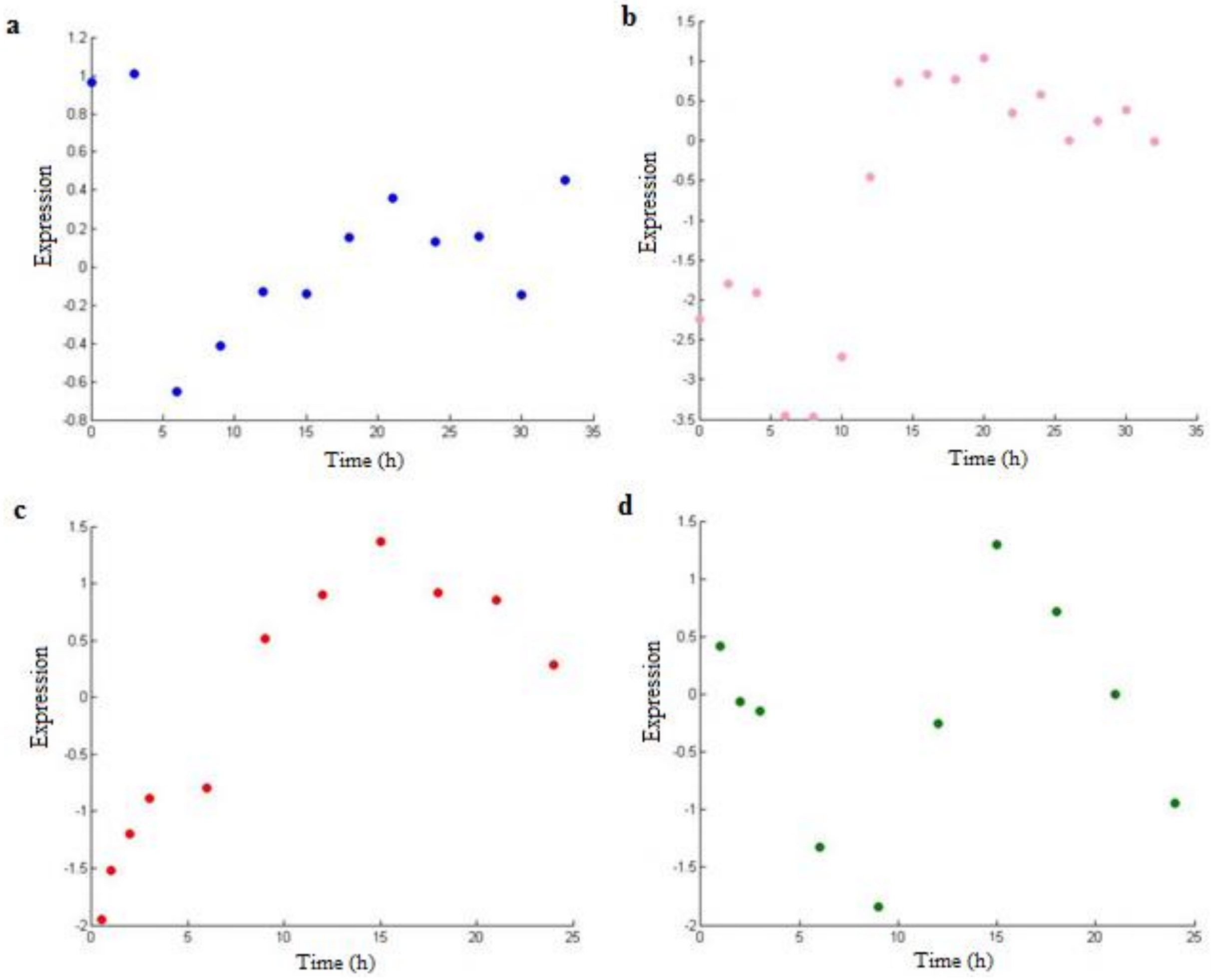

Figure 15: Time series expression profiles of CCNE2 in four differentially synchronized experiments. (a) CCNE2 expression profile in HaCaT experiment synchronized by double thymidine block (JCC); (b) CCNE2 expression profile in Foreskin Fibroblasts experiment synchronized by serum starvation (BJ). (c) CCNE2 expression profile in HeLa experiment synchronized by nocodazole (NS); (d) CCNE2 expression profile in HeLa experiment synchronized by double thymidine block (TT).

Similar to what was described for CCNE2 example, we can take the expression profiles for all the genes and use the time series features to determine whether the expression profile within a specific series is consistent with the expression profile within another series.

3.4. Construct feature sets to make OCSVM prioritization

In this section, three sets of features applied for training the OCSVM, and the approaches used to reach them are described.

3.4.1. The first set of features – Percentage of up regulated expression profiles

We assumed that calculating the percentage of up regulated expression profile for each gene would lead to identify genes with consistent cyclic expression profiles across multiple experiments**.** We used the following strategy to distinguish the up regulated expression profiles.

First, the average of all expression data in all time points was computed for all genes. The time points where the expression data of a particular gene is above the average were considered as up regulated. The time points where the expression data of the particular gene is below the average were considered as down regulated. To identify the cell cycle phases in which the genes are up regulated, we assigned the time points to their corresponding cell cycle phases (as discussed in section 3.3). Then we weighted them by the number of times a particular cell cycle phase has occurred. As an example, in the experiment synchronized by nocodazole (NS), gene expression values has been measured at 12 different time points in which 2 time points correspond to S phase, 6 time points correspond to G2/M phase, and 4 time points correspond to G1 phase. For the particular gene CCNE2 in the experiment synchronized by nocodazole (NS), first, we computed the average of expression data (av= -0.26) then, we compared each expression data with the average. It was found that CCNE2 is up regulated in 2 out of 2 S phases, 2 out of 4 G1 phases and 2 out of 6 G2/M phases (See Figure 13, Figure 15c and Table 2).

Table 2: An overview of identifying the cell cycle phases in which the gene CCNE2 is up regulated in HeLa experiment synchronized by nocodazole (NS).

Time (h)	0	0.5	1	2	3	6	9	12	15	18	21	24
Cell cycle phase	G2/M	G2/M	G2/M	G2/M	G1	G1	G1	G1	S	S	G2/M	G2/M
Expression data	-1.70	-1.95	-1.52	-1.2	-0.88	-0.79	0.51	0.90	1.36	0.92	0.85	0.28
Compare to av= -0.26	<	<	<	<	<	<	>	>	>	>	>	>
Up/Down (U/D)	D	D	D	D	D	D	U	U	U	U	U	U

Table 3: Percentage of up regulated expression profiles for the gene CCNE2 in HeLa experiment synchronized by nocodazole (NS).

Cell cycle phase	S	G1	G2/M
% of up regulated expression profile	100	50	33.3

After computing the percentages of up regulated expression profiles for all genes, each gene is represented as a vector with 3 property values (features) in each dataset. Since there are 4 datasets, we ended up with a matrix in which each vector contains 12 property values (3 properties for each dataset). Briefly, as a training set, we have a matrix with genes along the rows and cell cycle phases along the columns, and there is a percentage of up regulation corresponding to each row and column (Figure 16).

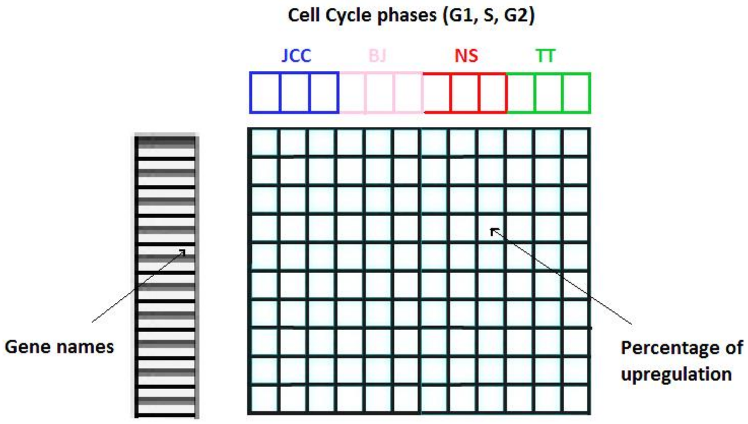

Figure 16: Illustration of the matrix created for the first set of features.

3.4.2. The second set of features – Assigning cell cycle phases

The following strategy was used to assign the cell cycle phases to each gene in each experiment. As it is shown in Figure 17(a), the location of each gene and each sample time point can be found in the PLS model first and second principle components. First, the arctangent of the gene’s first and second principle component in the PLS model was computed to assign a phase angle to each gene. Second, we computed the location of the samples’ time points within the PLS model by computing the arctangent of the sample’s first and second principle components. Then, we used the information about the percentage of cells assigned to the G1, S, and G2/M phases of the cell cycle for each time point to find to which part of the cell cycle the specific time point correspond (as discussed in section 3.3), and used the cell cycle phase assignment for time points. After getting the angles of samples in G1, G2 and S phases of the cell cycle, we manually subdivided the plot to cell cycle phases. For example, if the majority of samples in a region belong to a specific phase, that specific phase was considered for that region. Finally, we compared each gene's phase angle to the S, G2, and G1 regions of samples, and assigned the closest cell cycle phase to each gene.

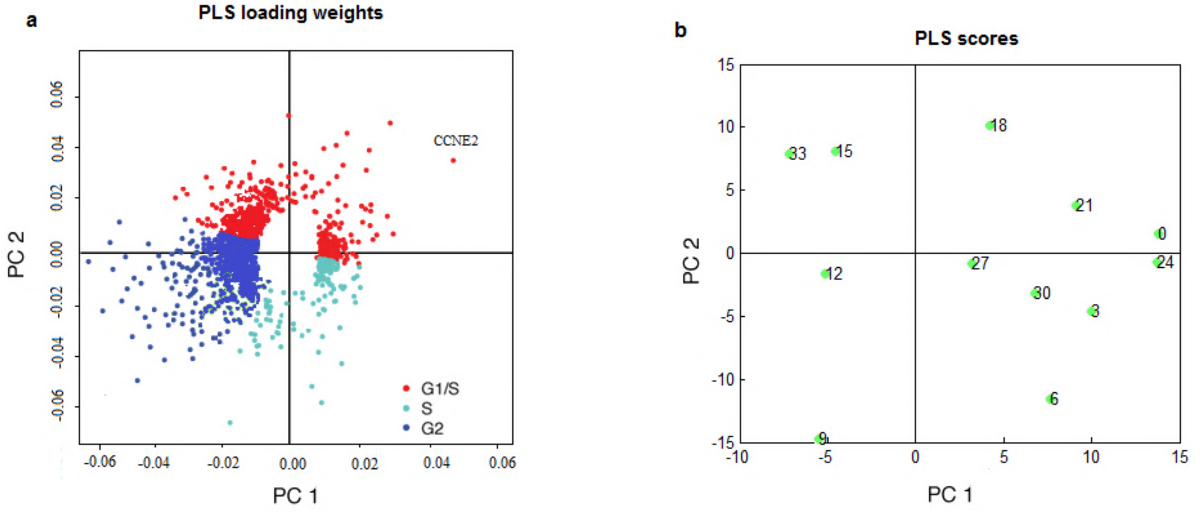

Figure 17: The loadings (a) and scores (b) plots for PLS model of the gene expression profiles from double thymidine blocked HaCaT cells (JCC). (a) Points show genes and their location within the PLS model's first and second components (PC1 and PC2). Colors show the cell cycle phase where the gene is predominantly expressed. (b) Points show the location of the samples time points within the PLS model's first and second components.

After assigning the cell cycle phase to each gene in all experiments, each gene is represented as a vector with 3 features in each dataset. Since there are 4 datasets, we ended up with a matrix in which each gene contains 12 property values (3 properties for each dataset). (Figure 18).

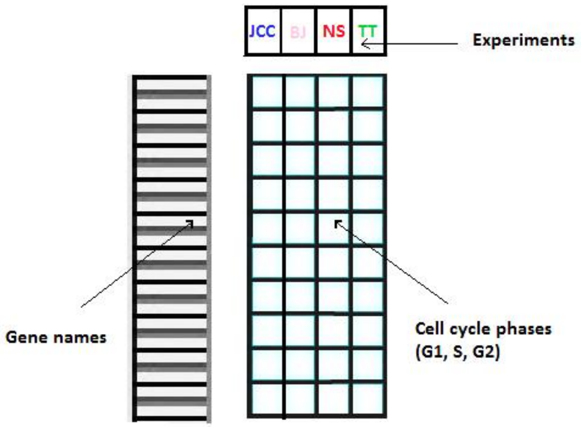

Figure 18: Illustration of the matrix created for the second set of features.

3.4.3. The third set of features – Gene expression in different tissue type

We added an additional feature, which does not depend on the characteristics of the time series expression profiles. This feature shows whether the genes are significantly up regulated or down regulated under different experimental conditions (tissue types). We specifically focused on 3 conditions namely brain, muscle and cell line. The reason for this selection lies behind the fact that the brain is a fairly stable tissue and once the brain cells are fully formed, they do not undergo cell division (Herrup and Yang 2007); also, the muscle is to some extent a stable tissue (Partridge 2002), therefore it is expected that genes involved in regulating the cell cycle would be down regulated in the brain and muscle. In contrast, the cell lines basically are proliferating continuously in culture (Browne and Al-Rubeai 2009), and we expect that genes that are involved in regulating the cell cycle would be up regulated in the cell lines.

To determine how the genes are expressed under three different conditions, we used the ArrayExpress repository (accession number: E-MTAB-62) (Kapushesky et al. 2010). To compare the significance of the differential expression, a t-statistic value was computed for each gene per condition. The t-statistic is a value, which is created by the statistical tests, and it is used to identify if a gene is differentially expressed in a given condition. A positive value indicates that the gene is over expressed, while a negative value shows that the gene is under expressed compared to the average expression of that gene over all conditions.

Finally, we ended up with a matrix in which genes are along the rows and the 3 different conditions (brain, muscle and cell line) are along the columns, and there is a t-statistic value corresponding to each row and column (Figure 19).

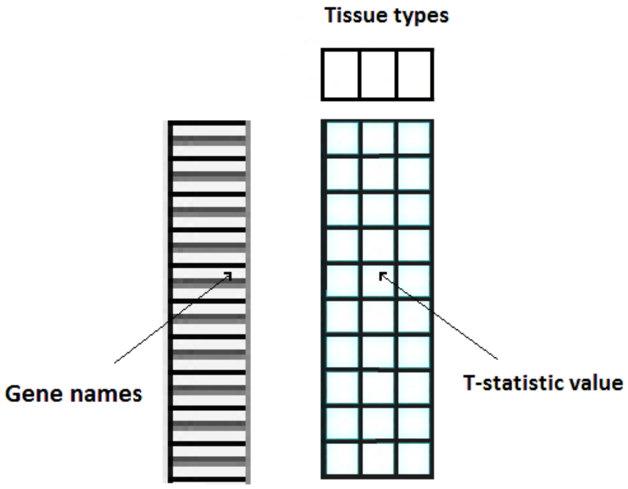

Figure 19: Illustration of the matrix created for the third set of features.

3.5. Model building for prioritization of cell cycle regulated genes

We defined the cell cycle regulated gene prioritization problem as a novelty detection problem and formulated it in machine learning terms. We used 4 datasets described in section 3.1. For the positive training set, we selected the genes identified by PLS analysis to have significant cyclic expression patterns across all four datasets, and have known cell cycle function. For the unlabeled set, genes that are significant in only 2 out of 4 datasets and have unknown cell cycle function were selected. To find whether genes are cell cycle regulated, the term "Cell Cycle" was searched by gene ontology (GO) annotations of all genes by using library "GO.db" that is available as an R package from the Bioconductor project: http://bioconductor.org.

To construct the positive training set and the unlabeled set that is supposed to be ranked, the following steps were applied.

1. Combine expression profile datasets of common genes derived from 4 experiments on which PLS models have been developed (JCC, BJ, NS, and TT).
2. Extract genes that are significant in all 4 datasets (PLS models) and have known cell cycle functions and consider them as the positive training set.

3. Extract genes that are significant in 2 out of 4 datasets and have unknown cell cycle functions and consider them as the unlabeled set to be ranked.
4. Compute percentage of up regulated expression profiles for extracted genes in step 2 and 3, and consider them as the first set of features.
5. Compute cell cycle phases for extracted genes in step 2 and 3, and consider them as the third set of features.
6. Extract t-statistic values in the 3 different conditions-brain, muscle and cell line- for extracted genes in step 2 and 3, and consider them as the third set of features.

Since datasets containing first, second and third feature sets have to be of the same length to be integrated, in the cases where no t-statistic value was found for genes in one dataset, those genes were removed from another datasets as well.

After creating datasets, a kernel based method was applied to solve the problem of prioritizing cell cycle regulated genes. For this purpose, after feature selection, all data sources were transformed into kernels using an RBF kernel function. We set RBF kernel width parameter (σ) to the average distance of a data point to its nearest neighbor in the union of the training and the test set. Then, for performing novelty detection, an OCSVM algorithm, for which only the positive training genes are trained, was used. To perform OCSVM, we used LIBSVM implementation which is an open source software package for SVMs (Chang and Lin 2001).

For training, the approach proposed in this dissertation finds a hyperplane that separates the positive data from the origin. Since we used three different sets of features, there is one dataset per feature set. To integrate the datasets, three integration techniques, early, intermediate and late integration, were applied. To combine the kernels in the intermediate integration, I used the approach of uniformly weighted kernels in which first the kernels are weighted equally, and then they are summed (K= $\frac{1}{2}K_1 + \frac{1}{2}K_2$).

To compare the performance of different features, we plotted a receiver operating characteristic (ROC) curve to compute the area under the curve (AUC), which varies between 0.0 and 1.0. AUC measures the probability that a predictive model will rank a randomly selected positive sample higher than a randomly selected negative one. An ROC curve is a two dimensional graph in which false positive rate (FPR) is drawn on the X-axis and true positive rate (TPR) is shown on the Y-axis. TPR, also known as sensitivity (SN), is the proportion of true positives (TPs), which are correctly determined as positive. FPR is the fraction of false positives (FPs) out of the negatives and equals to one minus the specificity (SP).

To estimate the performance of used kernel functions, we employed a LOOCV strategy in which a single cell cycle regulated gene was reserved as the validation data. For each LOOCV iteration, one of the training genes (known cell cycle regulated genes) was singled out while the remaining training genes were used for the training. The singled out gene and randomly selected unlabeled genes were put together to make the candidate set of a fixed size. Then, the genes were scored and a ranking number was assigned to each of the candidate genes and the singled out gene. For scoring, the distance between the hyperplane and the projection of the candidate gene profile along the direction of the hyperplane was used. If the algorithm performs perfectly, it is expected that the singled out gene is ranked among the first ones because there is already a known indication that it is a cell cycle regulated gene. This procedure was repeated so that every known cell cycle regulated gene in the training set was, in turn, singled out. Our schematic LOOCV process is shown in Figure 20. In the next step, our cross validation process and those rankings were followed by an ROC analysis. The area under ROC curve (AUC) was used as an indicator for the prioritization performance.

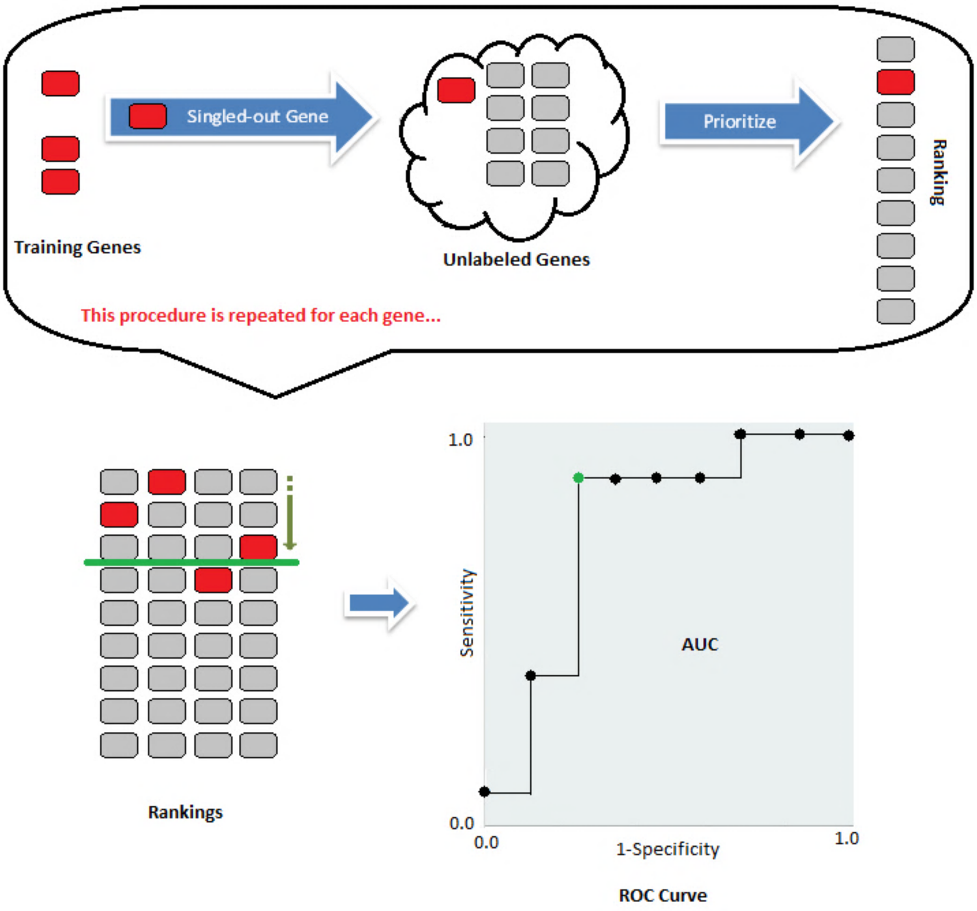

Figure 20: An overview of the LOOCV procedure. A repetition of prioritization runs is performed in the validation. In each validation run, one gene from the set of training genes (red boxes) is singled-out and combined to candidate genes (gray boxes). The remaining training genes are used for training, and the model determines the ranking of all the candidate genes including the singled-out gene after prioritization. This step is repeated so that all the training genes are, in turn, singled-out. In the next step, these rankings are used to create a ROC curve. Considering a threshold on the matrix (green line) makes it possible to define a binary classifier, the associated sensitivity and specificity, and therefore to draw a point in the ROC space. Changing the threshold along the matrix provides us to determine a complete ROC curve and it's AUC.

4. Results and discussion:

By selecting the genes that are common and significant in all 4 datasets, we ended up with 103 genes among which 66 genes are known as cell cycle regulated (as determined by GO). These 66 genes were considered as the positive training set for OCSVM.

For selecting candidate sets to be ranked (unlabeled sets), we used genes that have no known cell cycle function (as determined by GO), and are common in all the 4 experiments, however significant only in 2 out of 4 datasets. We mainly focused on two candidate sets as follows:

Candidate set I consists of genes that are only significant in datasets NS and TT (230 genes). Candidate set II consists of genes that are only significant in datasets JCC and BJ (191 genes).

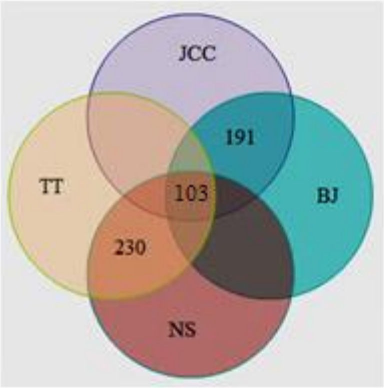

Figure 21: Venn diagram showing the overlap in significant genes between four datasets.

First, significant genes that are common in datasets NS and TT were used. This data set consists of 296 genes out of which, 66 genes made the positive training set and 230 genes made the unlabeled set (candidate set II). The following results were achieved by employing RBF-OCSVM through LOOCV.

- AUC of 0.89 for the first set of attributes, percentage of up regulated expression profiles.
- AUC of 0.51 for the second set of attributes, cell cycle phases.
- AUC of 0.83 for the third set of attributes, gene expression in different tissue types.

According to AUC values, the performance of the proposed approach was significant considering the first and third feature sets, while OCSVM performed poorly by using the second feature set. Plots of the above results in the ROC space are illustrated in Figure 22. In the next step, the first and third feature sets leading to a high performance were combined through three different integration methods, early, intermediate and late integration. We achieved AUC of 0.83 by applying early integration, and AUC of 0.92 by applying intermediate and late integration. Comparing the results indicates that both the intermediate and late integration outperform early integration (Figure 23).

The list of genes in candidate set I and their assigned ranks are shown in Table C.1 in Appendix C.

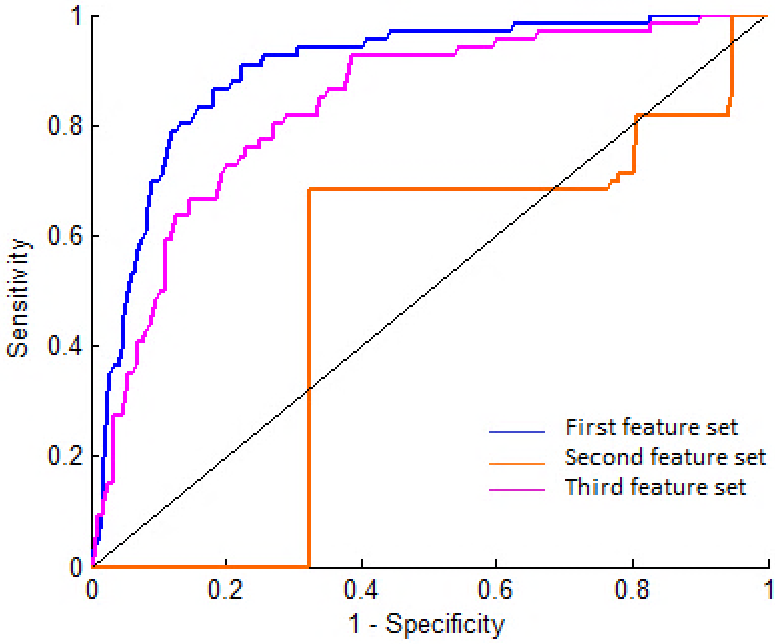

Figure 22: ROC graphs for the three feature sets used for significant genes in datasets NS and TT. ROC curve for the first feature set, percentage of up regulated expression profiles, in blue, shows an AUC of 0.89. ROC curve for the second feature set, cell cycle phases, in orange, results in an AUC of 0.51. ROC curve for the third feature set, gene expression in different tissue types, in red, results in an AUC of 0.83.

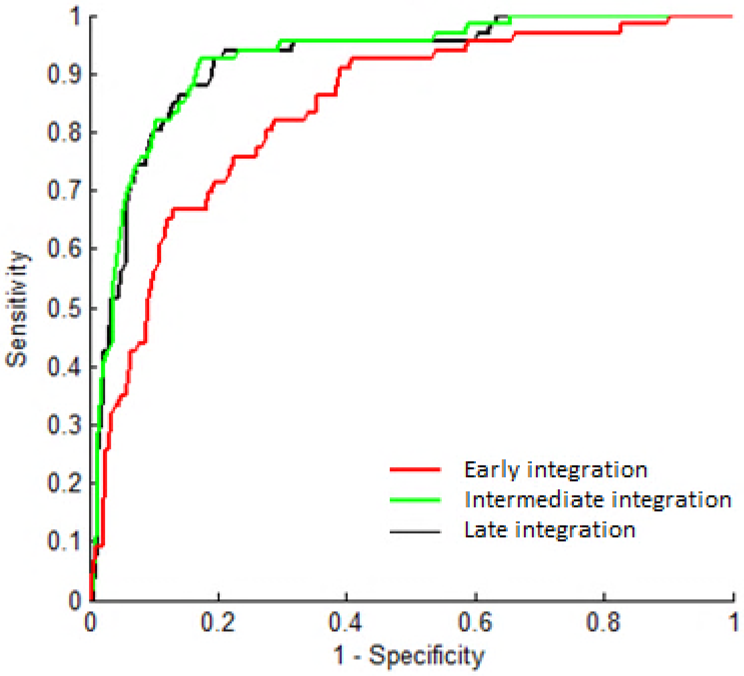

Figure 23: ROC graphs for combining the first and third feature sets used for significant genes in datasets NS and TT. By applying the early, intermediate and late integration, AUC of 0.83, 0.92 and 0.92 were achieved respectively.

To find which set of combined datasets, JCC, BJ, NS and TT, used for the first feature set resulted in better performance for OCSVM, one of the four combined datasets was removed each time, and the performance of the OCSVM was estimated based on the combination of three remaining datasets. Following results were achieved:

- Removing JCC and considering combination of three datasets of BJ, TT and NS for the first feature set resulted in AUC of 0.88.
- Removing BJ and considering combination of three datasets of JCC, TT and NS for the first feature set resulted in AUC of 0.84.
- Removing NS and considering combination of three datasets of BJ, JCC and TT for the first feature set resulted in AUC of 0.89.
- Removing TT and considering combination of three datasets of BJ, JCC and NS for the first feature set resulted in AUC of 0.89.

Based on the results, it was found that by combining 3 datasets instead of 4 datasets, OCSVM demonstrates a high performance for the first feature set. Then, we combined this feature set with the third feature set (AUC=0.83) through intermediate combination. The Roc graphs of the above results are shown in Figure 24.

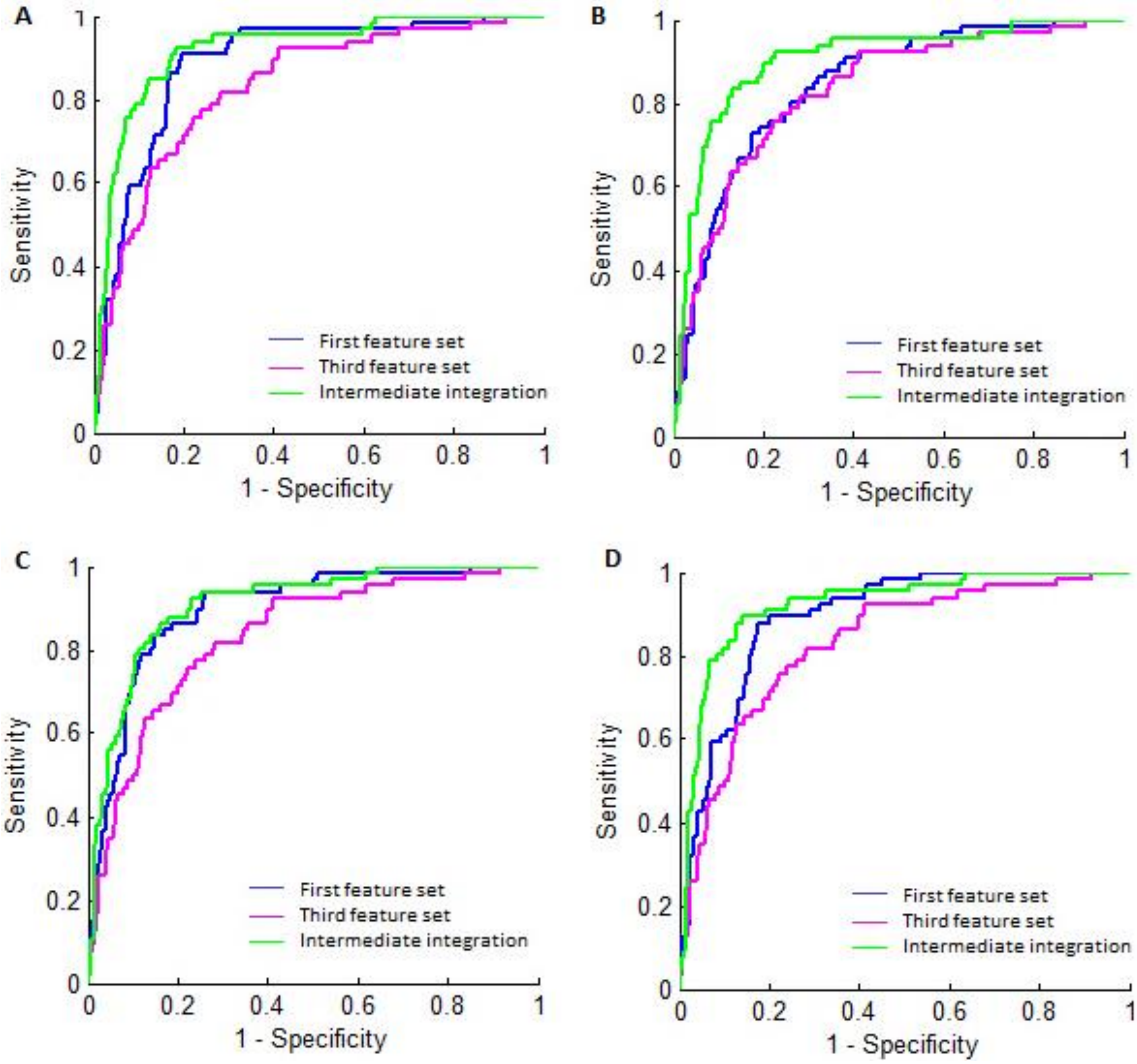

Figure 24: ROC graphs for two feature sets used for significant genes in datasets NS and TT by considering 3 datasets for the first feature set. **A)** Considering 3 datasets BJ, NS and TT for the first feature set results in AUC of 0.88. Combining first and third feature sets (AUC=0.83) through intermediate integration results in AUC of 0.92. **B)** Considering 3 datasets JCC, NS and TT for the first feature set results in AUC of 0.84, combining first and third feature sets through intermediate integration results in AUC of 0.9. **C)** Considering 3 datasets JCC, BJ and TT for the first feature set results in AUC of 0.89, combining first and third feature sets through intermediate integration results in AUC of 0.9. **D)** Considering 3 datasets JCC, BJ and NS for the first feature set results in AUC of 0.89, combining first and third feature sets through intermediate integration results in AUC of 0.92.

In the next attempt, we focused on significant genes that are common in datasets JCC and BJ. The performance of OCSVM was tested on the dataset consisting 258 genes, out of which 66 genes made the positive training set and 191genes (candidate set II) made the unlabeled set. This time, we tested the first and third feature sets leading to a high performance in the last attempt. We obtained a global AUC of 0.88 for the first set of attributes, percentage of up regulated expression profiles, and AUC of 0.82 for the third set of attributes, gene expression in different tissue types (Figure 25). According to AUC values, OCSVM demonstrates significant performance by applying these feature sets. Then, the two feature sets were combined through three different integration methods. By applying early, intermediate and late integration, AUCs of 0.82, 0.91, and 0.90 were achieved respectively. Comparing the results indicates that both intermediate and late integration outperform early integration (Figure 26).

The provided ranks for genes in candidate set II is shown in Table C.2 in Appedix C.

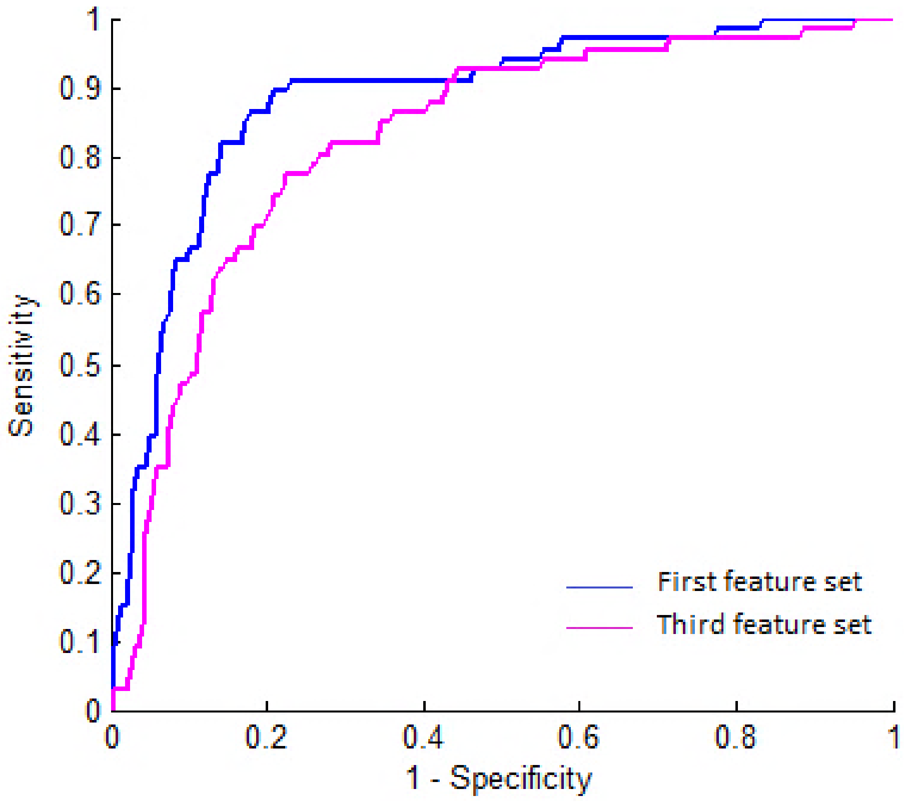

Figure 25: ROC graphs for two feature sets used for significant genes in datasets JCC and BJ. ROC curve for the first feature set, percentage of up regulated expression profiles, in blue, shows an AUC of 0.88, and ROC curve for the third feature set, in pink , results in an AUC of 0.82.

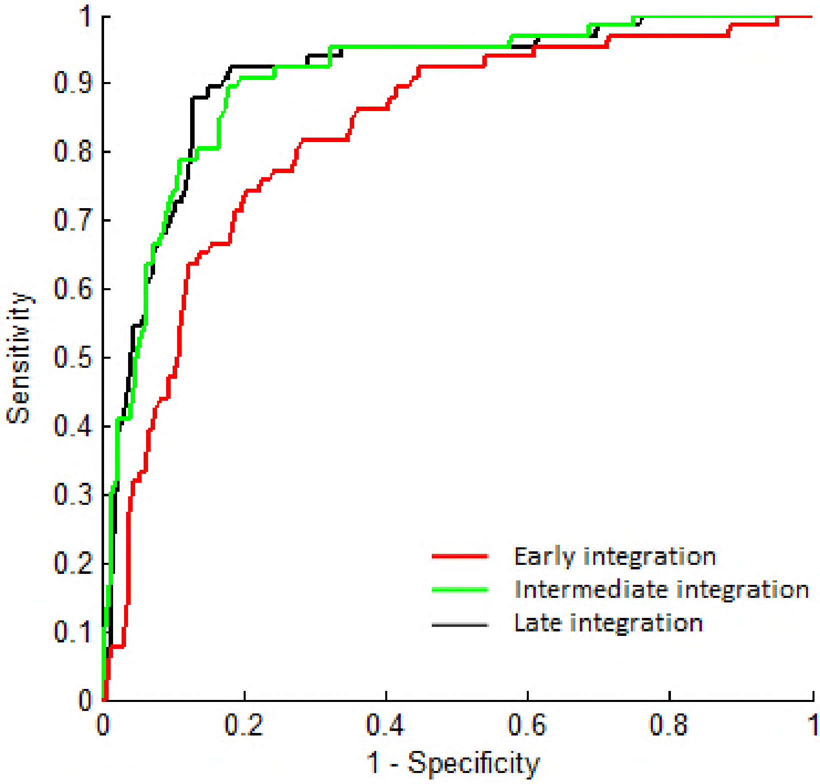

Figure 26: ROC graphs for combining first and third feature set used for significant genes in datasets JCC and BJ. By applying early, intermediate and late integration, AUC of 0.82, 0.91 and 0.90 were achieved respectively**.**

By removing the attributes of one of the four combined datasets each time, and estimating the performance of the OCSVM based on the combination of the remaining three ones, the following results were obtained for the performance of OCSVM considering first feature set:

- AUC of 0.89 on combination of three datasets JCC, TT and NS.
- AUC of 0.87 on combination of three datasets BJ, TT and NS.
- AUC of 0.87 on combination of three datasets JCC, BJ and NS.
- AUC of 0.88 on combination of three datasets JCC, BJ and TT.

Therefore, we found that the combination of three datasets also leads to a high performance for OCSVM. Then, we combined this feature set with the third feature set (AUC=0.82) through intermediate combination. The Roc graphs of the above results are shown in Figure 27.

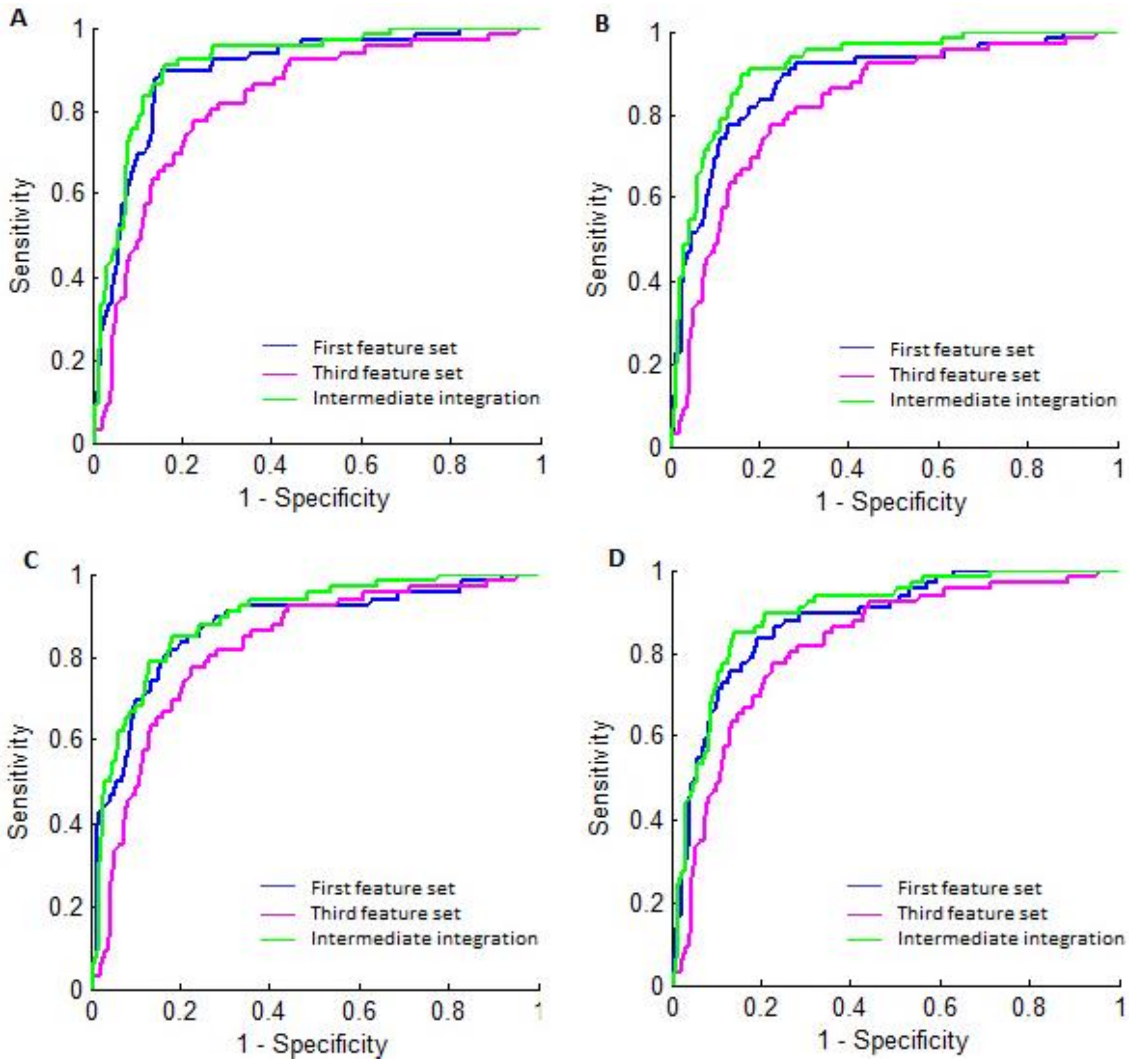

Figure 27: ROC graphs for two feature sets used for significant genes in datasets JCC and BJ by considering 3 datasets for the first feature set. **A)** Considering 3 datasets BJ, NS and TT for the first feature set results in AUC of 0.89. Combining the first and third feature sets (AUC=0.82) through intermediate integration results in AUC of 0.91. **B)** Considering 3 datasets JCC, NS and TT for the first feature set results in AUC of 0.87, combining the first and third feature sets through intermediate integration results in AUC of 0.91. **C)** Considering 3 datasets JCC, BJ and TT for the first feature set results in AUC of 0.87, combining the first and third feature sets through intermediate integration results in AUC of 0.89. D) Considering 3 datasets JCC, BJ and NS for the first feature set results in AUC of 0.88, combining the first and third feature sets through intermediate integration results in AUC of 0.9.

In the continue, we describe the results obtained from the comparison of top and bottom ranked genes in candidate set I and candidate set II.

To distinguish the difference between top and bottom ranked genes, first they were compared by considering two sets of attributes resulted to high performance for the classifier.
We plotted the expression profiles of 5 top and 5 bottom ranked genes (Appendix A.). After matching their time points to corresponding G1, S, and G2/M cell cycle phases (shown in Figure 11-14), it was found that there is a higher consistency in the expression profile of top scored genes in all series compared to bottom scored ones.

To distinguish the significance of the differential expression between top and bottom ranked genes, we compared their t-statistic value. It was found that the top ranked genes are mostly over expressed (up regulated) in cell line and under expressed (down regulated) in brain and muscle. While, bottom ranked genes follow to some extent an opposite pattern and most of them are down regulated in cell lines. Appendix B shows the t-statistic values recorded for 5 top and 5 bottom ranked genes in 2 candidate sets for 3 conditions cell line, brain and muscle.

Moreover, the function of top ranked genes was checked. Extraction of the functions of genes was done by using NCBI's database for gene-specific information (www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=gene) (Maglott et al. 2007). It was found that some of them are involved in biological processes related to cell cycle which is a good indication that our learning method prioritizes cell cycle related genes well. Then, it is predicted that top ranked candidate genes with no cell cycle function such as DONSON and C16orf59 are very likely to have cell cycle functions and it is worth studying them experimentally. The summary of biological function of 10 top ranked genes in candidate set I and candidate set II are listed in Table 4 and Table 5 respectively.

Table 4: The biological function of top ranked genes in candidate set I**.**

Gene Name	Biological function
KIF14	plays important roles in intracellular transport and cell division.
HMMR	is potentially associated with higher risk of breast cancer.
TRIM45	the encoded protein by this gene may function as a transcriptional repressor of the mitogen-activated protein kinase pathway.
RSBN1	its localization to the nucleus suggest a role of the RSBN1 protein in gene expression regulation.
BARD1	plays a central role in the control of the cell cycle in response to DNA damage.
DONSON	unknown function
ARHGAP11A	is a regulatory protein.
ASF1A	the protein encoded by this gene functions together with a chromatin assembly factor during DNA replication and repair.
C11orf17	may be involved in maintaining chromosome integrity during mitosis.
PTTG3P	The protein encoded by this gene is related to Rho-specific exchange factors and yeast cell cycle regulators.
KIF4A	may be involved in maintaining chromosome integrity during mitosis.

Table 5: The biological function of top ranked genes in candidate set II.

Gene Name	Biological function
HMMR	is potentially associated with higher risk of breast cancer.
KIF14	plays important roles in intracellular transport and cell division.
TROAP	involves with bystin and trophinin in a cell adhesion molecule complex.
ARHGAP11A	is a regulatory protein.
ASF1B	The encoded protein is the substrate of the tousled-like kinase family of cell cycle-regulated kinases.
DONSON	unknown function
BARD1	plays a central role in the control of the cell cycle in response to DNA damage.
KIF4A	may be involved in maintaining chromosome integrity during mitosis.
ECT2	The protein encoded by this gene is related to Rho-specific exchange factors and yeast cell cycle regulators.
EXO1	functions in DNA mismatch repair to excise mismatch.

We also checked the function annotations that are in common among the top and bottom ranked genes. We selected the first 30 top and the last 30 bottom scored genes of both gene candidate sets, then we used the online function annotation tool provided by database for annotation, visualization and integrated discovery (DAVID) (Huang et al. 2009) to investigate the enrichment of GO terms. GO terms analysis revealed that the significant GO biological terms among 30 top ranked genes, directly or potentially, are related to cell cycle, while no enriched GO annotation was identified for the last 30 bottom genes of the ranking list. Table 6 and Table 7 summarize the highest enriched GO terms for 30 top ranked genes of candidate set I and candidate set II respectively.

Table 6: Enriched GO terms analysis for 30 top ranked genes among candidate set I. The listed GO terms are the highest enriched GO terms (Benjamin value < 0.05) selected from the highest scored GO cluster (Enrichment score: 2.82).

GO term	Count	P_value	Benjamini
DNA metabolic process	7	2.3E-5	5.5E-3
DNA repair	5	3.6E-4	4.3E-2
cellular response to stress	6	5.1E-4	4.0E-2

Table 7: Enriched GO terms analysis for 30 top ranked genes among candidate set II. The listed GO terms are the highest enriched GO terms (Benjamin value < 0.05) selected from the highest scored GO cluster (Enrichment score: 2.05).

GO term	Count	P_value	Benjamini
DNA repair	6	2.7E-5	9.2E-3
DNA metabolic process	7	3.4E-5	5.8E-3
somatic diversification of immune receptors via somatic mutation	3	6.0E-5	6.8E-3
somatic hypermutation of immunoglobulin genes	3	6.0E-5	6.8E-3
response to DNA damage stimulus	6	9.9E-5	8.5E-3
dna repair	5	1.6E-4	5.7E-3
DNA damage	5	2.1E-4	5.0E-3
somatic recombination of immunoglobulin gene segments	3	2.5E-4	1.7E-2
DNA recombination	4	3.4E-4	1.9E-2
somatic diversification of immunoglobulins	3	3.8E-4	1.9E-2
somatic diversification of immune receptors via germline recombination within a single locus	3	4.5E-4	1.9E-2
somatic cell DNA recombination	3	4.5E-4	1.9E-2
somatic diversification of immune receptors	3	5.8E-4	2.2E-2
cellular response to stress	6	6.9E-4	2.3E-2
immunoglobulin production	3	7.1E-4	2.2E-2
production of molecular mediator of immune response	3	7.6E-4	2.2E-2

5. Conclusion and future work

The research problem we tackled in this thesis was prioritization of a list of genes with respect to being involved in the cell cycle related process. To do this, we used gene expression profiles produced by several gene expression experiments. It is already known that the gene expression profiles have been synchronized at different phases of the cell cycle. We were primarily interested in finding genes with consistent expression profiles within the starting point of the cell cycle in all series.

To solve the prioritization problem, a novelty detection approach was used by employing an OCSVM algorithm. All data were transformed into kernels applying the RBF kernel function. The candidate sets were scored according to their similarity to the genes with known cell cycle function and the rank of the all candidate genes was recorded. To estimate the performance of the used kernel functions, we employed a LOOCV strategy. Three sets of features were derived based on information regarding how experiments have been done as well as characteristics of genes. It was found that the performance of the proposed learning approach is significant by applying two sets of features. Then, the feature sets resulting high performance were combined through data integration methods and higher performance was achieved by applying the intermediate and late integration. Comparing the expression profiles of top ranked genes with bottom ranked ones, it was found that there is more consistency in expression profile of top scored genes. Finally, we checked the function of the top ranked genes and it was found that, some of them with known biological function are involved in biological processes related to cell cycle, which is a good indication that our learning method prioritizes cell cycle related genes well. Then, we can conclude that top ranked genes with unknown biological functions are very likely to have some functions related to cell cycle and it is worth studying them experimentally.

The future work for this study would be to test the performance of our learning method by using additional attributes which could be derived from existing information about expression profiles and considering other biological characteristic of genes. In addition, other methods can be used to compute the attribute sets suggested and applied in this project. For example, one can use more accurate approach to assign the cell cycle phases to genes. We also applied combination of three datasets rather than four, to test the first suggested feature set, percenge of upregulated expression profiles. OCSVM demonstrates a high performance for 3 combined datasets. However, analysis such as comparing the expresion profiles and biological functions of the top and bottom ranked genes to verify the results, when 3 datasets are applied, is still open for further research. In this work, we considered two candidate sets to be prioritized (two subgroups of the Venn diagram in Figure 21), the candidate genes in other subgroups are suggested to be ranked in the future research.

Reference

Adie, E.A., Adams, R.R., Evans, K.L., Porteous, D.J., Pickard, B.S. (2005). Speeding disease gene discovery by sequence based candidate prioritization. *BMC Bioinformatics*. 6:55.

Aerts, S., Lambrechts, D., Maity, S., Van Loo, P., Coessens, B., De Smet, F., Tranchevent, L.C., De Moor, B., Marynen, P., Hassan, B., Carmeliet, P., Moreau, Y. (2006). Gene prioritization through genomic data fusion. *NatureBiotechnology*. **24**(5): 537-544.

Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K.,Walter, P. (2008). Molecular Biology of the Cell (5th edition). New York: Garland Press.

Alpaydin, E. (2004). Introduction to Machine Learning. Cambridge, MA: The MIT Press.

Alter, O., Brown, P.O, Botstein, D. (2000). Singular value decomposition for genome-wide expression data processing and modeling. *Proc. Natl Acad. Sci. USA*. **97**: 10101-10106.

Ashburner, M. Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock,G. (2000). Gene ontology: tool for the unification of biology. The gene ontology consortium. *Nature Genetics*. **25**(1): 25-29.

Bach, F. R. Lanckriet, G. R. G, Jordan, M. I. (2004). Multiple kernel learning, conic duality, and the SMO algorithm. *In Proceedings of the 21st international conference on Machine learning. ACM.*

Bar-Joseph, Z., Siegfried, Z., Brandeis, M., Brors, B., Lu, Y., Eils, R., Dynlacht, B.D., Simon, I. (2008). Genome-wide transcriptional analysis of the human cell cycle identifies genes differentially regulated in normal and cancer cells. *Proc Natl Acad Sci U S A*. **105**(3): 955-960.

Barnes, M., Freudenberg, J., Thompson, S., Aronow, B., Pavlidis, P. (2005). Experimental comparison and cross-validation of the Affymetrix and Illumina gene expression analysis platforms. *Nucleic acids research*. **33**: 5914-5923.

Bennett, K. P., Momma, M., Embrechts, M.J. (2002). MARK: A boosting algorithm for heterogeneous kernel models. *In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 24-31.

Bi, J., Zhang, T., Bennett, K. P. (2004). Column-generation boosting methods for mixture of kernels. *In proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 521-526.

Boser, B.E., Guyon, I., Vapnik, V.N. (1992). A Training Algorithm for Optimal Margin Classifiers, in Fifth Annual Workshop on Computational Learning Theory. ACM press. 144-152.

Boulesteix, A.L., Strimmer, K. (2006). Partial least squares: a versatile tool for the analysis of high-dimensional genomic data, *Briefings in Bioinformatics*.**8**(1): 32-44.

Brown, M. P.S., Groundy, W.N., Lin, D., Cristianini, N., Sugnet, C.W., Furey, T.S., Ares, M. Jr., M., Haussler, D. (2000). Knowledge-based Analysis of Microarray Gene Expression Data using Support Vector Machines. *Proc. Natl Acad. Sci.USA*. **97**(1): 262-267.

Browne, S. M., Al-Rubeai, M. (2009). Selection Methods for High-Producing Mammalian Cell Lines' In: *Cell Engineering: Cell Line development*. Berlin. Springer.

Burges, C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*. **2**(2): 121-167.

Burnham, A.J., Mac Gregor, J.F., Viveros, R. (1999). Latent variable multivariate regression modeling. *Chemometrics and Intelligent Laboratory Systems*. **48**: 167-180.

Campbell, N.A., Reece, J.B., Urry, L.A., Cain, M.L., Wasserman, S.A., Minorsky, P.V., Jackson, R.B. (2008). Biology (8th edition). San Francisco: Benjamin Cummings.

Calvo, B., López-Bigas, N., Furney, S.J., Larrañaga, P., Lozano, J.A. (2007). A partially supervised classification approach to dominant and recessive human disease gene prediction. *Computer Methods and Programs in Biomedicine*. **85**(3): 229-237.

Chang, C.-C., Lin, C.-J. (2001). LIBSVM: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, detailed documentation (algorithms, formulae, etc) can be found in http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.ps.gz

Chou, R. C., Langan, T.J. (2003). In vitro synchronization of mammalian astrocytic cultures by serum deprivation. *Brain Res. Brain Res. Protoc*. **11**: 162- 167.

Cristianini, N., Shawe-Taylor, J. (2000). An introduction to support vector machines: and other kernel-based learning. Cambridge University Press.

Daemen, A., Gevaert, O., Ojeda, F., Debucquoy, A., Suykens, J. A., Sempoux, C., Machiels, J.-P., Haustermans, K., De Moor, B. (2009). A kernel-based integration of genome-wide data for clinical decision support. *Genome Medicine*. 1-39.

De Bie, T., Tranchevent, L. C., van Oeffelen, L. M., & Moreau, Y. (2007). Kernel-based data fusion for gene prioritization. *Bioinformatics*. **23**(13): 125-132.

Dufva, M. (2009). Introduction to Microarray Technology. *Methods Mol Biol*. **529**: 1- 22.

Elledge, S.J. (1996). Cell cycle checkpoints: preventing an identity crisis. *Science*. **274**(5293): 1664-1672.

Frank, I.E., Friedman, J.H. (1993). A statistical view of some chemometrics regression tools. *Technometrics*. **35**: 109-135.

Gardner, A.B., Krieger, A.M.,Vachtsevanos, G., Litt, B. (2006). One-Class Novelty Detection for Seizure Analysis from Intracranial EEG. *Journal of Machine Learning Research.***7**: 1025 - 1044.

Garthwaite, P.H. (1994). An interpretation of partial least squares. *J Am Stat Assoc*. **89**: 122-127.

Gaulton, K.J., Mohlke, K.L., Vision, T.J. (2007). A computational system to select candidate genes for complex human traits. *Bioinformatics*. **23**(9): 1132-1140.

Gönen, M., Alpaydin, E. (2011). Multiple Kernel Learning Algorithms. *Journal of Machine Learning Research*. **12**: 2211-2268.

Guttormsson, S.E., Marks, R.J., Sharkawi, M. A. E. (1999). Elliptical novelty grouping for on-line short-turn detection of excited running rotors", IEEE Trans. Energy Conversion. **14**: 16-22.

Herrup, K., Yang, Y. (2007). Cell cycle regulation in the postmitotic neuron: oxymoron or new biology? *Nature Reviews Neuroscience*. **8**(5): 368-78.

Hoskuldsson, A. (1996). Prediction Methods in Science and Technology, Thor Publishing, Copenhagen, Denmark.

Huang, D.W., Sherman, B.T., Lempicki, R.A. (2009). Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature protocols*. **4**(1)**:** 44-57.

Johansson, D., Lindgren, P., Berglund, A. (2003). A multivariate approach applied to microarray data for identification of genes with cell cycle-coupled transcription. *Bioinformatics*. **19**(4): 467-473.

Kapushesky, M., Emam, I., Holloway, E., Kurnosov, P., Zorin, A., Malone, J., Rustici, G., Williams, E., Parkinson, H., Brazma, A. (2010). Gene expression atlas at the European bioinformatics institute. *Nucleic Acids Res*. **38** (Database issue): D690-698.

Kecman, V. (2001). Learning and Soft Computing. London, UK: The MIT Press.

Khammanit, R., Chantakru, S., Kitiyanant, Y., Saikhun, J. (2008). Effect of serum deprivation and chemical inhibitors on cell cycle synchronization of canine dermal fibroblasts. *Theriogenology*. **70**(1): 27-34.

Lanckriet, G.R.G., De Bie, T., Cristianini, N., Jordan, M.I., Noble, W.S. (2004). A statistical framework for genomic data fusion. *Bioinformatics*. **20**: 2626-2635.

López-Bigas, N., Blencowe, B.B., Ouzounis, C.A. (2006). Highly consistent patterns for inherited human diseases at the molecular level. *Bioinformatics*. **22** (3): 269–277

López-Bigas, N., Ouzounis, C.A. (2004). Genome-wide identification of genes likely to be involved in human genetic disease. *Nucleic Acids Research*. **32**(10): 3108-3114.

Ma, X., Lee, H., Li, W., Sun, F. (2007). A new approach for prioritizing Genes by Combining Gene Expression and Protein- Protein Interactions. *Bioinformatic*. **23** (2): 215-221.

Madan Babu, M. (2004). Introduction to Microarray Data Analysis, chapter in Computational Genomics, Horizon press (Grant, R. Editor).

Maglott, D., Ostell, J., Pruitt, K.D., Tatusova, T. (2005). Entrez Gene: gene-centered information at NCBI. *Nucleic acids research*. **33**(suppl 1): D54-D58.

Martens, H., Næs, T. (1989). Multivariate Calibration. John Wiley and Sons, Chichester, UK.

Merrill, G. F. (1998). Cell synchronization. *Methods Cell Biol*. **57:** 229-249.

Morgan, D. O. (2007). The cell cycle: principles of control. London: New Science Press.

Müller, K. R.,Mika, S., Rätsch, G., Tsuda, K., Schölkopf, B. (2001). An introduction to kernel-based learning algorithms. K. R. *IEEE Transactions on Neural Networks*, **12**(2): 181-202.

Partridge, T. A. (2002). Cells that participate in regeneration of skeletal muscle. *Gene Therapy*. **9**: 752-753.

Pavlidis, P., Weston, J., Cai, J., Grundy, W.N. (2002). Learning gene functional classifications from multiple data types. *Journal* of Computational *Biology*. 9(2): 401-411.

Pedrali-Noy, G., Spadari, S., Miller-Faurès, A., Miller, A.O., Kruppa, J., Koch, G. (1980). Synchronization of HeLa cell cultures by inhibition of DNA polymerase alpha with aphidicolin. *Nucleic Acids Res*. **8**(2): 377-387.

Peña-Diaz, J., Hegre, S.A., Anderssen, E., Aas, P.A., Mjelle, R., Gilfillan, G.D., Lyle, R., Drabløs, F., Krokan, H.E., Sætrom, P. (2013). Transcription profiling during the cell cycle shows that a subset of Polycomb-targeted genes is upregulated during DNA replication. *Nucleic Acids Res*. **41**(5): 2846-2856.

Perez-Iratxeta, C., Bork, P., Andrade-Navarro, M.A. (2007). Update of the G2D tool for prioritization of gene candidates to inherited diseases. *Nucleic Acids Research*, **35** (Web Server issue): 212-216.

Rossi, S., Masotti, D., Nardini, C., Bonora, E., Romeo, G., Macii, E., Benini, L., Volinia, S. (2006). a web-based integrated approach for identification of candidate disease genes. *Nucleic Acids Research*, **34** (Web Server issue): 285-292.

Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J. and Williamson, R.C. (1999). Estimating the support of a high-dimensional distribution. Microsoft Research Corporation Technical report. MSR-TR-99-87.

Schölkopf, B., Platt, J.C, Taylor, J. S., Smola., A.J, Williams, R.C. (1999). Support Vector Method for Novelty Detection. *Advances in Neural Information Processing Systems* 12. Solla, S.A., Leen, T.K., Muller, K.R., Solla, S.A., Leen, T.K.,. Muller, K. R, editors. The MIT Press. 582-588.

Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J. and Williamson, R.C. (2001). Estimating the support of a high-demensional distribution. NeuralComputation, 13:1443-1472.

Schölkopf, B, Tsuda, K., Vert, J.P. (2004). Kernel methods in computaiaonal biology. Cambridge (Massachusetts): MIT Press.

Schölkopf, B., Smola, A.J. (2002). Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, chapter 7, pp.208–209, MIT Press, Cambridge, MA.

Selvaraj, S., Natarajan, J. (2011). Microarray Data Analysis and Mining Tools, *Bioinformation*. **6**(3): 95-99.

Spiegelhalter, D. J., Taylor, C. C., Michie, D. (1994). Machine learning, neural and statistical classification. New York ; London: Ellis Horwood.

Stears, R.L., Martinsky, T., Schena M. (2003). Trends in microarray analysis. *Nature Medicine* **9**(1): 140-145.

Stone M., Brook, R.J. (1990). Continuum regression: cross-validated sequentially constructed prediction embracing ordinary least squares, partial least squares and principal component regression. *J Roy Stat Soc B*. **52**: 237-269.

Trygg, J., Wold, S. (2001). Orthogonal projections to latent structures (O-PLS). *J. Chemometrics*. **58**: 131-150.

Vapnik, V. N. (1982). Estimation Dependences Based on Empirical Data. New York. Springer-Verlag.

Vapnik, V. N. (1998). Statistical Learning Theory: John Wiley & Sons. New York.

Whitfield, M.L., George, L.K., Grant, G.D., Perou, C.M. (2006). Common markers of proliferation. *Nat Rev Cancer*. **6**: 99-106.

Wu, R.-S., W.-H. Chung. (2009). Ensemble one-class support vector machines for content-based image retrieval. *Expert Systems with Applications*. **36**(3): 4451-4459.

Yeung, D.Y, Chow, C. (2002). Parzen-window network intrusion detectors. *Proceedings of the 6th International Conference on Pattern Recognition.***4**: 385-388.

Yeung, D.Y, Ding, Y. (2002). Host-based intrusion detection using dynamic and static behavioral models. *Pattern Recognition*. **36**: 229-243.

Zhang, P., Zhang, J., Sheng, H., Russo, J.J., Osborne, B., Buetow, K. (2006). Gene functional similarity search tool (GFSST). *BMC Bioinformatics*. 7-135.

Zieve, G.W., Turnbull, D., Mullins, J.M., McIntosh, J.R. (1980). Production of large numbers of mitotic mammalian cells by use of the reversible microtubule inhibitor nocodazole accumulated mitotic cells. *Exp Cell Res1.***26**: 397-405.

Zvelebil, M., Baum, J.O. (2008). Understanding bioinformatics. New York, USA: Garland Science.

Appendix A. Expression profiles for selected genes

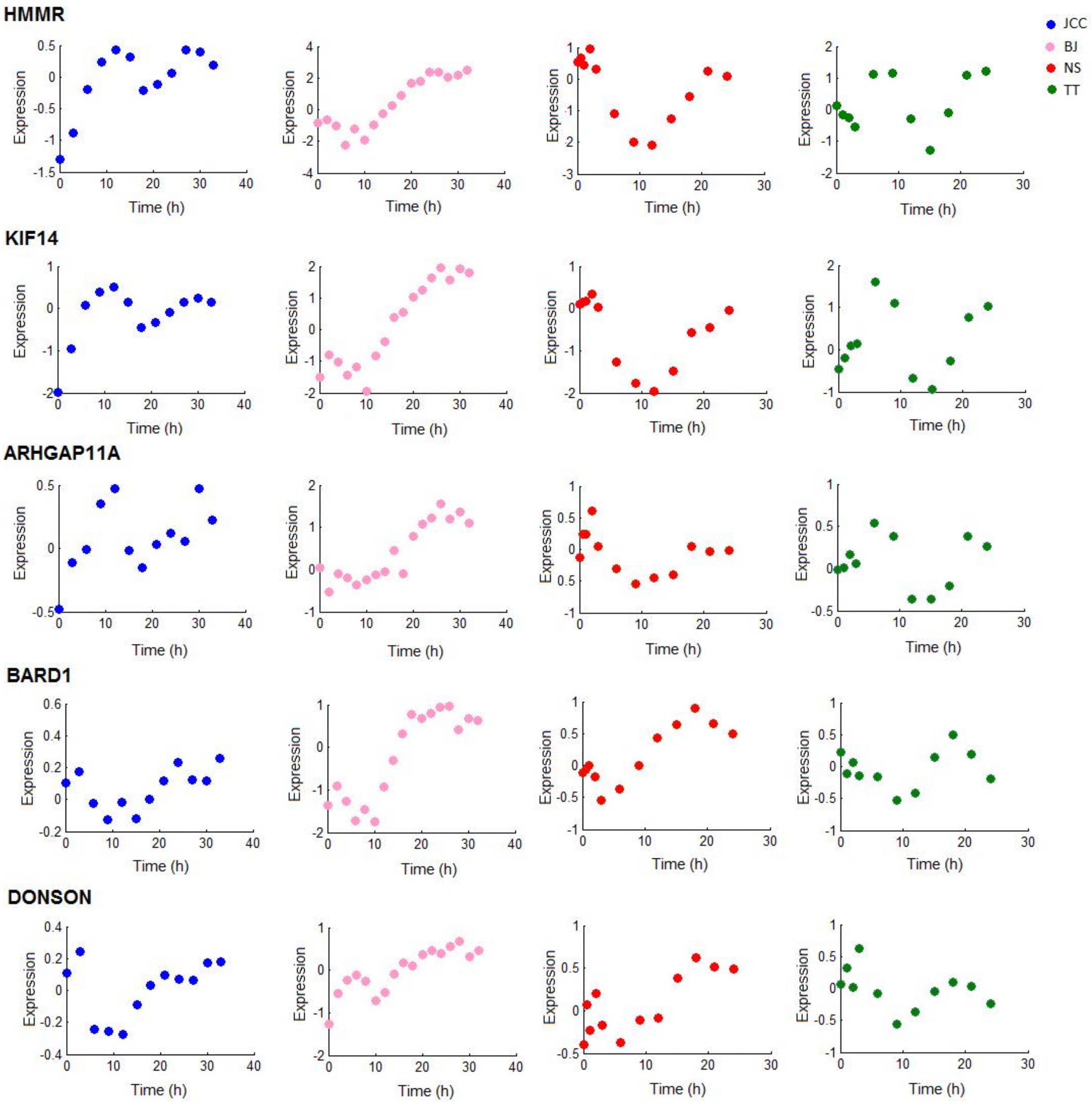

Figure A.1: Expression profile of selected top ranked genes in candidate set I and candidate set II. The selected genes are common in both candidate sets and are among the first 10 top ranked genes.

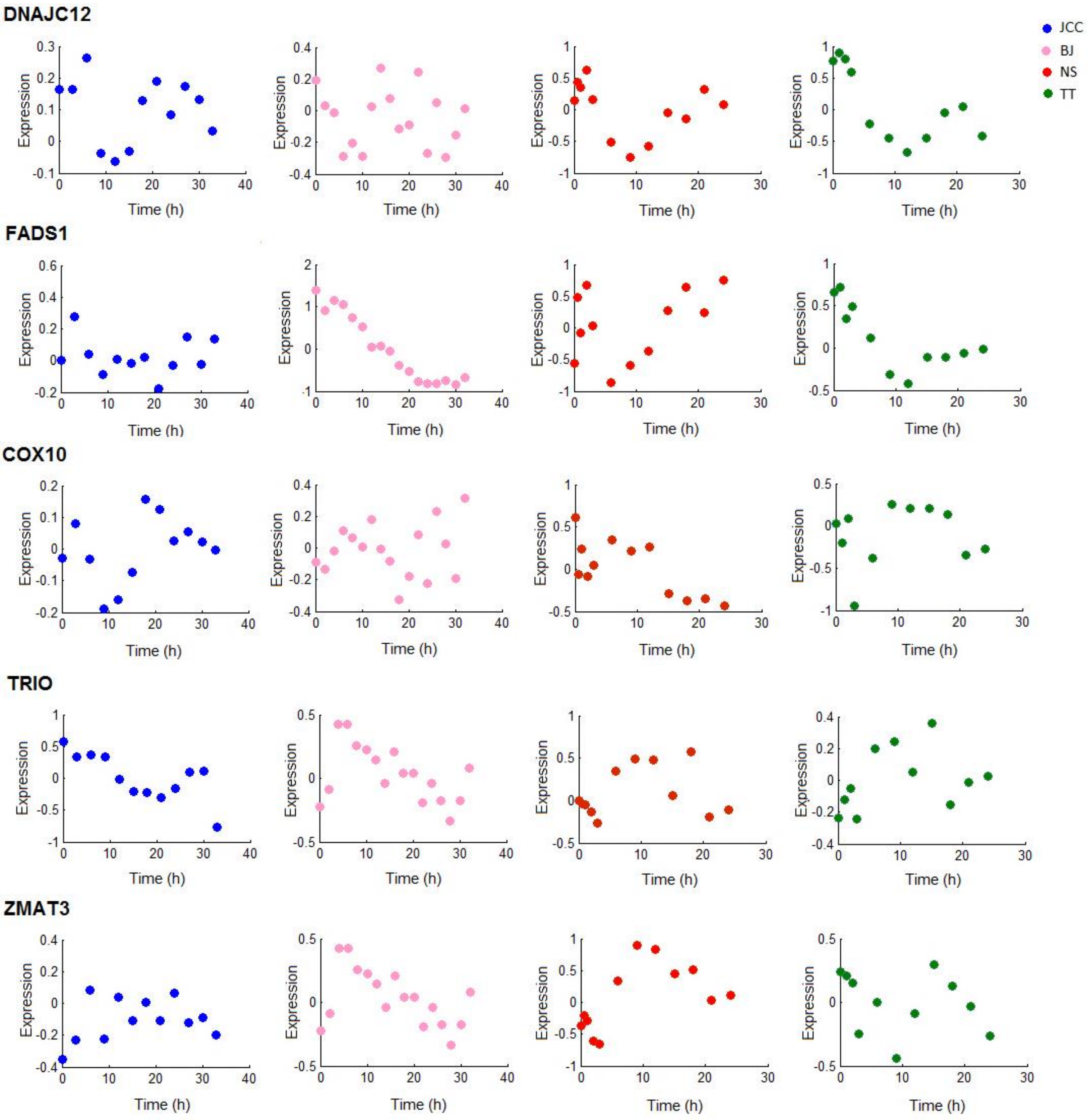

Figure A.2: Expression profiles of 5 bottom ranked genes in candidate set I.

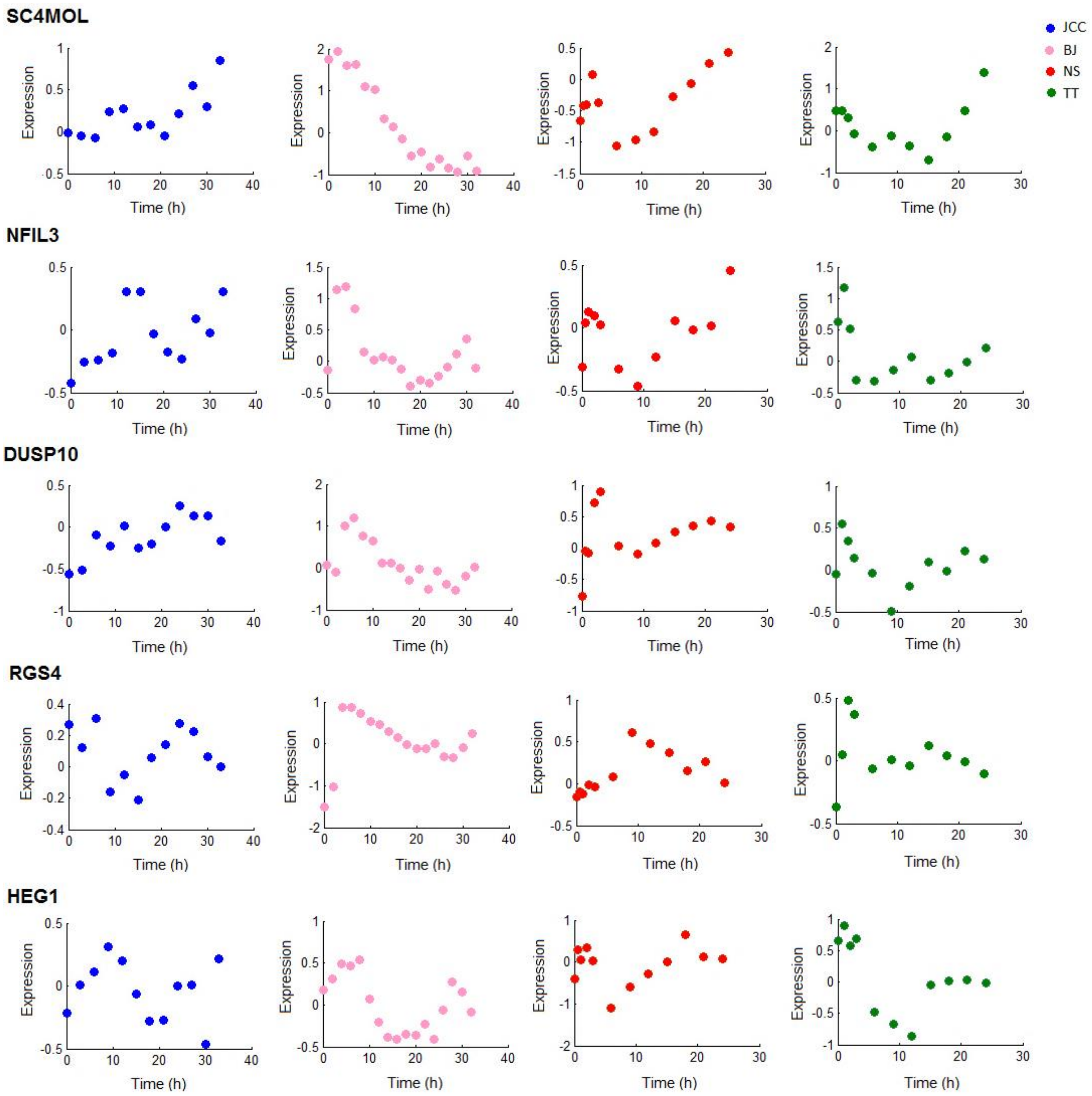

Figure A.3: Expression profiles of 5 bottom ranked genes in candidate set II.

Appendix B. T-statistic values recorded for selected genes in three conditions, cell line, brain, and muscle.

Table B.1: T-statistic values of selected top ranked genes in candidate set I and candidate set II. The selected genes are common in both candidate sets and are among the first 10 top ranked genes.

Gene names	T-statistic value cell line	T-statistic value brain	T-statistic value muscle
HMMR	40	-21	-14
KIF14	49	-22	-17
ARHGAP11A	46	-18	-15
BARD1	24	-19	-11
DONSON	30	-15	-13

Table B.2: T-statistic values of 5 bottom ranked genes in candidate set I.

Gene names	T-statistic value cell line	T-statistic value brain	T-statistic value muscle
DNAJC12	-3.1	28	-4.2
FADS1	12	8.7	-13
COX10	9.2	-20	49
TRIO	4.7	-5.5	3.1
ZMAT3	-37	46	-15

Table B.3: T-statistic values of 5 bottom ranked genes in candidate set II.

Gene names	T-statistic value cell line	T-statistic value brain	T-statistic value muscle
SC4MOL	9.8	9.8	-24
NFIL3	-14	-14	3.8
DUSP10	-18	-5.5	3.1
RGS4	-37	46	-15
HEG1	-11	-14	18

Appendix C. List of candidate genes and their predicted ranks.

Table C.1: The 230 candidate genes (candidate set I) and obtained ranks through gene prioritization.

Ranks	Genes Name	Ranks	Genes Name	Ranks	Genes Name
1	KIF14	46	BRD8	91	RAB23
2	HMMR	47	CDCA4	92	POLR1E
3	TRIM45	48	S100PBP	93	RAB30
4	RSBN1	49	RPP40	94	TIMM10
5	BARD1	50	PRR11	95	HARS2
6	DONSON	51	ZNF165	96	C2orf49
7	ARHGAP11A	52	DHX40	97	HOOK1
8	ASF1A	53	H1FX	98	SERPINH1
9	C11orf17	54	HMGB3	99	ACYP1
10	PTTG3P	55	PASK	100	NGFRAP1
11	WDR76	56	CASZ1	101	PTEN
12	KIF4A	57	MRPS12	102	MAPKAP1
13	EXO1	58	IMPDH2	103	UPF3B
14	MELK	59	IFIT1	104	TRIM45
15	FAM158A	60	ABCC2	105	C10orf2
16	C16orf59	61	FAM111A	106	AP1S2
17	TOPBP1	62	ICMT	107	HKDC1
18	PUS1	63	FERMT1	108	ZNF434
19	ARHGAP19	64	SEPX1	109	DUSP11
20	RNASEH1	65	SRSF6	110	PAQR4
21	CTDSPL	66	EBAG9	111	TMEM5
22	FAM64A	67	H1F0	112	AK4
23	METTL1	68	PDSS1	113	C14orf147
24	UNG	69	LBR	114	ASB9
25	TTF2	70	PHACTR4	115	FOXD2
26	NDRG3	71	C5orf13	116	FAM53B
27	GRPEL1	72	RAD51AP1	117	C16orf61
28	GAR1	73	GTPBP4	118	SRSF3
29	SNAP29	74	VAPA	119	SRPRB
30	MAP3K6	75	TPRA1	120	CD97
31	NEIL3	76	NUP62CL	121	RBM14
32	GOT1	77	ABCC5	122	GATA2
33	MSH6	78	ADCK2	123	GPSM2
34	NIP7	79	DDX18	124	AFAP1
35	ATAD2	80	STRA13	125	ANKS1A
36	DCAF7	81	IL17RB	126	GMCL1
37	UBR7	82	HIST1H4C	127	IMPAD1
38	FAM60A	83	DCTPP1	128	C8orf51
39	DDX46	84	SLC25A38	129	UTP14A
40	ATAD5	85	DCP2	130	UPF1
41	TMEM194A	86	REEP1	131	SLC6A9
42	RCL1	87	EBNA1BP2	132	ANXA10
43	MXD3	88	UAP1L1	133	ECE2
44	MRTO4	89	SNX2	134	ITFG2
45	REEP4	90	ENO2	135	SLC5A6

Ranks	Genes Name	Ranks	Genes Name	Ranks	Genes Name
136	MTRR	181	PVT1	226	DNAJC12
137	SOCS2	182	RPL29	227	FADS1
138	MTHFD2L	183	NR4A2	228	COX10
139	ZNF35	184	SNRPB2	229	TRIO
140	SQLE	185	MDFIC	230	ZMAT3
141	HNRNPA1	186	PLCXD1		
142	ZNF593	187	ARAP3		
143	BNIP3L	188	VWA5A		
144	WDR46	189	DPYSL2		
145	NHP2L1	190	PLEKHF1		
146	FOXF2	191	RPL34		
147	RBM12	192	DNAJB6		
148	ODC1	193	STAT2		
149	EEF1B2	194	EAF2		
150	CITED2	195	PDK4		
151	TRAK2	196	PALM		
152	CDR2	197	IMP4		
153	KIAA0355	198	DTNA		
154	SH3GL2	199	EBLN2		
155	QPCT	200	WIPI1		
156	KCTD9	201	PLSCR4		
157	KBTBD2	202	PTPRE		
158	APAF1	203	RGS20		
159	SMTN	204	ATP9A		
160	HTR3A	205	RAB40B		
161	NCAM2	206	TFAP2A		
162	ETV5	207	HEG1		
163	HADH	208	SPG11		
164	SFRS18	209	ZBED1		
165	IFT27	210	SRPK2		
166	IFIT2	211	DENND5B		
167	MTIF2	212	GBP2		
168	RHBDF1	213	PEX11B		
169	KRT16	214	PCDH7		
170	OPTN	215	PGM1		
171	HSPH1	216	UBE2A		
172	KIFAP3	217	SERPINE2		
173	DEXI	218	IFRD1		
174	RSBN1	219	TCEB3		
175	HMOX1	220	KIAA0913		
176	GK	221	INSIG2		
177	ACO1	222	GOLGA8B		
178	ACAD8	223	RAB26		
179	ADARB1	224	COL1A1		
180	GABARAPL1	225	WBP5		

Table C.2: The 191 candidate genes (candidate set II) and obtained ranks through gene prioritization.

Ranks	Genes Name	Ranks	Genes Names	Ranks	Gene Names
1	HMMR	46	UBR7	91	FJX1
2	KIF14	47	DHX40	92	EFHC1
3	TROAP	48	CASP6	93	DNAJA1
4	ARHGAP11A	49	FUBP1	94	CDR2
5	C16orf59	50	AGFG1	95	TNFRSF10B
6	ASF1B	51	MTDH	96	H2AFV
7	BARD1	52	VPS37B	97	FOXF2
8	KIF4A	53	HMGB2	98	RABGGTB
9	ECT2	54	NFU1	99	UTP14A
10	EXO1	55	MAT2A	100	NUP88
11	ASF1A	56	GAPVD1	101	KANK2
12	NIF3L1	57	CBR3	102	ZNHIT6
13	RMI1	58	CLINT1	103	LARP4
14	WDR76	59	FZD2	104	ADAMTS1
15	QTRTD1	60	CCNJ	105	MREG
16	HMGB3	61	HIST1H4C	106	MSL2
17	FAM64A	62	POLR1C	107	SIX1
18	C16orf59	63	GNL2	108	SLC38A1
19	SAP30	64	MINA	109	SEC24A
20	RPL39L	65	ZCCHC10	110	GPSM2
21	UNG	66	TMEM97	111	NUFIP1
22	GFPT2	67	RFX5	112	IBTK
23	SHCBP1	68	EXOSC8	113	LDLR
24	AVL9	69	APOBEC3B	114	SFPQ
25	PVRL2	70	TMEM149	115	ANXA3
26	PASK	71	MID1	116	TGIF1
27	MSH6	72	SMYD3	117	ZBTB11
28	RAD51	73	KDM5B	118	ZNF330
29	PRR11	74	SGPL1	119	CDYL
30	FAM60A	75	AHSA1	120	VEZF1
31	CTPS	76	TFB2M	121	RAB7L1
32	DKC1	77	MXD3	122	WSB1
33	BRD8	78	MYC	123	SLC7A11
34	CEBPG	79	GPR172A	124	MYST4
35	MPHOSPH10	80	NOC3L	125	ARL6IP1
36	H1FX	81	NOC3L	126	STX6
37	WDR67	82	DDIT4	127	GRSF1
38	SPA17	83	METTL21D	128	TCF7L2
39	ARFGEF2	84	EIF3J	129	GRPEL1
40	NET1	85	CCNT2	130	EMP1
41	CLDN4	86	RPIA	131	ENTPD4
42	UCK2	87	MAP4K4	132	ZNF12
43	IFIT1	88	H1F0	133	OPCML
44	EIF3A	89	FANCE	134	MKNK2
45	EPS8	90	TRAF3	135	ACSL3

Ranks	Genes Name	Ranks	Genes Name
136	TFPI	181	PDCD4
137	PDGFA	182	BTG1
138	SPTLC2	183	HOMER1
139	FKBP14	184	NPC1
140	SLC2A6	185	OPTN
141	SLC33A1	186	PIKFYVE
142	C11orf95	187	ACSL1
143	IFIT2	188	SC4MOL
144	EAPP	189	NFIL3
145	C1orf103	190	DUSP10
146	PLXNA1	191	RGS4
147	ACSL4	192	HEG1
148	CDK19		
149	TMEM2		
150	MAP2K3		
151	GFPT2		
152	SLC7A6		
153	HNRNPA3		
154	TM4SF1		
155	EIF2AK3		
156	OSBPL2		
157	SENP6		
158	IFIT3		
159	SLC19A2		
160	SS18L1		
161	JMJD6		
162	NUAK1		
163	SACS		
164	ECM2		
165	CREB3L2		
166	PANK3		
167	BIRC3		
168	ATP2B1		
169	IRF7		
170	ANKRD27		
171	AJAP1		
172	FZD7		
173	RPL28		
174	SNX1		
175	GDPD5		
176	KDM5A		
177	PLEKHF1		
178	PLEKHF1		
179	AHI1		
180	CCL2		