LDA-BASED DIMENSIONALITY REDUCTION AND DOMAIN ADAPTATION WITH APPLICATION TO DNA SEQUENCE CLASSIFICATION

by

SURBHI MUNGRE

B.E., Rajiv Gandhi Proudyogiki Vishwavidyalaya, India, 2006

A THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2011

Approved by:

Major Professor
Dr. Doina Caragea

Abstract

Several computational biology and bioinformatics problems involve DNA sequence classification using supervised machine learning algorithms. The performance of these algorithms is largely dependent on the availability of labeled data and the approach used to represent DNA sequences as *feature vectors*. For many organisms, the labeled DNA data is scarce, while the unlabeled data is easily available. However, for a small number of well-studied model organisms, large amounts of labeled data are available. This calls for *domain adaptation* approaches, which can transfer knowledge from a *source* domain, for which labeled data is available, to a *target* domain, for which large amounts of unlabeled data are available. Intuitively, one approach to domain adaptation can be obtained by extracting and representing the features that the source domain and the target domain sequences share. *Latent Dirichlet Allocation* (LDA) is an unsupervised dimensionality reduction technique that has been successfully used to generate features for sequence data such as text. In this work, we explore the use of LDA for generating predictive DNA sequence features, that can be used in both supervised and domain adaptation frameworks. More precisely, we propose two dimensionality reduction approaches, LDA Words (LDAW) and LDA Distribution (LDAD) for DNA sequences. LDA is a probabilistic model, which is generative in nature, and is used to model collections of discrete data such as document collections. For our problem, a sequence is considered to be a “document” and k-mers obtained from a sequence are “document words”. We use LDA to model our sequence collection. Given the LDA model, each document can be represented as a distribution over topics (where a topic can be seen as a distribution over k-mers). In the LDAW method, we use the top k-mers in each topic as our features (i.e., k-mers with the highest probability); while in the LDAD method, we use the

topic distribution to represent a document as a feature vector. We study LDA-based dimensionality reduction approaches for both supervised DNA sequence classification, as well as domain adaptation approaches. We apply the proposed approaches on the splice site predication problem, which is an important DNA sequence classification problem in the context of genome annotation. In the supervised learning framework, we study the effectiveness of LDAW and LDAD methods by comparing them with a traditional dimensionality reduction technique based on the information gain criterion. In the domain adaptation framework, we study the effect of increasing the evolutionary distances between the source and target organisms, and the effect of using different weights when combining labeled data from the source domain and with labeled data from the target domain. Experimental results show that LDA-based features can be successfully used to perform dimensionality reduction and domain adaptation for DNA sequence classification problems.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I express my deepest appreciation to my advisor, Dr. Doina Caragea, for her support. Under her guidance, I have learned many things in the last two years. I specially acknowledge her patience to help me figure out the right project to work on, providing me flexibility to implement my new ideas and provide valuable inputs for this work. It has been a pleasure and an honor working with her.

I would also like to thank Dr. Gurdip Singh for his support and for graciously accepting to serve on my committee.

My special thanks to Dr. Torben Amtoft for his comments and valuable feedback that helped in refining the quality of this work.

I am also thankful to the staff in the department of Computing and Information Sciences at KSU for their good natured support and for providing me with the resources needed to materialize ideas into an accomplished thesis.

I would also like to thank all my friends for chipping in with useful suggestions and for always being there to help me during the entire course of this work and degree. Thank you for your well wishes and blessings.

The research described in this thesis was supported in part by a grant from the National Science Foundation (NSF 0711396) and Arthropod Genomics Center at Kansas State University.

Dedication

I dedicate this thesis to my father, Mr. Sandeep Mungre, and my mother, Mrs. Sampada Mungre, for their love and support at every stage of my life. It is because of their motivation and encouragement that I am able to complete my Masters.

My husband, Swarnim Kulkarni, has always had faith in me. This thesis would not be complete without dedicating it to him as an acknowledgment for words of encouragement and helping me to get through the difficult times.

Chapter 1

Introduction

In this chapter, we begin by providing the motivation for this work along with the problem statement. We then give a brief overview of the approaches proposed to solve these problems.

1.1 Motivation and Problem Statement

Today, we have a plethora of biological data available to us, due to the next generation sequencing technologies, which make it possible to sequence DNA at an ever-faster speed for lower cost. With the introduction of next generation sequencing, throughput per machine has increased 500,000-fold, while the number of reads per genome has increased by $\sim$ 100-fold [Monya, 2010]. Everything from storage to data processing to data analysis has to catch up with the speed of the new sequencing machines. Machine learning algorithms, which are extensively used to annotate DNA sequences, also have to keep the pace with the new sequencing machines. Some of the challenges with DNA sequences classification using machine learning algorithms are given by:

- *Need for compact feature representation*: DNA sequences contain certain 'features' (signals), which make it possible to apply learning algorithms on such data. Extracting and representing DNA sequences as feature vectors, which can capture useful information, is a challenging problem. If we naively use a brute force approach to obtain features from sequence data, then, we might end up a with large number of features. In such a situation, there will be many irrelevant features. This will have an adverse effect on the accuracy of the classifier learned from the data. Furthermore, due to the large number of features, the processing time and resources needed would also be large.

- *Need for labeled data*: There are some model organisms which are extensively studies and huge amounts of labeled data are available for such organisms. However, for many species there is not much labeled data available, even though huge amounts of unlabeled data is available for them. Given such scenarios, it has become essential to design techniques for learning how to transfer knowledge from one domain for which labeled data is available to a new domain for which not much labeled data is available. This problem is known as Domain Adaptation [Arnold et al., 2008], [DaumeIII, 2007], or more generally, Transfer Learning [Arnold et al., 2008]. With domain adaptation we can use labeled data from one organism to make predictions about another organism.

Latent Dirichlet Allocation (LDA) is unsupervised dimensionality reduction technique that has been successfully used to generate features for sequence data such as text, but not for DNA sequences. In this work, we want to explore if we can use LDA to identify predictive features for DNA sequence classification in a supervised learning framework. Furthermore, we explore if we can use LDA to identify features shared by two domains, a source domain, for which labeled data is available, and a target domain, for which not much labeled data exists, but large amounts of unlabeled data are available. Thus, the problems address in this work are as follows:

- Study LDA-based dimensionality reduction approaches in the context of supervised DNA sequence classification.

- Study the usefulness of LDA features in the context of DNA sequence classification using domain adaptation approaches (where the goal is to transfer knowledge from a source domain to a target domain).

- Apply the proposed approaches to the problem of predicting splice sites in DNA sequences.

1.2 Overview of the Proposed Approaches

As discussed in the above section, a major issue in building a classifier for biological sequence classification tasks is how to represent a sequence, as the accuracy of the classification largely depends on the adopted representation. A simple way to represent a biological sequence is to consider all k-mers for the sequence [Islamaj et al., 2006]. However, this technique will produce a large number of k-mers. We propose two approaches for feature extraction, (a) LDA words (LDAW) (b) LDA distribution (LDAD). Both approaches make use of Latent Dirichlet Allocation (LDA) for feature extraction. LDA, is a generative probabilistic model for collections of discrete data such as text corpora [Blei et al., 2003]. Given a set of sequences represented in form of k-mers, LDA models each sequence in this collection as a mixture of topics, where each topic is a mixture of k-mers. In LDAW we use the top k-mers in each topic (i.e., k-mers with the highest probability) as our features. On the other hand, in LDAD we use the topic distribution to represent a document as a feature vector. These approaches are explained in detail in Chapter 4.

We used supervised machine learning algorithms to study the LDAW and LDAD approaches for feature generation in the first part of this work. For supervised algorithm we need enough labeled data to learn a model during training phase. If we do not have enough data then the performance degrades. Domain adaptation can be used even when we do not have enough labeled data to learn a model. It makes use of labeled data from a related domain called *source domain* to learn a model for a domain called *target domain*, which does not have enough labeled data. There are several assumptions about the source and target datasets that we make in this study. First, the source domain has a large number of labeled samples and the target domain has a small number of labeled samples but a lot of unlabeled samples. Nevertheless, we are studying the behavior for various data set sizes.

Second, both the source and the target domains are represented by the same set of features. Third, for our study we also assumed that both the source and the target domains have the same classes, which means that we are addressing a domain adaptation problem. The basic assumption in any domain adaptation method is that, the source and the target domains have different data distribution. If this was the case then there would not be any need of domain adaptation.

The rest of the thesis is organized as follows: Chapter 2 provides biological background and gives an overview of the machine learning algorithms used. This chapter also gives an overview of Latent Dirichlet Allocation, which is the foundation of our work. A discussion of the related work can be found in Chapter 3. Chapters 4 presents the two approaches for dimensionality reduction, along with the overview of domain adaptation algorithms. Chapters 5 and 6 describe the experimental setup and the results obtained for our experiments, respectively. Chapters 7 presents several directions for future work and summarizes conclusions drawn from this work.

Chapter 2

Background

In this chapter, we provide some biological background and also explain machine learning techniques and topic modeling fundamentals, which form the basic building blocks of this work.

2.1 Biology

The mRNA splicing prediction problem, which is the main focus of this work, is a subproblem of the gene prediction problem. Before getting into the details of gene prediction, we need to understand gene structure. Thus, we begin our discussion by describing the gene structure.

2.1.1 Gene Structure

The modern working definition of a gene is “A locatable region of genomic sequence, corresponding to a unit of inheritance, which is associated with regulatory regions, transcribed regions and/or other functional sequence regions.” [Pearson, 2006]. We restrict this discussion to eukaryote protein-coding genes, which are the type of genes we will be dealing with in this work.

Transcription, splicing and translation are the three main steps involved in protein synthesis (see Figure 2.1). We will explain each of these steps in details and, in the process, we will also explain various regions in a gene.

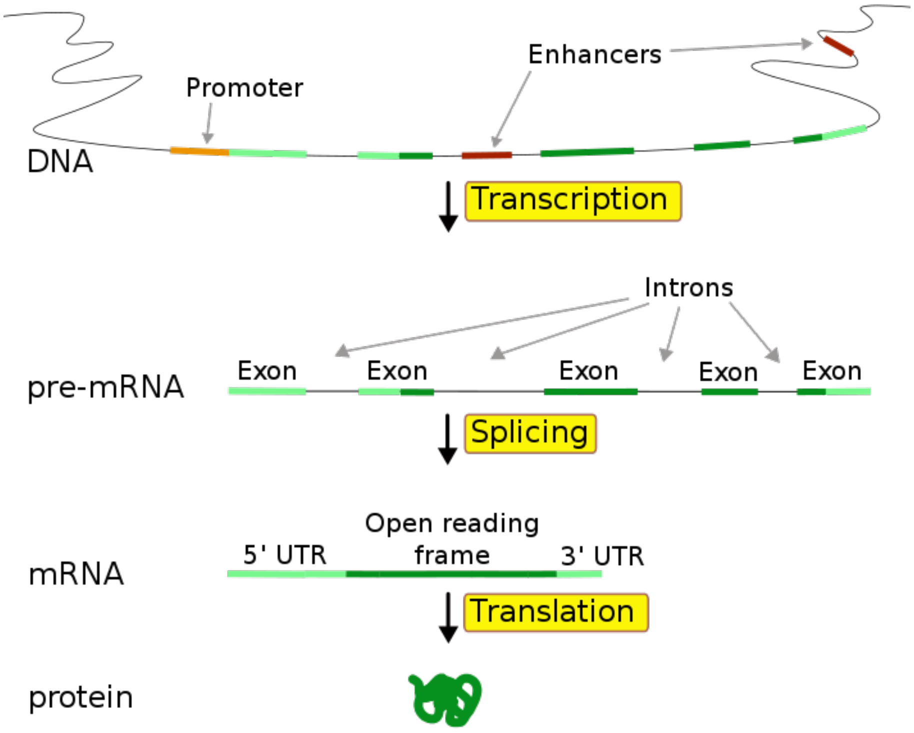


Figure 2.1: *Central dogma of molecular biology: transcription, splicing and translation.* [1]

1. Transcription: In simple terms, *transcription* is defined as the process of formation of RNA from DNA. There are three important phases of this process. Initiation is the first phase in which a mRNA polymerase binds to a sequence of DNA located immediately upstream of a gene, called promoter. Elongation is the next phase, in which there is a covalent addition of nucleotides to the 3' end of the DNA. This produces a short stretch of DNA that is single-stranded. Termination is the last phase of this process, in which the transcription termination sequence is recognized, the RNA polymerase is released and a poly-A tail is appended. The final product of this step is *primary mRNA* [Marketa and Jeremy, 2008].

2. Splicing: A gene contains coding regions known as *exons*, which are expressed (i.e. transcribed into mRNA), and intervening sequences, known as *introns*, which are not

[1] Figure source: http://en.wikipedia.org/wiki/File:Gene2-plain.svg, Date: 04/01/2011, License type: This work has been released into the public domain by its author.

expressed. Primary mRNA contains both exons and introns. Most introns start with the sequence GT, which is called *donor site* and end with the sequence AG, which is called *acceptor site*. Introns are removed from primary mRNA to form *mature mRNA*. This process is called *splicing* [Marketa and Jeremy, 2008] (see Section 2.1.2 for more details).

3. Translation: *Translation* is the final step in the information flow from gene to protein. During translation the mature mRNA is translated into protein. Three nucleotide of an RNA sequence code for one amino acid. The triplets are called *codons*. Translation has three important steps. Initiation is the first step, in which translation starts with a standard condon (AUG). This codon is called *start codon*. Elongation is the second step, amino acids get added to the elongating polypeptide chain in this step. Termination is the last step, in which translation stops at another standard sequence signal called *stop codon*, which is either UAA, UAG or UGA. Segments at both ends of the mRNA which are not translated are called *5' UTR* (UnTranslated Region) and *3'UTR*, respectively [Marketa and Jeremy, 2008].

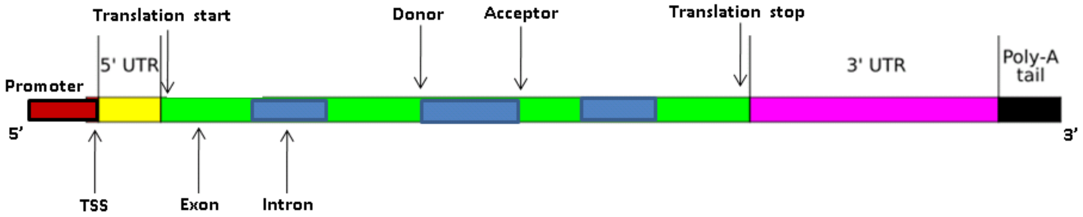


Figure 2.2: *A simple gene structure showing the position of sequence signals relevant to gene finding.*

A simple gene structure and the positions of sequence signals relevant to gene finding are shown in Figure 2.2. The promoter is a potentially long region upstream of the transcription start site (TSS). The region between TSS and translation start is referred to as 5' untranslated region (5' UTR). The figure also shows introns and exons, introns are bounded by the donor site on the 5' end and by acceptor site on the 3' end. Translation is terminated at translation stop, the region between translation stop and the cleavage site is called 3' UTR. After transcription, the poly-A tail is appended to the mRNA. The poly-A tail plays important role in mRNA stability.

In this work, the main focus is on recognition of splice sites. This problem is discussed in details in Section 2.1.2.

2.1.2 Splicing

Splicing is an important step in gene expression, as it is responsible for gene regulation, and protein diversity in eukaryote. We have already mentioned above that, during splicing, introns are removed from primary mRNA to produce mature mRNA. This process is quite complex. It involves several other proteins and five snRNPs (small nuclear ribonucleoprotein): U1,U2, U5, U4, and U6, each of these contain a small RNA bound by proteins [Douglas, 2003] (see Figure 2.3).

As soon as the primary mRNA is transcribed, it is bound by snRNPs. The snRNPs are responsible for splicing introns out of primary mRNA. They bind to sites of a primary mRNA at or near the intron-exon boundaries. These sites, called donor/acceptor sites, contain nucleotide sequences that are shared by most primary mRNAs. The donor/acceptor sites and snRNPs have complementary base pairing, so that snRNPs can bound to them [Nilsen, 1994].

The snRNPs attach not only to the consensus sequence but some of them attach to other sequences in the intron. These snRNPs group together into a large complex called a *spliceosome*. The intron loops out with the formation of the spliceosome. The spliceosome cuts the primary mRNA at one intron-exon boundary leaving a free hydroxyl (-OH) group on the exon. It uses this hydroxyl group to attack the other end of the intron, and in the

[2] Figure source: http://en.wikipedia.org/wiki/File:Two-step_Splicing_Reaction.png, Date: 04/01/2011, License type: GNU Free Documentation License.

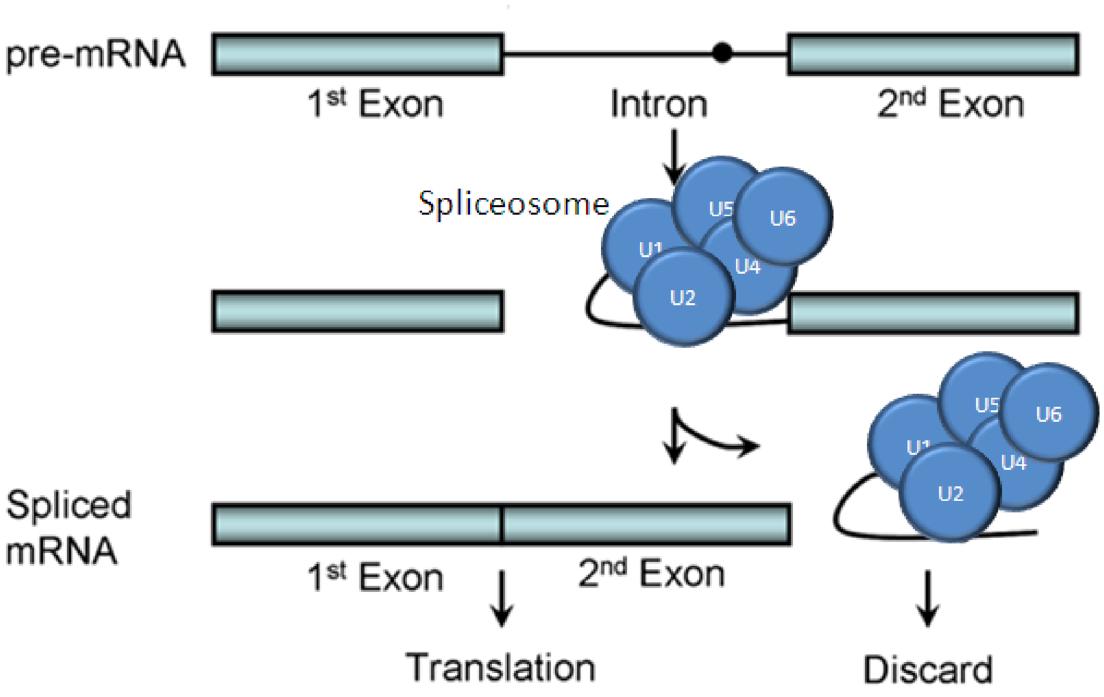


Figure 2.3: *Formation of the spliceosome during RNA splicing.*[2]

process removes the intron and joins the ends of two exons, producing a *mature mRNA* [Scott and Gilbert, 2006], [Collins and Guthrie, 1999].

Although introns are discarded, they do contain important sequences. snRNPs bind to the consensus sequences within the introns. The snRNPs use these sequences as markers to direct them to the correct splice sites. We make use of Machine Learning techniques to identify the splicing sites with the help of consensus sequence. We will discuss the Machine Learning techniques used in this work in the next section.

2.2 Machine Learning

Machine Learning is a branch of Artificial Intelligence which involves developing algorithms that can learn by repetition and experience just as humans learn [Mitchell, 1997]. It is widely used for classification tasks. A classification task in machine learning is defined as a method for assigning a label (or category) to an instance, from a number of categories. A simple example would be assigning a label as *spam* or *non-spam* for an email. Supervised learning is a class of machine learning techniques which are very popular for classification tasks. The main idea of supervised learning algorithms is to use externally supplied labeled instances to learn a general hypothesis, which can make prediction about new instances. The general hypothesis which is generated by a classification algorithm is called a classifier, which is usually a mathematical function or a probabilistic model. The classifier maps the unlabeled instances to labels. We have used Logistic Regression classifier and Support Vector Machine classifier for our classification task. We will discuss each of these in detail in the next sub-sections.

2.2.1 Logistic Regression

Logistic regression predicts the probability of occurrence of an event by fitting the data to a logistic curve [Kleinbaum et al., 1994].

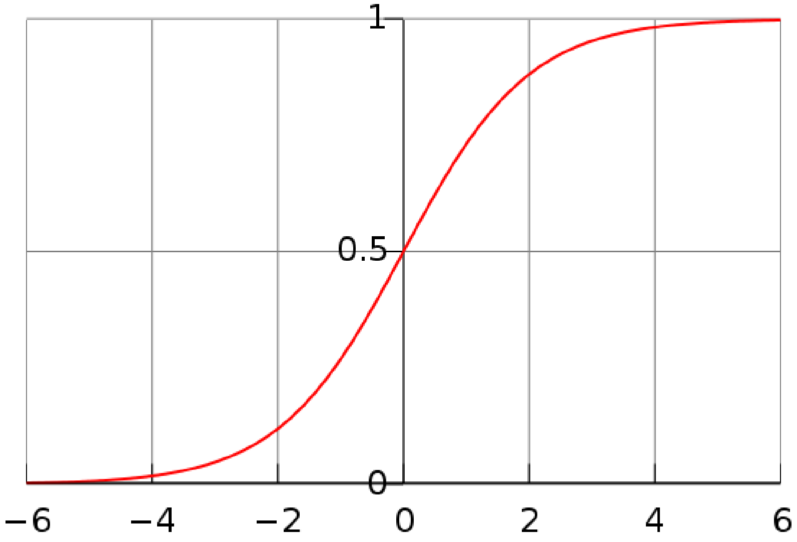


Figure 2.4: *The logistic function, with z on the horizontal axis and f(z) on the vertical axis. The variable z represents the exposure to some set of independent variables and f(z) represents the probability of a particular outcome*[3].

The logistic curve is shown in the Figure 2.4. The equation of this curve is given as:

$$f(z) = \frac{1}{1+e^{-z}} \tag{2.1}$$

where z variable represents the exposure to a set of independent variables and $f(z)$ represents the probability of particular outcome.

An important property of this curve is that its input value z can range from a negative number until infinity, but its output $f(z)$ can range only from 0 to 1. Furthermore, z is usually given as:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \cdots + \beta_k x_k, \tag{2.2}$$

For a classification task in machine learning $x_1, x_2, ..., x_k$ represent the features of train or test instances, i.e. they are the independent variables; $\beta_0, \beta_1, ..., \beta_k$ are called regression coefficients. The training instances are used to determine the values of these coefficients, who produce a logistic regression classifier. Once the value of these coefficients is known, unlabeled instances can easily be classified by using Equation (2.1). This equation tells us that what is the probability that a particular instance belongs to a given class.

2.2.2 Support Vector Machines

The *Support Vector Machine* (SVM) classifier works by finding a hyperplanes which separates points belonging to one class from the points belonging to another class. Figure 2.5 shows how points belonging to two classes can be separated by a hyperplane. However, there could be several hyperplanes separating these points. A good choice for classification is the maximum margin hyperplane (see Figure 2.6), which maximizes the distance from the nearest data points on each side [Cortes and Vapnik, 1995].

[3] Figure source: http://en.wikipedia.org/wiki/File:Logistic-curve.svg, Date: 04/01/2011, License type: This work has been released into the public domain by its author.

[4] Figure source: http://en.wikipedia.org/wiki/File:Svm_separating_hyperplanes.png, Date: 04/01/2011, License type: GNU Free Documentation License.

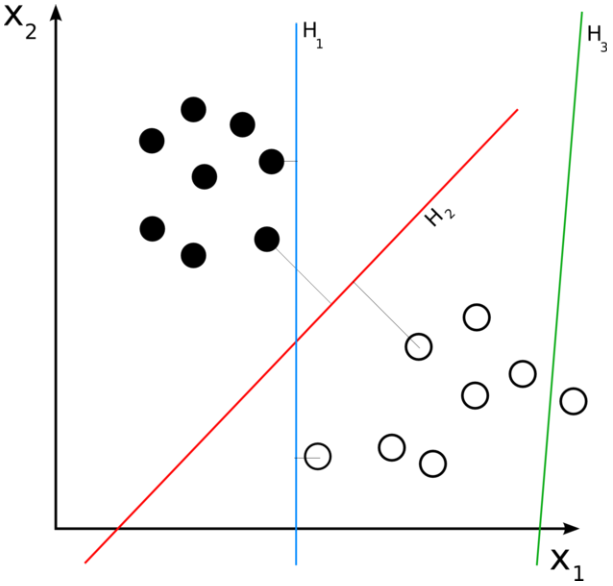


Figure 2.5: *Example of hyperplanes in a 2-dimensional space and a data set consisting of two classes. H3 (green) does not separate the 2 classes. H1 (blue) does, with a small margin and H2 (red) separates the two class with the maximum margin.*[4]

Formally, let us suppose that we have a set of training vectors belonging to two classes denoted by 1, -1. This set of data points will be denoted by:

$$D = \{(x_1, y_1), ...(x_m, y_m)|x_i \in R^n, y_i \in \{1, -1\}\} \tag{2.3}$$

A hyperplane in the R^n space can be represented as:

$$\mathbf{w} \cdot \mathbf{x} - b = 0 \tag{2.4}$$

In the above equation, w is a normal vector which is perpendicular to the hyperplane. The parameter $\frac{b}{\|\mathbf{w}\|}$ determines the offset of the hyperplane from the origin along the normal vector $\|\mathbf{w}\|$.

Parameters **w** and b are to be constrained, such that the margin between two classes is maximized, i.e. the parallel hyperplanes are as far as possible (see Figure 2.5). These hyperplanes can be described by the equations below:

$$\mathbf{w}\cdot\mathbf{x}-b=1 \tag{2.5}$$

and

$$\mathbf{w}\cdot\mathbf{x}-b=-1 \tag{2.6}$$

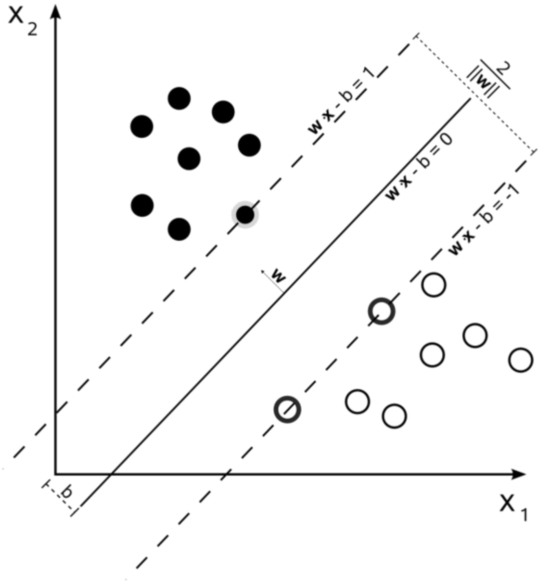


Figure 2.6: *Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called support vectors.*[5]

The distance between these two hyperplanes is given by $\frac{2}{\|\mathbf{w}\|}$ (see Figure 2.6). In order to maximize this distance, we need to minimize $\|\mathbf{w}\|^2$, with respect to $\mathbf{w}$ and b, subject to several constraints specified below.

Any of the points representing instances (see Equation (2.3)) must not lie in between the hyperplanes represented by Equation (2.5) and Equation (2.6)). Therefore for the first class we have:

$$\mathbf{w}\cdot\mathbf{x}_i-b\geq 1 \qquad \text{for } \mathbf{x}_i \text{ of the class 1} \tag{2.7}$$

[5] Figure source: http://en.wikipedia.org/wiki/File:Svm_max_sep_hyperplane_with_margin.png, Date: 04/01/2011, License type: GNU Free Documentation License.

And for the second class we have:

$$\mathbf{w} \cdot \mathbf{x}_i - b \leq -1 \qquad \text{for } \mathbf{x}_i \text{ of the class -1} \tag{2.8}$$

By simplifying Equations (2.7) and (2.8) we get:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1, \quad \text{for all } 1 \leq i \leq m \tag{2.9}$$

The solution for the above optimization problem is found by considering the dual problem and using the technique of Lagrange multipliers. The solution is represented as:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x_i} \tag{2.10}$$

where, α_i's are the non-negative Lagrange multipliers corresponding to the constraints in the primal problem.

Thus, given the training points we can determine the maximum-margin hyperplane in the training phase. Later, to classify an unlabeled instance we find its class by determining on which side of the separating hyperplane it is located. The SVM formulation above is for perfectly linearly separable data. If the data is almost linearly separable, a soft-margin SVM is used, which includes a term for error penalty in the optimization function [Cortes and Vapnik, 1995].

There are two important points to note here. First, we described SVM for two-class problems but it can be extended to multi-class classification problems, although some good properties of the binary classifier are lost, when extending SVM to the multi-class classifiers [Crammer et al., 2001] (we are only using binary SVM classifiers in this work). Second, the algorithm described above is for linear SVM classifier. However, it is possible to create a non-linear SVM classifier using the kernel trick[6].

[6] The kernel trick is used to convert any linear classifier into a non-linear classifier, given that the original linear classifier solely depends on dot products between two vectors. In the extension to the non-linear case, the dot product is replace by a the kernel function. [Cristianini and Shawe-Taylor, 2000], [Aizerman et al., 1964], [Boser et al., 1992]

2.3 Topic Modeling

We make use of topic modeling for dimensionality reduction for DNA sequence classification task. The first part of this section will give a general overview of generative topic models. The second part of this section will discuss Latent Dirichlet Allocation in detail, as it is the topic modeling scheme which we have used in our work.

2.3.1 Generative Topic Models

Topic models provide an easy way to analyze unlabeled text of very large volume. Words that occur together frequently are clustered together in a topic. Topic modeling can help us to group words with similar meaning, as well as distinguish between the same words with different meaning.

Topic models are establish on the idea that documents are mixtures of topics, while topics are mixtures of words. These models give us a simple probabilistic procedure to generate documents. Topic model are, thus, seen as generative models for a collection of documents. If we want to generate a document on the basis of a topic model, then we first choose a distribution over topics. After that, we randomly select a topic according to the distribution and generate a word according to the distribution over words. This process is repeated to generate each word in the document and, therefore, a complete document is generated. By reversing this process we can infer the set of topics which generated a document corpus [Steyvers and Griffiths, 2007].

Probabilistic sampling rules, which describe how words in documents might be generated based on latent (random) variables, form the basis of generative models. In order to find a model that produced a document corpus, we need to find the best set of latent variables which can justify the observed data. This is done with the assumption that the model actually generated the data. The left side of Figure 2.7 shows how a document is generated given a topic model. In this example we have two topics, which generated three documents based on the known topic distributions. The right side of the Figure 2.7 shows how topic

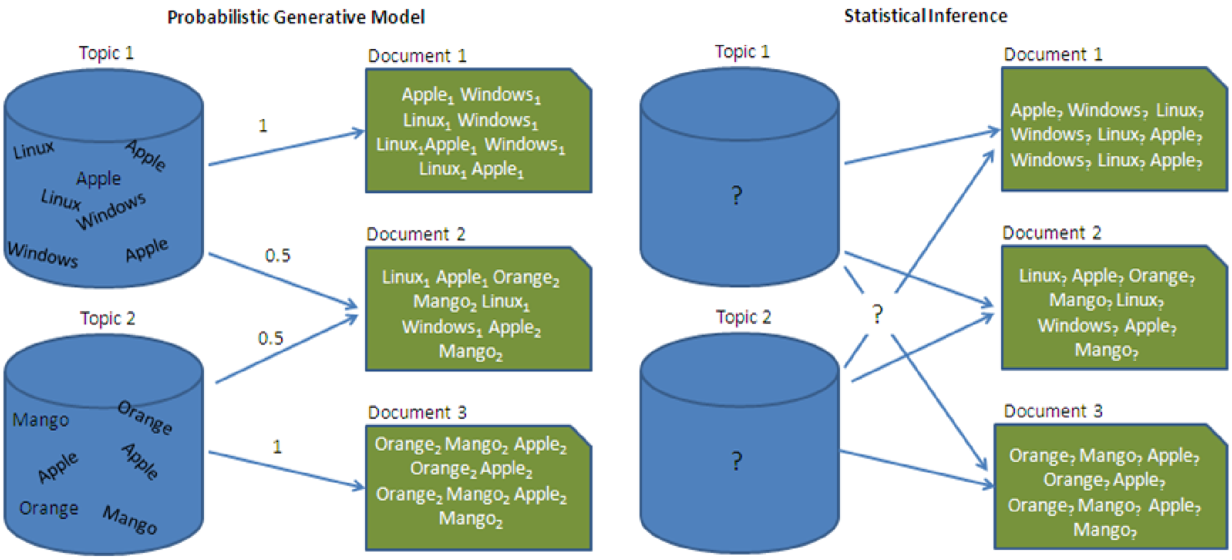

Probabilistic Generative Model
Statistical Inference
Topic 1
Linux
Apple
Apple
Linux
Windows
Windows
Apple
Topic 2
Mango
Orange
Apple
Apple
Orange
Mango
1
0.5
0.5
1
Document 1
Apple$_1$ Windows$_1$ Linux$_1$ Windows$_1$ Linux$_1$Apple$_1$ Windows$_1$ Linux$_1$ Apple$_1$
Document 2
Linux$_1$ Apple$_1$ Orange$_2$ Mango$_2$ Linux$_1$ Windows$_1$ Apple$_2$ Mango$_2$
Document 3
Orange$_2$ Mango$_2$ Apple$_2$ Orange$_2$ Apple$_2$ Orange$_2$ Mango$_2$ Apple$_2$ Mango$_2$
Topic 1
?
Topic 2
?
?
Document 1
Apple$_?$ Windows$_?$ Linux$_?$ Windows$_?$ Linux$_?$ Apple$_?$ Windows$_?$ Linux$_?$ Apple$_?$
Document 2
Linux$_?$ Apple$_?$ Orange$_?$ Mango$_?$ Linux$_?$ Windows$_?$ Apple$_?$ Mango$_?$
Document 3
Orange$_?$ Mango$_?$ Apple$_?$ Orange$_?$ Apple$_?$ Orange$_?$ Mango$_?$ Apple$_?$ Mango$_?$

Figure 2.7: *Illustration of the generative process and the problem of statistical inference underlying topic models (this figure is adapted from [Steyvers and Griffiths, 2007]).*

modeling can be viewed as a problem of statistical inference. Here, we are given three documents and we need to find the best model (i.e. topics or set of latent variables) that might have generated these documents. Note that the model shown in this figure also captures polysemy, by allowing a same word to appear in multiple topics. The word bank appears in both topics and the subscript in the figure helps to distinguish between bank from topic 1 and bank from topic 2. Another important thing to note here is that this model makes the *bag of word assumption* [Blei et al., 2003], i.e. this model does not take into consideration the order in which the words appear in documents.

2.3.2 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a generative probabilistic model which is used to model any collections of discrete data [Blei et al., 2003]. A text document corpus can be seen as a collection of discrete data, therefore LDA can be used to model a document corpus. LDA was first introduced by Blei et al. [2003]. They defined LDA as "A three-level hierarchical Bayesian model, in which each item of a collection is modeled as a finite mixture over an underlying set of topics. Each topic is, in turn, modeled as an infinite mixture over an underlying set of topic probabilities". To fit a document corpus to this definition, we can view a document as a collection of topics and each topic is viewed as a collection of words. The paper Blei et al. [2003] presented an efficient method from which given a collection of documents, it is possible to approximate the parameters of the model representing this corpus, i.e. we can estimate topic probabilities and word probabilities within a topic.

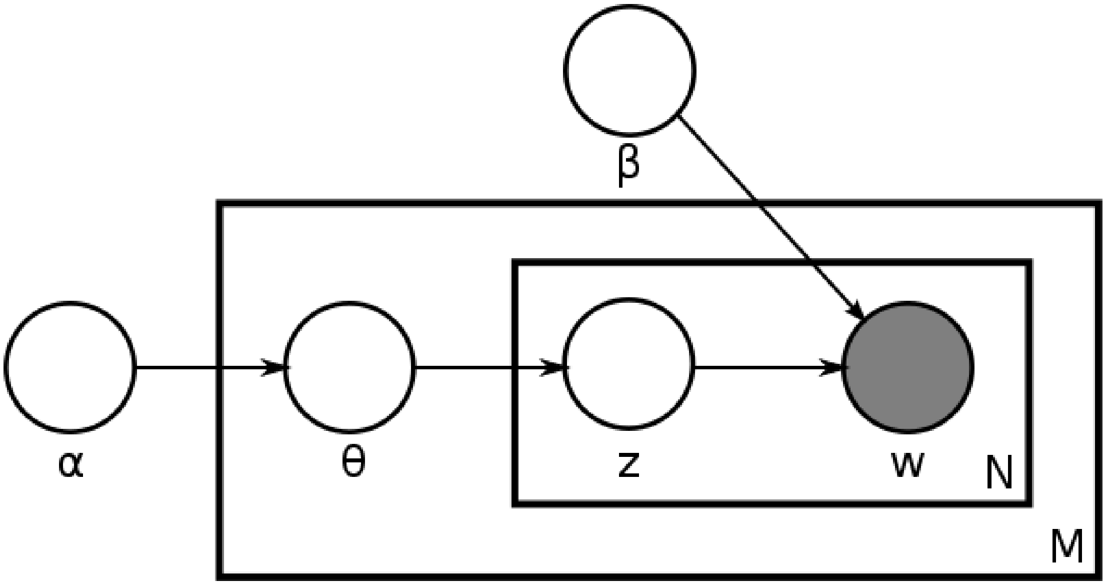


Figure 2.8: *Graphical model representation of LDA. The boxes are "plate" representing replicates. The outer plate represents documents, while the inner plate represents the repeated choice of topics and words within a document.* [7]

According to Blei et al. [2003], LDA assumes the following generative process for each document **w** in a corpus D.

1. Choose $N \sim$ Poisson(ξ).

2. Choose $\theta \sim \mathit{Dirichlet}(\alpha)$.

3. For each of the N words w_n in document $\mathbf{w}$:

[7] Figure source: http://en.wikipedia.org/wiki/File:Latent_Dirichlet_allocation.svg, Date: 04/01/2011, Licence type: GNU Free Documentation License.

(a) Choose a topic $z_n \sim \mathit{Multinomial}(\theta)$.

(b) Choose a word w_n from $p(w_n \mid z_n, \beta)$, a multinomial probability conditioned on the topic z_n.

Some of the assumptions that we make to obtain a generative probabilistic LDA model of a corpus are as follows. Firstly, we assume that the dimensionality k of the Dirichlet distribution hence the dimensionality of topic variable z is known and fixed. Second, we assume that the word probabilities are parameterized by a kXV matrix where $\beta_i j = p(w^j = 1|z^i = 1)$, which is treated as a fixed quantity that is to be estimated.

A k-dimensional Dirichlet random variable θ which lies in the $(k-1)$-simplex has the following probability density on this simplex:

$$p(\theta|\alpha) = \frac{\Gamma(\varepsilon_{i=1}^{k}\alpha_i)}{\prod_{i=1}^{k}\Gamma(\alpha_i)}\theta_1^{\alpha_1-1}...\theta_k^{\alpha_k-1} \tag{2.11}$$

where α is a k-vector with components $\alpha_i > 0$, and Γ is the Gamma function.

α and β are corpus level parameters, which are sampled only once during the process of corpus generation. θ_d is a document-level variable and it is sampled once per document (θ_d is the topic distribution for document d). N is independent of all the other data generating variables (θ and $\mathbf{z}$). Therefore, it is an ancillary variable and we will generally ignore its randomness. For given values of α and β, we can find a joint distribution of a topic mixture θ, a set of N topics $\mathbf{z}$, and a set of N words $\mathbf{w}$ as:

$$p(\theta, \mathbf{z}, \mathbf{w} \mid \alpha, \beta) = p(\theta \mid \alpha)\prod_{n=1}^{N} p(z_n \mid \theta)p(w_n \mid z_n, \beta) \tag{2.12}$$

In the above equation $p(z_n \mid \theta)$ is θ_i for a unique i such that $z_n^i = 1$. If we integrate over θ and then sum over z, then we get:

$$p(\mathbf{w} \mid \alpha, \beta) = \int p(\theta \mid \alpha)\left(\prod_{n=1}^{N}\sum_{z_n} p(z_n \mid \theta)p(w_n \mid z_n, \beta)\right)d\theta \tag{2.13}$$

Finally, taking the product of the marginal probabilities of single documents, we obtain the probability of a corpus:

$$p(D \mid \alpha, \beta) = \prod_{d=1}^{M} \int p(\theta_d \mid \alpha) \left(\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn} \mid \theta_d) p(w_{dn} \mid z_{dn}, \beta) \right) d\theta_d \tag{2.14}$$

The probabilistic graphical model representation of LDA is shown in Figure 2.8. It is clear from the figure that LDA has three levels. Finally, the variables z_{dn} and w_{dn} are word-level variables, which are sampled once for each word in each document. This is how, under LDA, documents can be associated with multiple topics and each topic is associated with multiple words.

LDA can find a short description of a large collection of documents. As a result, LDA can be used for processing large collections, while preserving the essential statistical relationships that are useful for document classification [Blei et al., 2003]. In our work, we used LDA to model our sequence collection (which is the equivalent of the document corpus). This enabled us to obtain a set of "topics" and set of "words" representing our sequence collection. The set of words obtained from modeling the sequence collection with LDA is much smaller than the actual number of words in the sequence collection. We only used the set of "important" words (words with high probability in a topic) obtained from LDA as features for representing each document in our sequence collection. This is how, LDA was used for dimensionality reduction.

Several techniques are proposed to estimate the parameters of LDA model, α and β, given a document corpus. Empirical Bayes approach [Blei et al., 2003] and Gibbs sampling [Steyvers and Griffiths, 2007] are two popular approaches. In the implementation that we used, MALLET: A Machine Learning Toolkit [McCallum, 2002], Gibbs sampling is used for inference.

Chapter 3

Related Work

This chapter reviews a number of works done in the past, which are related to the objectives of this thesis. Section 3.1 presents previous works on techniques employed for feature generation and selection. Section 3.2 describes the techniques employed in the past for solving the RNA splicing problem. Section 3.3 presents previous work related to domain adaptation techniques for biological data. Finally, in Section 3.4 we discuss some of the applications of topic modeling.

3.1 Dimensionality Reduction

Dimensionality reduction techniques for text data have been studied extensively. Some of the previous works which presented dimensionality reduction techniques for text data are [Liu and Motoda, 1998], [Koller and Sahami, 1996], [Yu and Liu, 2003] and [Tasci and Gungor, 2009]. Liu and Motoda [1998] cover all the basic concepts related to feature generation and feature selection. Koller and Sahami [1996] examine an approach for dimensionality reduction based on information theory. Yu and Liu [2003] introduce a novel concept of predominant correlation, and present a fast filter method which can identify relevant features as well as redundancy among relevant features without pairwise correlation analysis.The work presented by Tasci and Gungor [2009] uses LDA for dimensionality reduction and selection of key features. Their work suggests that the performance of LDA dimensionality reduction technique was not better than that of the Information Gain technique (for details of Information Gain technique refer to [McCallum and Nigam, 1998]). However, they obtain a 75% reduction in corpus size without any loss in performance by using these features.

Lately, feature extraction for biological data has also gained attention from several researchers. Some of the feature extraction techniques used for the mRNA splicing site prediction problem are presented in [Yeo and Burge, 2004], [Zhang et al., 2003a] and [Degroeve et al., 2002]. Yeo and Burge [2004] model sequence motifs based on the Maximum Entropy principle (MEP). Zhang et al. [2003a] use SVM algorithm to discover sequence information (motifs/features), which is further used for classification. Degroeve et al. [2002] present a wrapper-based feature subset selection algorithm. Although, LDA had been used to solve some biological problem (see Section 3.4), to the best of our knowledge, LDA has not been used for dimensionality reduction on biological sequences.

3.2 The mRNA Splicing Prediction Problem

In the past few decades the genome sequences of many organisms have become available. Due to the availability of labeled data, it has been possible to use supervised machine learning techniques to automate the splice site prediction process. A comprehensive overview of splicing site recognition can be found in [Sonnenburg, 2002] and a general overview along with a comparison of several gene and splice site prediction methods can be found in [Mathe et al., 2002] and [Zhang, 2002]. Among several supervised machine learning techniques that have been used for splicing site prediction, are methods based on Maximum Entropy Modeling [Yeo and Burge, 2004] and Support Vector Machines (SVM) [Zhang et al., 2003b]. SVMs along with related kernel methods are found to be very effective for solving gene prediction problems [Boser et al., 1992], [Scholkopf and Smola, 2002]. The main reason for the SVM popularity in computational biology is grounded ability to handle high-dimensional spaces and large data sets [Scholkopf et al., 2004], [Scholkopf et al., 2005]. Some other techniques used for solving the mRNA splicing site problem are based on genetic programming [Vukusic et al., 2007] and artificial neural networks combined with a rule based system [Hebsgaard et al., 1996]. Some researches also used micro-arrays for mRNA splicing site prediction problem [Johnson et al., 2003], [Zheng et al., 2004].

3.3 Transfer Learning Techniques for Biological Data

All the methods mentioned in Section 3.2 are based on supervised machine learning. As a result, these methods require a large amount of labeled data (already annotated instances) to learn models, which can be then used to make predictions for the unknown instances in the rest of the genome. Semi-supervised learning may be useful in the situation when we have a small amount of labeled data and large amount of unlabeled data available. However, we may also have a large amount of labeled data from a related domain, which may have a different feature representation or data distribution. Transfer learning tries to find the similarity and relatedness of different domains with the goal of learning what can be transfered from a source domain to a target domain. In recent years, it has been used to solve bioinformatics problems. Protein name extraction [Arnold et al., 2007] and micro-array data classification [Widmer et al., 2010] are among the few bioinformatics problems for which researchers have used domain adaptation techniques.

Some very recent studies have also focused on domain adaptation techniques for the gene prediction problem [Schweikert et al., 2009; Widmer et al., 2010]. These techniques are based on the assumption that the cellular mechanisms that are responsible for transcription and translation of genes are conserved between organisms. Therefore, it should be possible to use the knowledge from an organism, which has a large number of labeled genes to make predication about an organism which has a very small number of labeled genes. In the work done by Schweikert et al. [2009], they consider different domain adapation methods and evaluate them on genomic sequence data from model organisms of varying evolutionary distances. In this work, the source organism used was *C. elegans*. In 1963, Sydney Brenner introduced *Caenorhabditis elegans* (*C. elegans*) as a model organism. *C. elegans* is a free-living, non-parasitic soil nematode that can be safely used in the laboratory and is common around the world. It is 1mm in length and can be cheaply housed and cultivated in large numbers (10,000 worms/petri dish) [Riddle et al., 1997]. All these properties made many researchers to work on this organism. Consequently, large amounts of labeled data are available for this organism, making it an ideal choice as a source organism. Convex combination of source and target data, weighted combination of source and target data, multitask learning, kernel mean matching are some of the domain adaptation algorithms considered by Schweikert et al. [2009],. The results in [Schweikert et al., 2009] showed that the use of domain adaption improves classification performance in cases where the organisms are not closely related.

One of the factors that contributed to the success of the approach in [Schweikert et al., 2009] is the use of the weighted degree kernel, which works by counting matching subsequences between two sequences. Due to the use of the weighted degree kernel [Fodor, 2002], they do not have to represent the data in a high dimensional space. We are also using the same dataset as used by Schweikert et al. [2009]. However, as an alternative approach to dimensionality reduction, we propose to use Latent Dirichlet Allocation (LDA) model to generate features for our experiments. In what follows, we review some of the previous work done based on LDA model.

3.4 Topic Modeling Applications

LDA is a generative probabilistic model for collections of discrete data such as text corpora. It was originally used for document modeling, text classification and collaborative filtering [Blei et al., 2003]. LDA has been used for many different applications after it was used by Blei et al. [2003] for document modeling. Among others, it has been used for various problems involving text data like tag recommendation [Krestel et al., 2009], word sense disambiguation [Boyd-Graber et al., 2007], named entity recognition [Guo et al., 2009], friendship link prediction problem [Parimi, 2010], etc. There have been some attempts in the past to use LDA for biological data. Items in a biomedical text corpus were modeled using the Latent Dirichlet Allocation (LDA) model by Blei et al. [2006]. An adapted version of LDA, called Latent Process Decomposition (LPD), which can explicitly model expression levels, was proposed by Rogers et al. [2005]. This work used LPD for clustering expression micro-array data. Biologically-aware Latent Dirichlet Allocation (BaLDA), which extends LDP, was introduced and used by Perina et al. [2010] for classification of expression micro-array data. However, to the best of our knowledge, LDA has not been used for feature extraction for the RNA splicing site prediction problem or for any other prediction problem where DNA sequences are used.

Chapter 4

Problem Definition and Approach

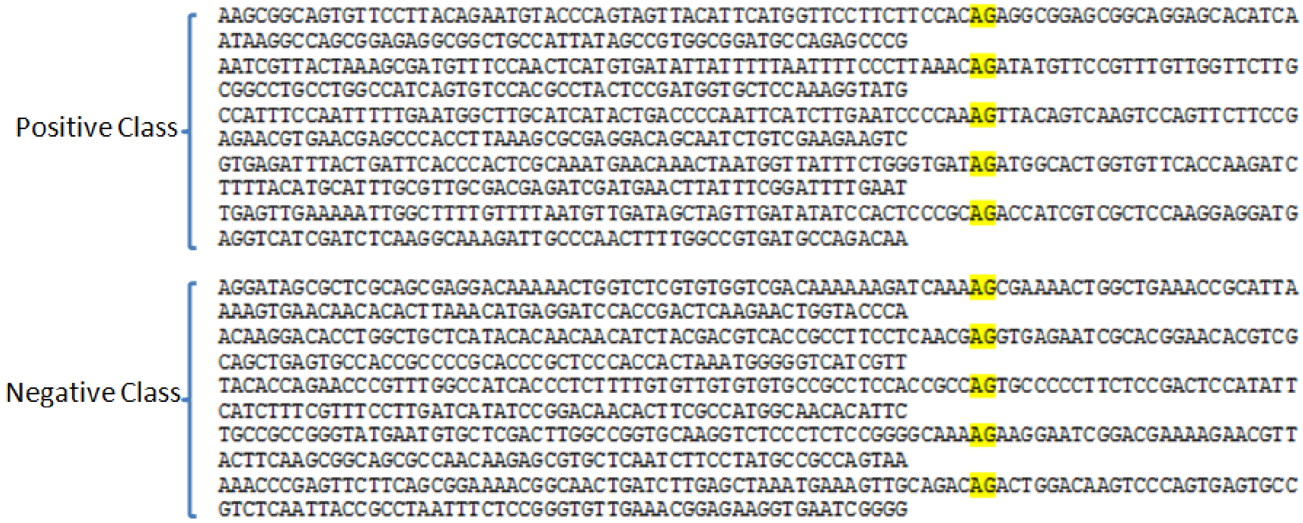
We begin this chapter by describing the splice site prediction problem. Then, we present LDAW and LDAD approach considered in the first part of this work. Finally, we describe all the Domain Adaptation algorithms which we used for the second part of this work.

4.1 Splice Site Prediction Problem

We will be validating the performance of our approaches on the splice site prediction problem. We have already described the gene structure and splicing in Section 2.1.1 and Section 2.1.2, respectively. We discussed in these sections that there is a consensus sequence surrounding the acceptor site (GT) and the donor site (AG) to which the snRNPs bind during splicing. Due to presence of consensus it is possible to use machine learning algorithms to predict splice site. We will now discuss the details of this problem.

The donor site is identified by a GT sequence and acceptor site is identified by AG. However, this GT-AG rule does not always hold. Therefore, we can model this problem as a binary classification problem. The sequences with experimentally confirmed splice sites are our positive examples and the sequences confirmed as not real splice site are our negative examples. Given a DNA sequence surrounding a donor or acceptor site, our goal is to predict if it is a real splice site or not.

Figure 4.1 shows examples from our dataset of both positive and negative instances. Each instance has a possible acceptor site as it contain the AG sequence. However, only the instances in the positive class are real acceptor sites. Although the instances in negative class have AG sequence, still they are not real acceptor sites. In our dataset, each instance is composed of 60pb before and 79 bp after the possible donor site (AG). The information surrounding the AG sequence is used to discriminate between the real acceptor site and decoy acceptor site.


Positive Class
Negative Class

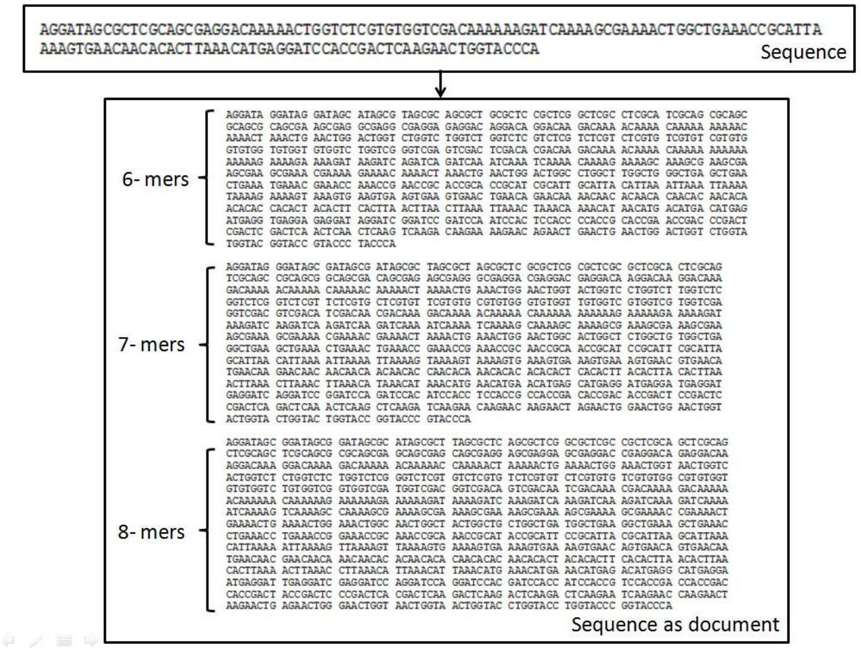
```
AAGCGGCAGTGTTCCTTACAGAATGTACCCAGTAGTTACATTCATGGTTCCTTCTTCCACAGAGGCGGAGCGGCAGGAGCACATCA
ATAAGGCCAGCGGAGAGGCGGCTGCCATTATAGCCGTGGCGGATGCCAGAGCCCG
AATCGTTACTAAAGCGATGTTTCCAACTCATGTGATATTATTTTTAATTTTCCCTTAAACAGATATGTTCCGTTTGTTGGTTCTTG
CGGCCTGCCTGGCCATCAGTGTCCACGCCTACTCCGATGGTGCTCCAAAGGTATG
CCATTTCCAATTTTTGAATGGCTTGCATCATACTGACCCCAATTCATCTTGAATCCCCAAAGTTACAGTCAAGTCCAGTTCTTCCG
AGAACGTGAACGAGCCCACCTTAAAGCGCGAGGACAGCAATCTGTCGAAGAAGTC
GTGAGATTTACTGATTCACCCACTCGCAAATGAACAAACTAATGGTTATTTCTGGGTGATAGATGGCACTGGTGTTCACCAAGATC
TTTTACATGCATTTGCGTTGCGACGAGATCGATGAACTTATTTCGGATTTTGAAT
TGAGTTGAAAAATTGGCTTTTGTTTTAATGTTGATAGCTAGTTGATATATCCACTCCCGCAGACCATCGTCGCTCCAAGGAGGATG
AGGTCATCGATCTCAAGGCAAAGATTGCCCAACTTTTGGCCGTGATGCCAGACAA

AGGATAGCGCTCGCAGCGAGGACAAAAACTGGTCTCGTGTGGTCGACAAAAAAGATCAAAAGCGAAAACTGGCTGAAACCGCATTA
AAAGTGAACAACACACTTAAACATGAGGATCCACCGACTCAAGAACTGGTACCCA
ACAAGGACACCTGGCTGCTCATACACAACAACATCTACGACGTCACCGCCTTCCTCAACGAGGTGAGAATCGCACGGAACACGTCG
CAGCTGAGTGCCACCGCCCCGCACCCGCTCCCACCACTAAATGGGGGTCATCGTT
TACACCAGAACCCGTTTGGCCATCACCCTCTTTTGTGTTGTGTGTGCCGCCTCCACCGCCAGTGCCCCCTTCTCCGACTCCATATT
CATCTTTCGTTTCCTTGATCATATCCGGACAACACTTCGCCATGGCAACACATTC
TGCCGCCGGGTATGAATGTGCTCGACTTGGCCGGTGCAAGGTCTCCCTCTCCGGGGCAAAAGAAGGAATCGGACGAAAAGAACGTT
ACTTCAAGCGGCAGCGCCAACAAGAGCGTGCTCAATCTTCCTATGCCGCCAGTAA
AAACCCGAGTTCTTCAGCGGAAAACGGCAACTGATCTTGAGCTAAATGAAAGTTGCAGACAGACTGGACAAGTCCCAGTGAGTGCC
GTCTCAATTACCGCCTAATTTCTCCGGGTGTTGAAACGGAGAAGGTGAATCGGGG
```

Figure 4.1: *Samples from positive and negative classes. The 3' splice site or splice acceptor site (AG) is highlighted in yellow.*

4.2 LDAW and LDAD Approaches to Feature Generation and Construction

We have already presented a brief overview of our approaches in Chapter 1. As discussed in Chapter 1, we need to represented biological sequences as feature vector before applying any classification algorithm on them. A simple way to do this would be to find all the k-mers and use each k-mer as a feature. However, this method will produce a large number of features, which would be difficult to work with when learning. Therefore, it is desirable to perform dimensionality reduction on the set of k-mers. We are using LDA to select the best features from the set of all k-mers representing the sequences.



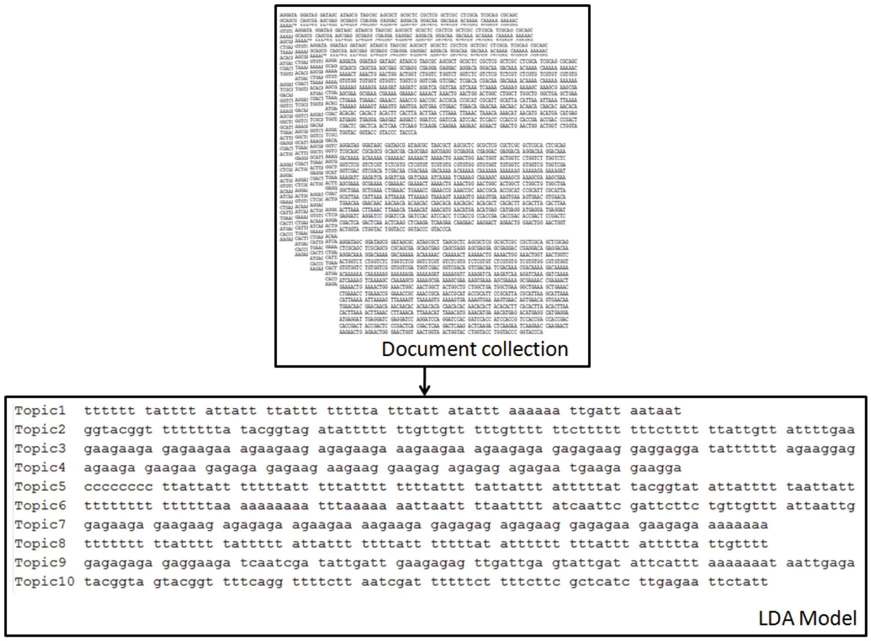
```
AGGATAGCGCTCGCAGCGAGGACAAAAACTGGTCTCGTGTGGTCGACAAAAAAGATCAAAAGCGAAAACTGGCTGAAACCGCATTA
AAAGTGAACAACACACTTAAACATGAGGATCCACCGACTCAAGAACTGGTACCCA
```

```
AGGATA GGATAG GATAGC ATAGCG TAGCGC AGCGCT GCGCTC CGCTCG GCTCGC CTCGCA TCGCAG CGCAGC
GCAGCG CAGCGA AGCGAG GCGAGG CGAGGA GAGGAC AGGACA GGACAA GACAAA ACAAAA CAAAAA AAAAAC
AAAACT AAACTG AACTGG ACTGGT CTGGTC TGGTCT GGTCTC GTCTCG TCTCGT CTCGTG TCGTGT CGTGTG
GTGTGG TGTGGT GTGGTC TGGTCG GGTCGA GTCGAC TCGACA CGACAA GACAAA ACAAAA CAAAAA AAAAAA
AAAAAG AAAAGA AAAGAT AAGATC AGATCA GATCAA ATCAAA TCAAAA CAAAAG AAAAGC AAAGCG AAGCGA
AGCGAA GCGAAA CGAAAA GAAAAC AAAACT AAACTG AACTGG ACTGGC CTGGCT TGGCTG GGCTGA GCTGAA
CTGAAA TGAAAC GAAACC AAACCG AACCGC ACCGCA CCGCAT CGCATT GCATTA CATTAA ATTAAA TTAAAA
TAAAAG AAAAGT AAAGTG AAGTGA AGTGAA GTGAAC TGAACA GAACAA AACAAC ACAACA CAACAC AACACA
ACACAC CACACT ACACTT CACTTA ACTTAA CTTAAA TTAAAC TAAACA AAACAT AACATG ACATGA CATGAG
ATGAGG TGAGGA GAGGAT AGGATC GGATCC GATCCA ATCCAC TCCACC CCACCG CACCGA ACCGAC CCGACT
CGACTC GACTCA ACTCAA CTCAAG TCAAGA CAAGAA AAGAAC AGAACT GAACTG AACTGG ACTGGT CTGGTA
TGGTAC GGTACC GTACCC TACCCA
```

```
AGGATAG GGATAGC GATAGCG ATAGCGC TAGCGCT AGCGCTC GCGCTCG CGCTCGC GCTCGCA CTCGCAG
TCGCAGC CGCAGCG GCAGCGA CAGCGAG AGCGAGG GCGAGGA CGAGGAC GAGGACA AGGACAA GGACAAA
GACAAAA ACAAAAA CAAAAAC AAAAACT AAAACTG AAACTGG AACTGGT ACTGGTC CTGGTCT TGGTCTC
GGTCTCG GTCTCGT TCTCGTG CTCGTGT TCGTGTG CGTGTGG GTGTGGT TGTGGTC GTGGTCG TGGTCGA
GGTCGAC GTCGACA TCGACAA CGACAAA GACAAAA ACAAAAA CAAAAAA AAAAAAG AAAAAGA AAAAGAT
AAAGATC AAGATCA AGATCAA GATCAAA ATCAAAA TCAAAAG CAAAAGC AAAAGCG AAAGCGA AAGCGAA
AGCGAAA GCGAAAA CGAAAAC GAAAACT AAAACTG AAACTGG AACTGGC ACTGGCT CTGGCTG TGGCTGA
GGCTGAA GCTGAAA CTGAAAC TGAAACC GAAACCG AAACCGC AACCGCA ACCGCAT CCGCATT CGCATTA
GCATTAA CATTAAA ATTAAAA TTAAAAG TAAAAGT AAAAGTG AAAGTGA AAGTGAA AGTGAAC GTGAACA
TGAACAA GAACAAC AACAACA ACAACAC CAACACA AACACAC ACACACT CACACTT ACACTTA CACTTAA
ACTTAAA CTTAAAC TTAAACA TAAACAT AAACATG AACATGA ACATGAG CATGAGG ATGAGGA TGAGGAT
GAGGATC AGGATCC GGATCCA GATCCAC ATCCACC TCCACCG CCACCGA CACCGAC ACCGACT CCGACTC
CGACTCA GACTCAA ACTCAAG CTCAAGA TCAAGAA CAAGAAC AAGAACT AGAACTG GAACTGG AACTGGT
ACTGGTA CTGGTAC TGGTACC GGTACCC GTACCCA
```

```
AGGATAGC GGATAGCG GATAGCGC ATAGCGCT TAGCGCTC AGCGCTCG GCGCTCGC CGCTCGCA GCTCGCAG
CTCGCAGC TCGCAGCG CGCAGCGA GCAGCGAG CAGCGAGG AGCGAGGA GCGAGGAC CGAGGACA GAGGACAA
AGGACAAA GGACAAAA GACAAAAA ACAAAAAC CAAAAACT AAAAACTG AAAACTGG AAACTGGT AACTGGTC
ACTGGTCT CTGGTCTC TGGTCTCG GGTCTCGT GTCTCGTG TCTCGTGT CTCGTGTG TCGTGTGG CGTGTGGT
GTGTGGTC TGTGGTCG GTGGTCGA TGGTCGAC GGTCGACA GTCGACAA TCGACAAA CGACAAAA GACAAAAA
ACAAAAAA CAAAAAAG AAAAAAGA AAAAAGAT AAAAGATC AAAGATCA AAGATCAA AGATCAAA GATCAAAA
ATCAAAAG TCAAAAGC CAAAAGCG AAAAGCGA AAAGCGAA AAGCGAAA AGCGAAAA GCGAAAAC CGAAAACT
GAAAACTG AAAACTGG AAACTGGC AACTGGCT ACTGGCTG CTGGCTGA TGGCTGAA GGCTGAAA GCTGAAAC
CTGAAACC TGAAACCG GAAACCGC AAACCGCA AACCGCAT ACCGCATT CCGCATTA CGCATTAA GCATTAAA
CATTAAAA ATTAAAAG TTAAAAGT TAAAAGTG AAAAGTGA AAAGTGAA AAGTGAAC AGTGAACA GTGAACAA
TGAACAAC GAACAACA AACAACAC ACAACACA CAACACAC AACACACT ACACACTT CACACTTA ACACTTAA
CACTTAAA ACTTAAAC CTTAAACA TTAAACAT TAAACATG AAACATGA AACATGAG ACATGAGG CATGAGGA
ATGAGGAT TGAGGATC GAGGATCC AGGATCCA GGATCCAC GATCCACC ATCCACCG TCCACCGA CCACCGAC
CACCGACT ACCGACTC CCGACTCA CGACTCAA GACTCAAG ACTCAAGA CTCAAGAA TCAAGAAC CAAGAACT
AAGAACTG AGAACTGG GAACTGGT AACTGGTA ACTGGTAC CTGGTACC TGGTACCC GGTACCCA
```

Figure 4.2: *Illustration of the process of obtaining the set of k-mers from a biological sequence. The biological sequence is taken from the same dataset which we have used in our work.*



```
Topic1  tttttt tatttt attatt ttattt ttttta tttatt atattt aaaaaa ttgatt aataat
Topic2  ggtacggt tttttta tacggtag atatttt ttgttgtt tttgtttt ttcttttt tttctttt ttattgtt attttgaa
Topic3  gaagaaga gagaagaa agaagaag agagaaga aagaagaa agaagaga gagagaag gaggagga tatttttt agaaggag
Topic4  agaaga gaagaa gagaga gagaag aagaag gaagag agagag agagaa tgaaga gaagga
Topic5  cccccccc ttattatt tttttatt tttatttt ttttattt tattattt atttttat tacggtat attatttt taattatt
Topic6  tttttttt tttttaa aaaaaaaa tttaaaaa aattaatt ttaatttt atcaattc gattcttc tgttgttt attaattg
Topic7  gagaaga gaagaag agagaga agaagaa aagaaga gagagag agagaag gagagaa gaagaga aaaaaaa
Topic8  ttttttt ttatttt tattttt attattt ttttatt tttttat attttttt tttattt attttta ttgtttt
Topic9  gagagaga gaggaaga tcaatcga tattgatt gaagagag ttgattga gtattgat attcattt aaaaaaat aattgaga
Topic10 tacggta gtacggt tttcagg ttttctt aatcgat tttttct tttcttc gctcatc ttgagaa ttctatt
```

Figure 4.3: *Illustration of the process of obtaining the LDA model from a document collection. The LDA model has 10 topics with 10 words (k-mers) in each topic.*

Our first step for feature generation was to find the set of all possible k-mers for a given biological sequence. The biological sequences in our dataset are of length 141bp and we represented each sequence using 6-mers, 7-mers and 8-mers. Figure 4.2 shows an example from our dataset depicting how a sequence can be represented as a collection of k-mers. Each set of k-mers corresponding to a sequence represents a "document" associated with the sequence. The next step in our approach is to find the LDA features by topic modeling this document collection. This step is shown in Figure 4.3. The order of the k-mers does not matter in a document as LDA makes the *bag of words* [Blei et al., 2003] assumption.

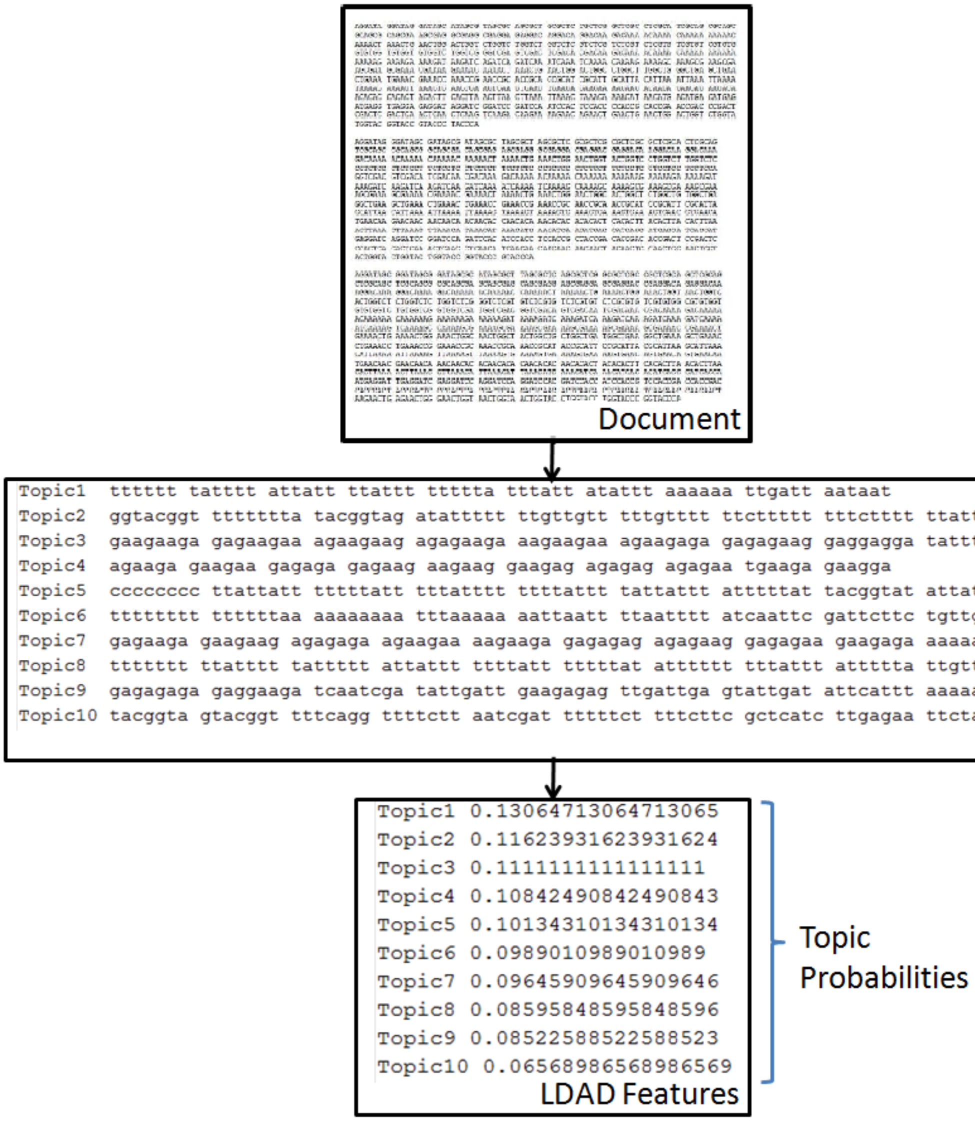
As mentioned earlier in Chapter 1, we used two approaches for dimensionality reduction: (a) LDA words (LDAW) (b) LDA distribution (LDAD). We will now discuss each of these approaches in details.

4.2.1 LDAW Approach

In LDAW, we use all the k-mers (with high probability) obtained by modeling the document collection as our features. Figure 4.4 illustrates the process of representing a test or a training document using LDAW features. To represent a document, we simply count the number of occurrences of each word (k-mer) from each LDA topic. Thus, for an LDA model represented by 10 topics with 10 words (k-mers) in each topic, we will have 100 features (unless there is word overlap between topics). An important point to note is that, we can control the number of LDA features, by controlling the number of topics and the number of words shown in a topic used for modeling our document collection. For example, if we set the number of topics as 10 and number of words in each topic as 10, then after modeling the documents using LDA we will be left with only 100 LDAW features.

4.2.2 LDAD Approach

In LDAD we use the topic distribution for each topic as our features. Figure 4.5 illustrates how we can represent a training or a test document using LDAD features. A document is represented by the topic distribution of each topic obtained from LDA model. Therefore,



```
Topic1  tttttt tatttt attatt ttattt ttttta tttatt atattt aaaaaa ttgatt aataat
Topic2  ggtacggt tttttta tacggtag atattttt ttgttgtt tttgtttt ttcttttt tttctttt ttattgtt attttgaa
Topic3  gaagaaga gagaagaa agaagaag agagaaga aagaagaa agaagaga gagagaag gaggagga tatttttt agaaggag
Topic4  agaaga gaagaa gagaga gagaag aagaag gaagag agagag agagaa tgaaga gaagga
Topic5  cccccccc ttattatt tttttatt tttatttt ttttattt tattattt atttttat tacggtat attatttt taattatt
Topic6  tttttttt tttttaa aaaaaaaa tttaaaaa aattaatt ttaatttt atcaattc gattcttc tgttgttt attaattg
Topic7  gagaaga gaagaag agagaga agaagaa aagaaga gagagag agagaag gagagaa gaagaga aaaaaaa
Topic8  ttttttt ttatttt tattttt attattt ttttatt tttttat atttttt tttattt attttta ttgtttt
Topic9  gagagaga gaggaaga tcaatcga tattgatt gaagagag ttgattga gtattgat attcattt aaaaaaat aattgaga
Topic10 tacggta gtacggt tttcagg ttttctt aatcgat tttttct tttcttc gctcatc ttgagaa ttctatt
```

```
Topic1  0.13064713064713065
Topic2  0.11623931623931624
Topic3  0.1111111111111111
Topic4  0.10842490842490843
Topic5  0.10134310134310134
Topic6  0.0989010989010989
Topic7  0.09645909645909646
Topic8  0.08595848595848596
Topic9  0.08522588522588523
Topic10  0.06568986568986569
```

Figure 4.4: *Illustration of the process of representing a document with LDAW features. A document is represented by the count of occurrences of each word (k-mer) in LDA model*

the number of features is equal to the number of topics, which represent the LDA model of the document collection. For example, if the LDA model has 10 topics then the number of features would also be 10, as shown in the example in Figure 4.5.

The number of LDAD features only depends on the number of topics and not on the number of words. Figure 4.4 illustrates this approach.

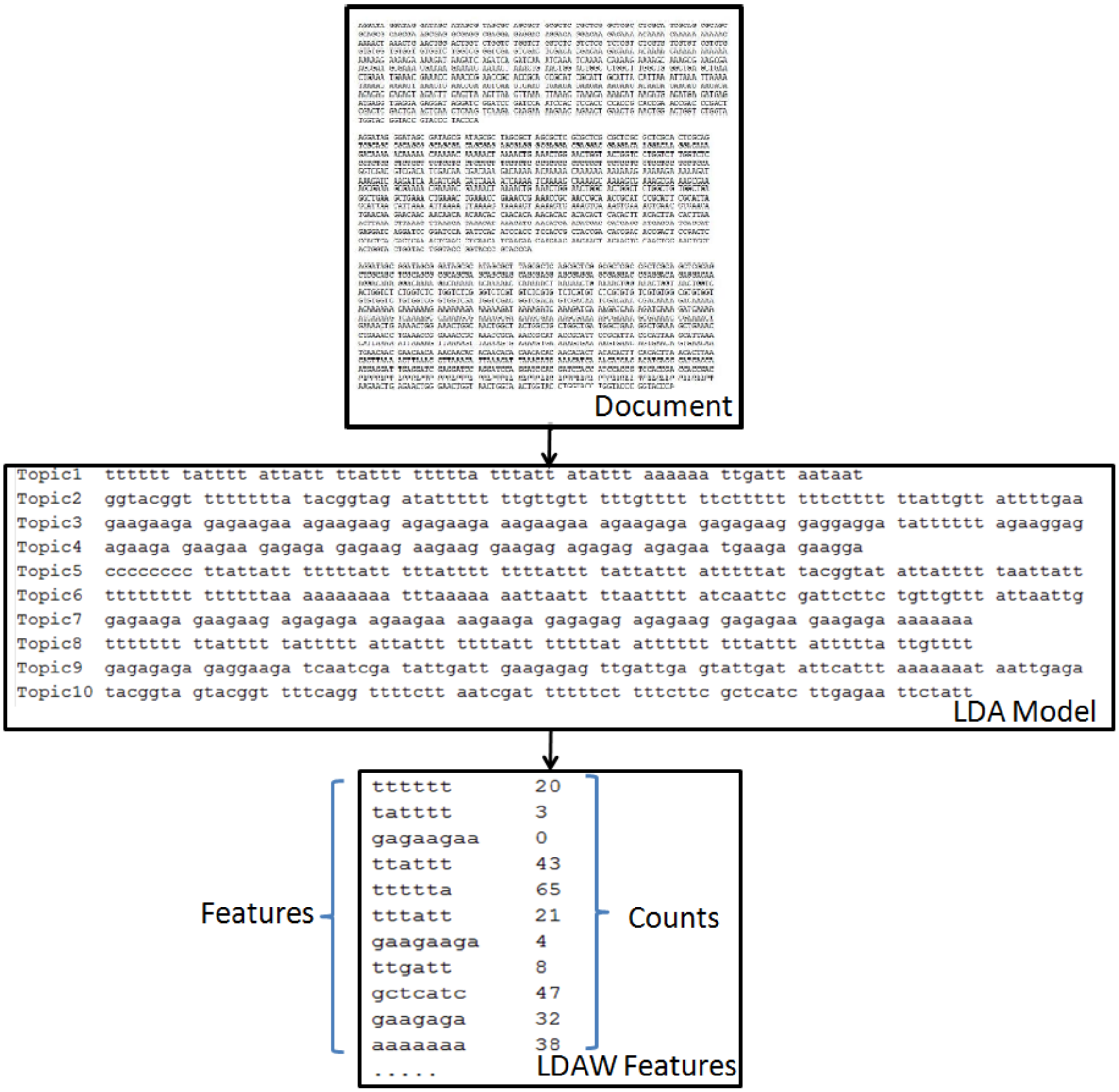

Document
Topic1 tttttt tatttt attatt ttattt ttttta tttatt atattt aaaaaa ttgatt aataat
Topic2 ggtacggt tttttta tacggtag atattttt ttgttgtt tttgtttt ttcttttt tttctttt ttattgtt attttgaa
Topic3 gaagaaga gagaagaa agaagaag agagaaga aagaagaa agaagaga gagagaag gaggagga tatttttt agaaggag
Topic4 agaaga gaagaa gagaga gagaag aagaag gaagag agagag agagaa tgaaga gaagga
Topic5 cccccccc ttattatt ttttatt tttatttt ttttattt tattattt atttttat tacggtat attatttt taattatt
Topic6 tttttttt tttttaa aaaaaaaa tttaaaaa aattaatt ttaatttt atcaattc gattcttc tgttgttt attaattg
Topic7 gagaaga gaagaag agagaga agaagaa aagaaga gagagag agagaag gagagaa gaagaga aaaaaaa
Topic8 ttttttt ttatttt tattttt attattt ttttatt ttttat atttttt tttattt atttta ttgtttt
Topic9 gagagaga gaggaaga tcaatcga tattgatt gaagagag ttgattga gtattgat attcattt aaaaaaat aattgaga
Topic10 tacggta gtacggt tttcagg ttttctt aatcgat tttttct tttcttc gctcatc ttgagaa ttctatt
LDA Model
Features
tttttt 20
tatttt 3
gagaagaa 0
ttattt 43
ttttta 65
tttatt 21
gaagaaga 4
ttgatt 8
gctcatc 47
gaagaga 32
aaaaaaa 38
.....
Counts
LDAW Features

Figure 4.5: *Illustration of the process of representing a document with LDAD features. A document is represented by the topic distribution of each topic in the LDA model.*

4.2.3 Using LDA Features for Classification

Once we obtain the features using either LDAW or LDAD, we represent both our training and test instances using these features. We then use a classifier to built a model from the training instance. We used logistic regression and Support Vector Machine (SVM) classifiers to learn models during training phase. Finally, the use the model learned during training phase to determine if a given instance belongs to positive class or negative class. An important point to note is that the sequence collection from which we obtained LDA model consisted of only the training instances and not the test instances.

4.3 Domain Adaptation Algorithms

In the second part of this work, we evaluate the effect of using LDA features in a domain adaptation framework. For this purpose we use different domain adaptation methods, similar to the methods used in Schweikert et al. [2009]. Before going into the details of these approaches let us first define the source domain and the target domain with respect to domain adaptation. A classifier trained using a traditional supervised learning algorithm can make predictions only about the unlabeled instances from the domain to which the training instances belong. However, domain adaptation algorithms make use of data from a different but related domain called as *source domain* to help make predictions about another domain called *target domain.* Basically, a source domain has some informative knowledge, which can be used to improve the classification accuracy of a target domain. Mostly, transfer learning is used when the number of labeled instances is very small in the target domain and we need to make use of labeled instances from the source domain to improve the performance of the classifier. Given this description of the source domain and the target domain, let us now describe each of the domain adaptation algorithms, and the baselines we have used.

4.3.1 Target Only (DA_t) and Source Only (DA_s) Baselines

As baseline algorithms, we used two methods named as target only (DA_t) and source only (DA_s) (see (A) and (B) of Figure 4.6). As the names suggest, in the target only method, we use labeled data from target domain only to train a classifier, and in the source only method we use labeled data from source domain only to train the classifier. An important point to note here is that the setting used for domain adaptation DA_t is the same as the one corresponding to the supervised learning framework. In other words, all the results obtained for supervised learning are the same as those obtained for DA_t.

4.3.2 Weighted Combination (DA_{xs+yt})

In the weighted combination approach the classifier is trained on the combination of source and target data (see (C), (D) and (E) of Figure 4.6). The weight of source data was kept double of target data in one set of experiments (DA_{2s+t}). In the second set of experiments the weight of target data was kept double of the weight of target data (DA_{s+2t}). Finally, in the third set of experiments the weight of target and source data was kept equal (DA_{s+t}).

An important point to note is that the sequence collection from which we obtained the LDA model in the domain adaptation setting consisted of the training instances from both source and target training instances. Unlike, the supervised learning framework where we used training instances only from the target domain.

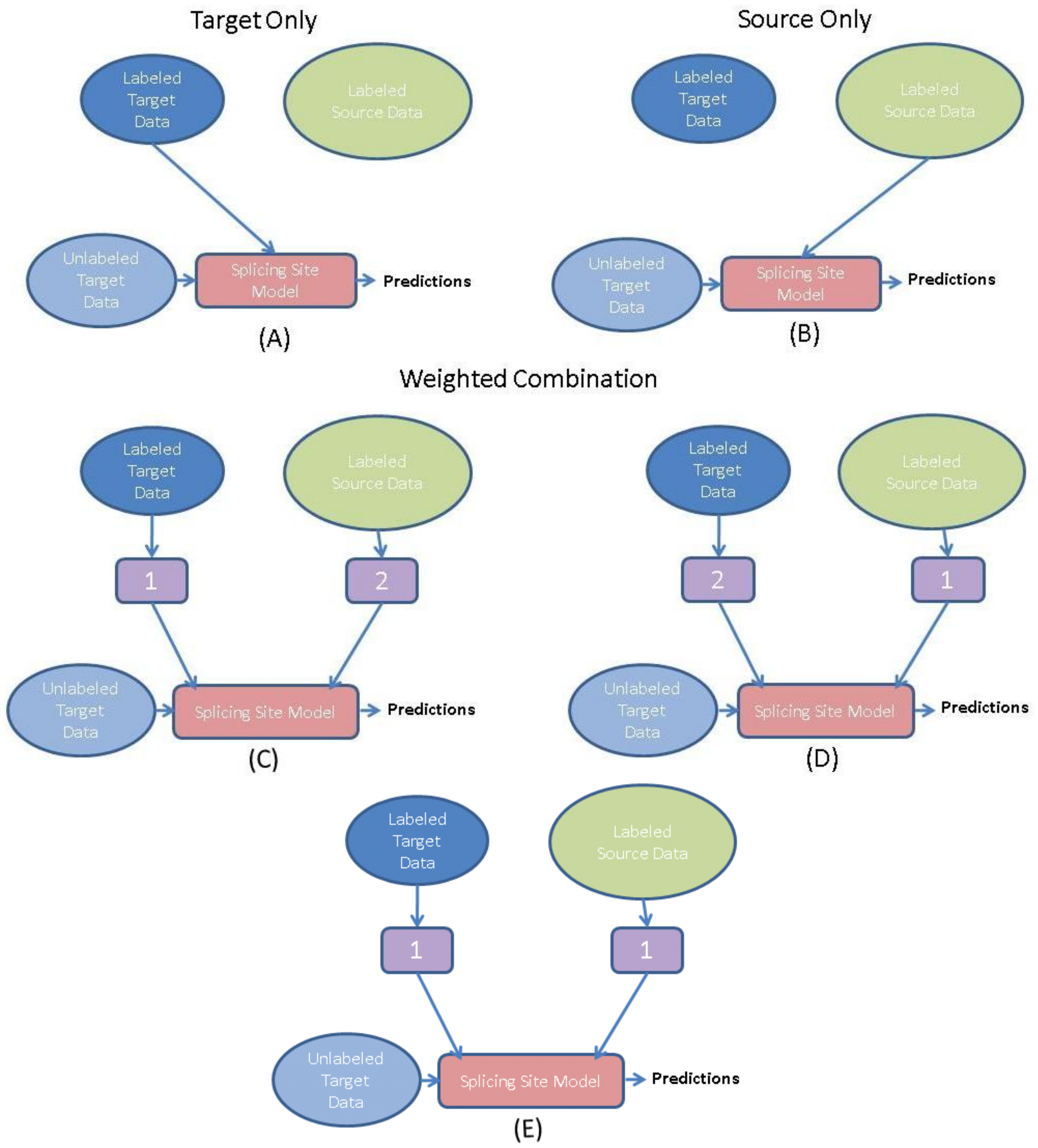


Figure 4.6: *Illustration of Target Only (DA_t), Source Only (DA_s) and Weighted Combination (DA_{xs+yt}) approaches.*

Chapter 5

Experimental Setup

This Chapter describes the details of the experiments conducted to find the effectiveness of LDA as a dimensionality reduction tool for biological sequences in a supervised learning framework and domain adaptation framework.

In supervised learning framework, we conducted experiments with the aim to answer the following questions:

- How effective are the LDAW features for classification of DNA sequences?
- How effective are the LDAD features for classification of DNA sequences?
- How does the performance vary with the number of LDA topics used?
- How does the performance vary with the size of k-mers?
- How does the performance vary with the amount of labeled data in the target domain?
- What is the effect of using different classifiers?

In domain adaptation framework, we conducted experiments with the aim to answer the following questions:

- How effective are the domain adaptation approaches with increasing evolutionary distances between species?

- What is the effect of using a weighted combination of source and target data as training data?

We will begin by explaining experimental setup for supervised learning framework and then we will discuss the experimental setup for domain adaptation framework.

5.1 Supervised Learning Framework

We begin this section by describing the dataset that we used and how we split the dataset for training and testing. We then discuss about the set of experiments which we performed.

5.1.1 Dataset Description

We consider the task of identifying acceptor and donor splice sites within a large set of potential splice sites on the basis of the sequence surrounding the potential site. We obtained the dataset for this problem from Schweikert et al. [2009]. This dataset has larger amounts of labeled samples and unlabeled samples for *C. remanei, P. pacificus, D. melanogaster* and *A. thaliana*, which we used to study our approaches in supervised learning framework.

5.1.2 Dataset Splits

The dataset splits which we used for our experiments are shown in Figure 5.1. In the training phase, we used the datasets of sizes 1,000, 2,500, 6,500, 16,000, 25,000, 40,000 and 100,000. These datasets were formed by randomly selecting samples from the labeled data for each organism. For all sample sets, the positive to negative ratio was 1/100. For cross-validation purpose, each target set was divided in three parts and all experiments were performed three times by considering two-third of the set for training. For our experiments, we balanced the training dataset such that we were left with equal number of positive and negative samples. If the training data is skewed towards a class, called majority class, then the classifier tends to predict always the majority class (even though they belong to minority class). To avoid this behavior we balance the training data by randomly removing samples from majority class. A set of 60,000 samples was kept aside for each organism, which was used for testing purpose. Each of the three classifiers built in training phase were tested with one-third of the testing dataset of size 20,000. We averaged the results obtained from all three sets of experiments.

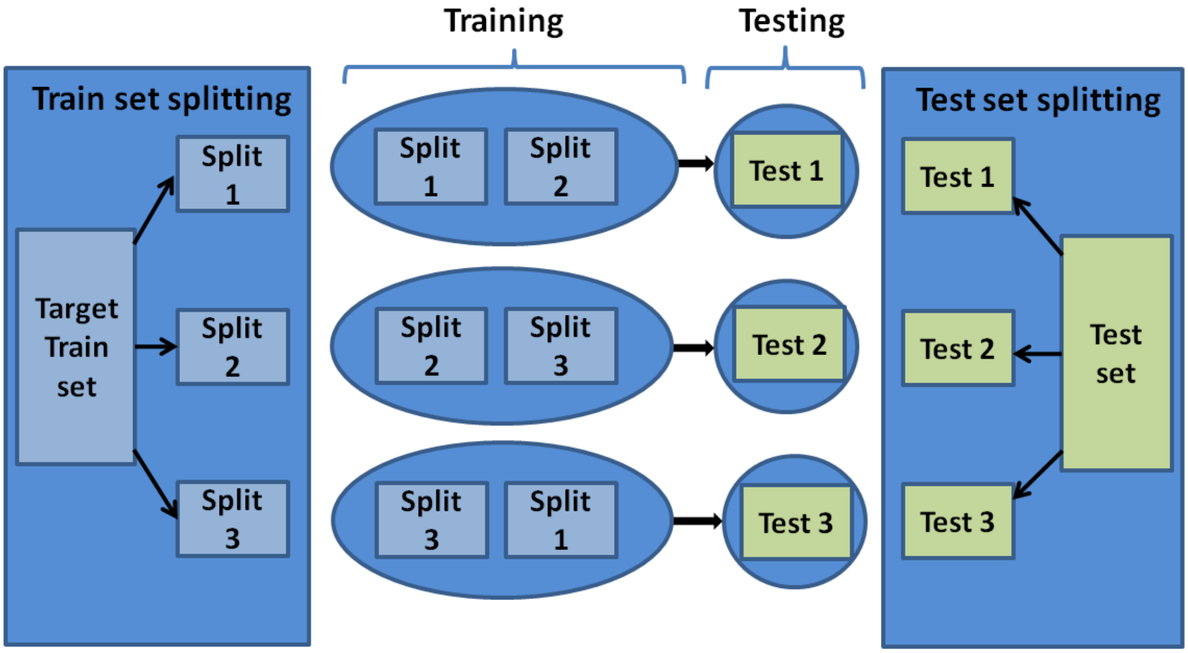


Figure 5.1: *The train set is split in three parts. Two-third of the train set is used for training. Similarly, test set is split in three parts and each part is used in each of the three experiments.*

5.1.3 Experiments

We performed experiments using both LDAW and LDAD approaches with all the four species, *C. remanei, P. pacificus, D. melanogaster* and *A. thaliana* to study the effectiveness of these approaches for classification of DNA sequences. Below are some other details about the experiments which we performed in supervised learning framework:

- To evaluate the effectiveness of LDAW and LDAD approaches we compared their results with a traditional dimensionality reduction approach based on the information gain criterion. This dimensionality reduction approach is called Mutual information (MI) method (for details refer to [McCallum and Nigam, 1998]). We use Weka's [1] implementation for information gain (InfoGainAttributeEval along with Ranker's search algorithm) to rank a set of features in the decreasing order of their mutual information with the class variable [Hall et al., 2009]. We performed four sets of experiments by selecting top 100, 500, 1000 and 2000 features from this ordered list of features.

- To study the performance variation with the number of LDA topics, we used 10, 50, 100 and 200 topics. The number of words in each topic were kept constant as 10. Therefore, the number of LDAW features obtained were 100, 500, 1000 and 2000 and the number of LDAD features obtained were 10, 50, 100 and 200, respectively.
- To study the performance variation with the size of k-mers we performed experiments using only 6-mers and only 8-mers.
- To study the effect of using different classifiers, we performed experiments with both Logistic Regression classifier and Support Vector Machine classifier.

5.2 Domain Adaptation Framework

We begin this section by describing the dataset that we used and how we split the dataset for training and testing. We then discuss about the set of experiments which we performed.

5.2.1 Dataset Description

We consider the same task of identifying acceptor and donor splice sites on the basis of the sequence around the potential site, as we considered in supervised learning framework. We obtained the dataset for this problem from Schweikert et al. [2009]. This dataset is specifically designed to be used in a domain adaptation setting. *C. elegans* is a very well

[1] http://weka.wikispaces.com/

studied model organism, which makes it appropriate as a source domain. The dataset has a large amount of splicing site labeled data for *C. elegans*. It also has larger labeled samples and unlabeled samples for *C. remanei, P. pacificus, D. melanogaster* and *A. thaliana*, which will be used as target domains. Although, domain adaptation is used when the target domain does not have enough labeled data, however, in our experiments we also studied the effect of increasing labeled data in target domain. *C .remanei* is the closest to *C. elegans* amongst all other organisms and they diverged around 100 million years ago [Stein et al., 2003]. More distantly related, *P. pacificus* diverged from *C. elegans* more than 200 million years ago [Pires-Dasilva and Sommer, 2004]. *D. melanogaster* diverged from *C.elegans* around 990 million years ago [Abel et al., 2003]. Lastly, *A. thaliana* is a plant and it separated from other organisms more than 1,600 million years ago (see Figure 5.2). As the target domain organisms have varying evolutionary distances from the source domain, therefore, this dataset will be suitable to study the effectiveness of the domain adaptation approach for increasing evolutionary distances among the species.

5.2.2 Dataset Splits

We used the same dataset splits as used by Schweikert et al. [2009]. In the training phase, we used the target datasets of sizes 1,000, 2,500, 6,500, 16,000, 25,000, 40,000 and 100,00 and a source dataset of size 25,000. These datasets were formed by randomly selecting samples from the labeled data for each organism. For all sample sets, the positive to negative ratio was always kept as 1/100. We balance the training data by randomly removing samples from majority class. A set of 60,000 samples was kept aside for each target organism, which was used for testing purpose. Each of the three classifiers built in training phase were tested with one-third of the testing dataset of size 20,000. We averaged the results obtained from all three sets of experiments. (see Figure 5.3).

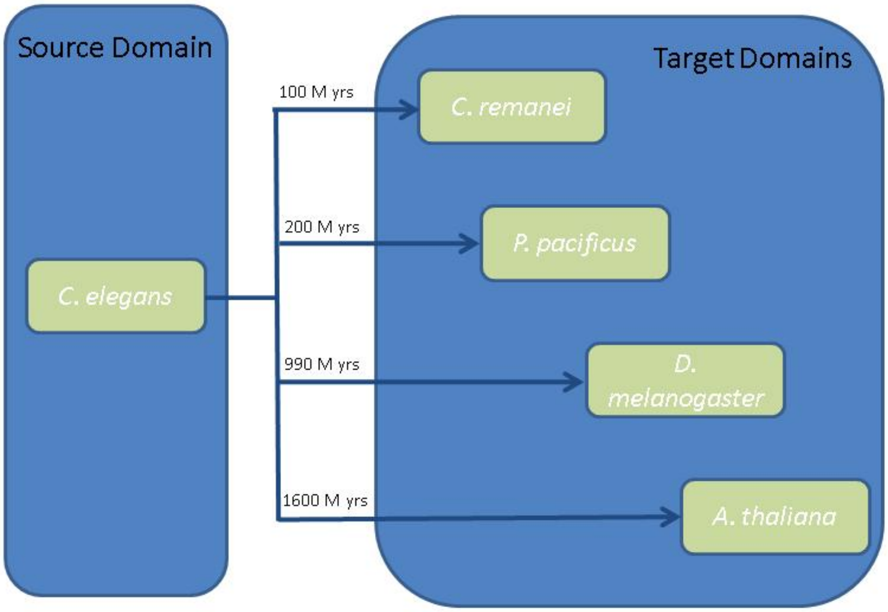


Figure 5.2: *Information transfer from source to target domains. As source organism, we used the well-annotated model organism* C. elegans*. As target domains, we used four organisms with varying evolutionary distances to* C. elegans*.*

5.2.3 Experiments

We used only LDAW approach to study the effect of using different Domain Adaptation algorithms. We performed experiments for both the baselines (DA_t and DA_s) algorithms four target species. To study the variation of the performance with changing the weights of labeled data in source and target domain, we performed experiments for DA_{s+t}, DA_{2s+t} and DA_{s+2t}. The weighted combination experiments were also performed for all the four target species. We used Logistic Regression classifier for all experiment performed in domain adaptation framework.

The results obtained from all the experiments discussed in this chapter are presented in Chapter 6 and conclusions derived from them are discussed in Chapter 7.

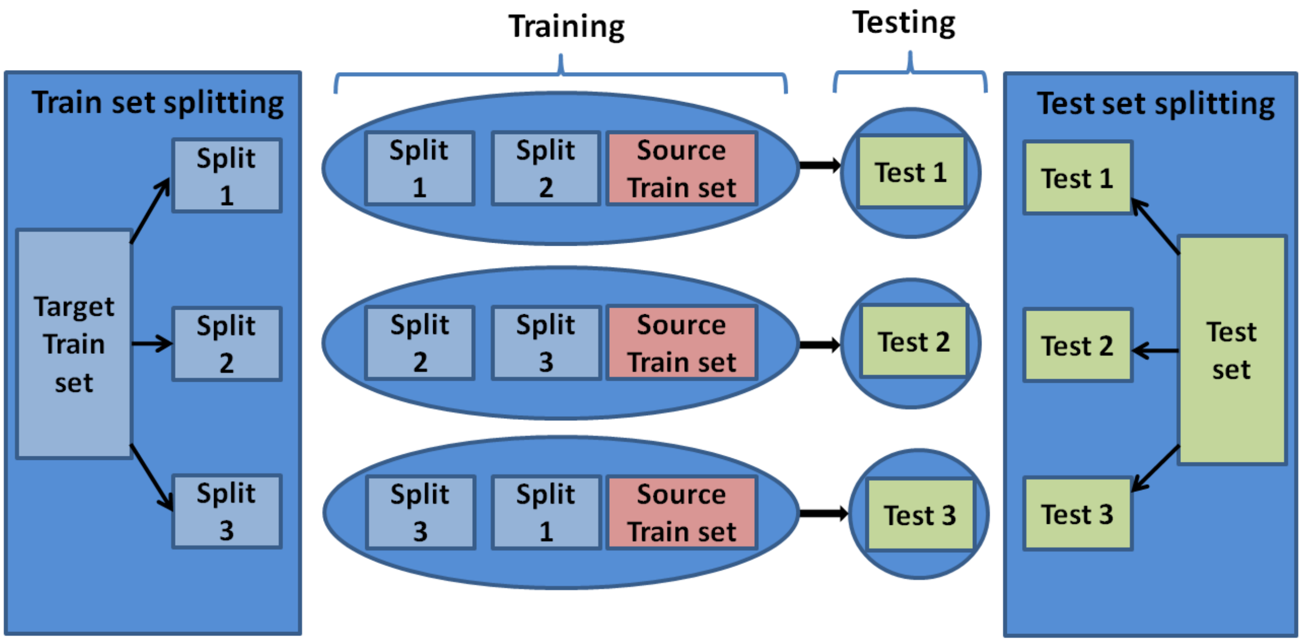


Figure 5.3: *The target train set is split in three parts. Two-third of the labeled samples from target domain is combined with the labeled sampled from source domain and is used for training. Similarly, target test set is split in three parts and each part is used in each of the three experiments.*

Chapter 6

Results

In this chapter, we discuss the results obtained by running the experiments described in Chapter 5. We begin by discussing the results obtained for the experiments performed in supervised learning framework in Section 6.1. We then discuss the results obtained for the experiments performed in domain adaptation framework in Section 6.2.

6.1 Supervised Learning Framework Results

The organization of this section is as follows: in Section 6.1.1 we report the results of the experiments performed to study the effectiveness of LDAW and LDAD approach as compared to MI features; in Section 6.1.2 we examine the results of using different classifiers; in Section 6.1.3 we discuss the effect of varying different parameters for LDAW method i.e. number of topics and size of k-mers, and in Section 6.1.4, we study the effect of increasing labeled data in training dataset.

6.1.1 Study of Effectiveness of LDA Topic Modeling Features

In this section, we investigate the effectiveness of using LDA topic modeling for dimensionality reduction. This section is further divided into two sub-sections. In the first sub-section, we compare LDAD and LDAW methods to see which one gives better performance. In the second sub-section, we compare LDAD results with MI results, which is a traditional dimensionality reduction method.

LDAW vs. LDAD

We first compare the results obtained from LDAW and LDAD approaches. For this experiment, we only use the supervised learning framework (DA_t) approach with logistic regression classifier. We perform experiments for different training data sizes and for different number of LDA topics. The results obtained with LDAW approach are shown in Table 6.1, while the LDAD approach results are shown in Table 6.2. These tables show that auROC values obtained from LDAW methods are greater than that of LDAD approach.

C. remanei							
	1000	2500	6500	16000	25000	40000	100000
10	0.615	0.574	0.549	0.635	0.682	0.699	**0.718**
50	0.586	0.56	0.613	0.536	0.546	0.544	**0.621**
100	0.591	0.592	0.585	**0.638**	0.528	0.559	0.568
200	0.634	0.62	**0.658**	0.606	0.563	0.56	0.645
P. pacificus							
	1000	2500	6500	16000	25000	40000	100000
10	0.486	0.494	0.501	0.489	0.491	**0.492**	0.488
50	0.49	0.49	0.507	0.49	0.487	**0.491**	0.487
100	0.493	0.592	0.585	**0.638**	0.492	0.506	0.487
200	0.473	0.5	0.477	**0.515**	0.501	0.502	0.501
D. melanogaster							
	1000	2500	6500	16000	25000	40000	100000
10	0.512	0.552	0.579	0.544	0.524	0.577	**0.664**
50	0.553	0.576	0.598	0.603	0.602	0.626	**0.643**
100	0.594	0.523	0.613	**0.655**	0.606	0.549	0.521
200	0.621	0.587	0.636	**0.646**	0.575	0.629	0.512
A. thaliana							
	1000	2500	6500	16000	25000	40000	100000
10	0.525	0.525	0.523	0.579	0.651	0.64	**0.667**
50	0.554	0.548	0.585	0.556	0.543	0.549	**0.587**
100	0.561	0.592	**0.593**	0.546	0.571	0.56	0.524
200	0.572	0.593	**0.603**	0.597	0.592	0.553	0.501

Table 6.1: *auROC values obtained from LDAW method in supervised learning framework (DA_t approach) with logistic regression classifier. The target data set sizes used are 1000, 2500, 6500, 16000, 25000, 40000 and 100000, respectively (column names). The number of topic used are 10, 50, 100 and 200, respectively (row names). The best value obtain for a given number of topics with increasing data set sizes is highlighted in each row.*

C. remanei							
	1000	2500	6500	16000	25000	40000	100000
10	0.458	0.486	0.524	**0.533**	0.512	0.481	0.489
50	0.455	0.505	0.52	**0.525**	0.506	0.476	0.469
100	0.438	0.491	0.525	**0.526**	0.508	0.483	0.496
200	0.489	0.521	**0.533**	0.531	0.522	0.509	0.503
P. pacificus							
	1000	2500	6500	16000	25000	40000	100000
10	0.472	0.472	0.502	0.498	**0.507**	0.493	0.481
50	0.481	0.482	**0.513**	0.509	0.495	0.485	0.473
100	0.47	0.484	0.482	**0.491**	0.482	0.479	0.467
200	0.494	0.517	0.524	**0.523**	0.523	0.518	0.518
D. melanogaster							
	1000	2500	6500	16000	25000	40000	100000
10	0.453	0.487	0.523	**0.551**	0.502	0.473	0.462
50	0.497	0.518	0.515	0.521	**0.533**	0.48	0.46
100	0.514	0.527	**0.53**	0.526	0.503	0.492	0.459
200	0.499	0.506	0.503	0.511	**0.537**	0.516	0.511
A. thaliana							
	1000	2500	6500	16000	25000	40000	100000
10	0.495	0.506	0.505	**0.516**	0.514	0.509	0.492
50	0.454	0.499	0.51	0.514	**0.517**	0.484	0.484
100	0.476	0.482	0.506	0.51	**0.531**	0.487	0.493
200	0.43	0.498	0.52	**0.528**	0.528	0.505	0.509

Table 6.2: *auROC values obtained from LDAD method in supervised learning framework (DA_t approach) with logistic regression classifier. The target data set sizes used are 1000, 2500, 6500, 16000, 25000, 40000 and 100000, respectively (column names). The number of topic used are 10, 50, 100 and 200, respectively (row names). The best value obtain for a given number of topics with increasing data set sizes is highlighted.*

Figure 6.1 shows a graph obtained by plotting auROC values against training data size for both LDAW (red) and LDAD (blue) approaches. It is clear from this graph that LDAW gives better performance as compared to LDAD for various training data sizes. Furthermore, even for the same number of LDA topics LDAW performs better that LDAD. An explanation for this behavior could be that for the same number for LDA topics the number of features for LDAW is ten times larger that the number of features for LDAD. For example, for 10 LDA topics the number of LDAW features is 100 as each topic has 10 words, whereas for

LDAD we just consider the topic distribution, therefore, there will be only 10 features.

Note that we have shown results only for *C. remanei*, *D. melanogaster* and *A. thaliana*. The results obtained for *P. pacificus* were very poor. All the auROC values were less that 0.5 and there was no *improvement* with training data size for any of the methods.

LDAW vs. MI

The results from the above sub-section clearly show that LDAW is more effective than the LDAD approach. In this section, we present the results of the experiments which compare LDAW with a traditional dimensionality reduction method, specifically Mutual information (MI).

Summary of all the results obtained using the MI dimensionality reduction method is shown in Table 6.3. We performed experiments using MI features for all four organisms and varied both training data set sizes and number of features. The classifier used for this study was logistic regression and the approach used was DA_t.

As discussed in the previous section, the results for *P. pacificus* are very poor using LDAW method. These results do not follow the pattern observed for the other three organisms. Thus, all the observations made in this section are based on results obtained from *C. remanei, D. melanogaster* and *A. thaliana*.

First, to study the effect of number of features on performance of LDAW method and MI method, we plotted a graph against the number of features which gave best performance for a given training data set size. Figure 6.2 shows the graphs obtained by plotting the number of features which gave best performance for difference data set sizes for *C. remanei, P. pacificus, D. melanogaster* and *A. thaliana*, respectively. It can be seen from these graphs that for MI method the best performance is obtained for 1000 features, irrespective of the size of training data set. However, in case of the LDAW method, for smaller training data sets, larger number of features give better value whereas for large training data set smaller number of features (10 topics i.e. 100 features) give better value. The best performance obtained for small number of features with large training set can be explained for biological

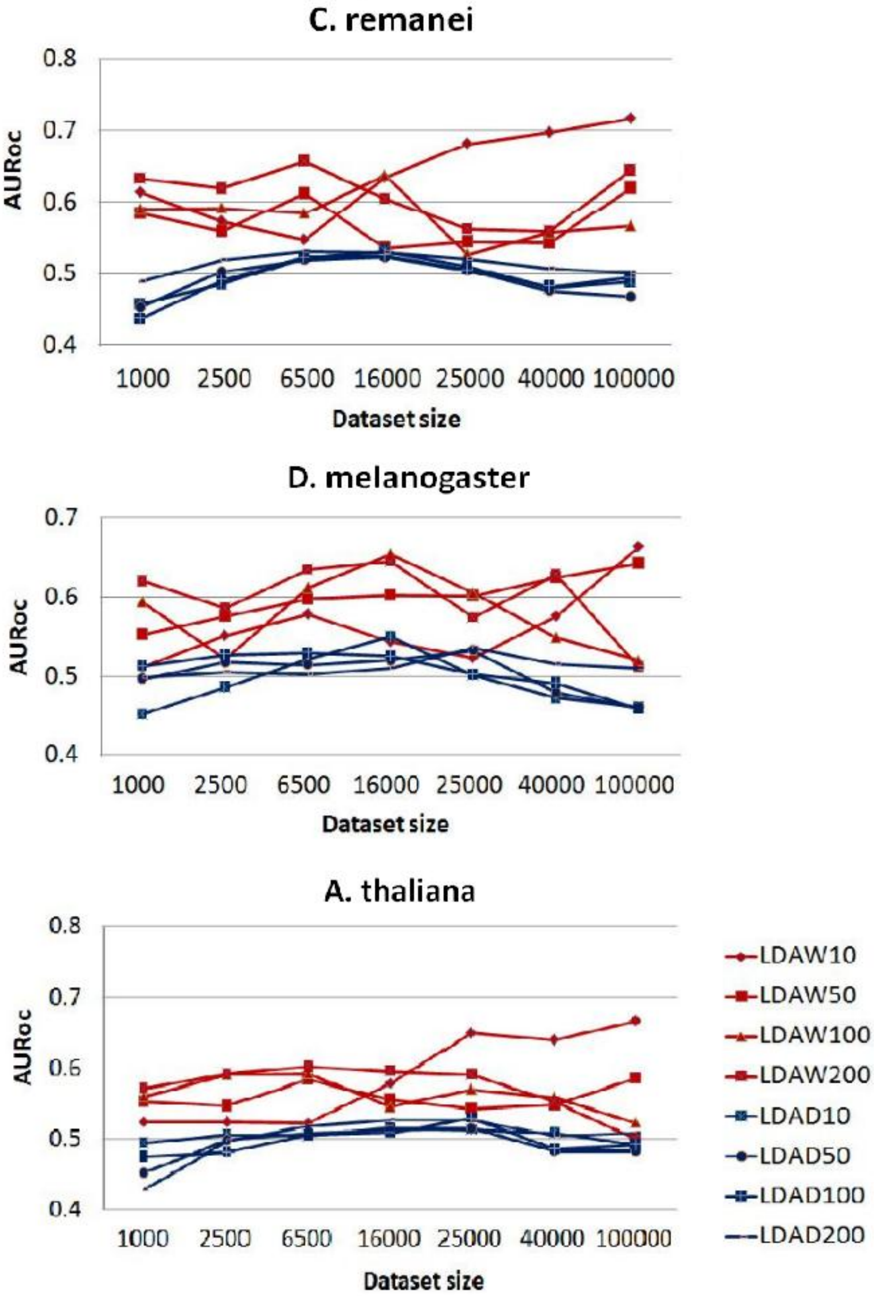


Figure 6.1: *Graphs obtained by plotting auROC values against training data size for C. remanei, D. melanogaster and A. thaliana, respectively. Each data series corresponds to a different number of LDA topics (LDAW10 corresponds to LDAW method with 10 topics).*

data, however, the best performance obtained for small number of topics with larger number of features is counter intuitive. Discussion of such behavior is presented in Section 6.1.3.

Also, from Table 6.3, it can be observed that for MI method the performance increases with the increase in number of features and then it starts decreasing after a threshold. For our experiments the best performance is obtained for 1000 features, however, for 2000 features again the performance degrades.

Second, to compare the performance of LDAW and MI method, we plotted a graph of the best auROC values obtained from LDAW method and MI method against dataset sizes. Figure 6.3 shows a graph obtained by plotting best auROC value, irrespective of the number of features against the training dataset for *C. remanei, P. pacificus, D. melanogaster* and *A. thaliana*. The most important thing to note here is that for all three organisms, *C. remanei, D. melanogaster* and *A. thaliana*, the performance of LDAW method is better than MI method for large training data sets. Otherwise the MI method gives better result than LDAW method. Thus, we can say that LDAW is able to select better features than MI method when the size of the training dataset is very large.

6.1.2 Study of Support Vector Machine vs. Logistic Regression Classifiers

In the previous subsection, we analyzed the effectiveness of using LDA features. All these experiments were performed using Logistic Regression classifier (see Section 2.2.1). In this section, we will study the effect of using a different classifier. For this comparison we used Support Vector Machine classifier (see Section 2.2.2), as a classifier and compared it to the Logistic Regression classifier.

Table 6.4 shows the results obtained for LDAW method using the Support Vector Machine (SVM) classifier (with default parameters). We performed experiments using SVM classifier for all four organisms and varied both training data set sizes and number of features. We have performed these experiments only in supervised learning framework.

To compare the performance of logistic regression classifier and SVM classifier we plot-

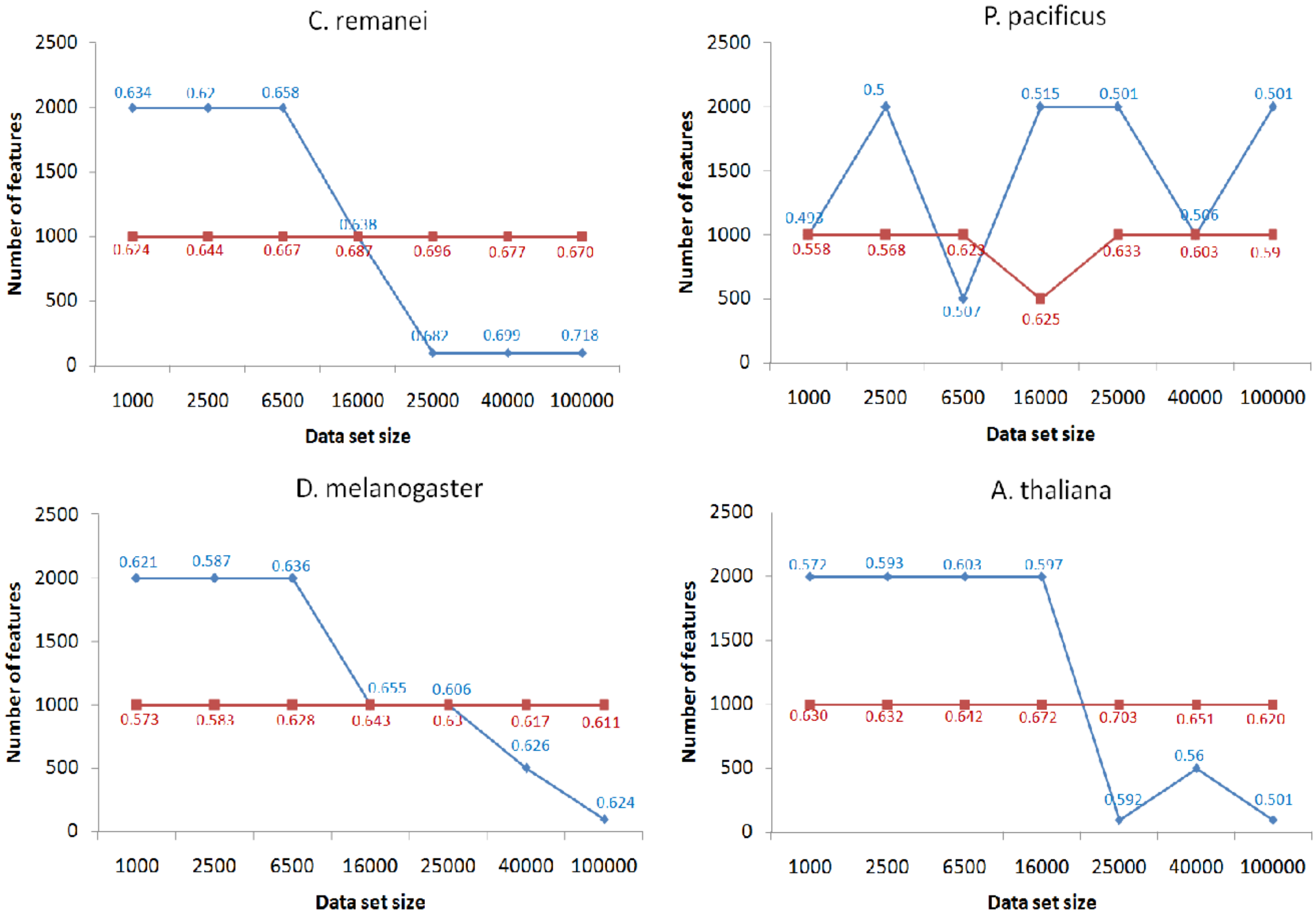


Figure 6.2: *Graphs obtained by plotting the number of features which gave the best performance against training data size. Blue data series corresponds to the LDAW method and Orange corresponds to the MI method.*

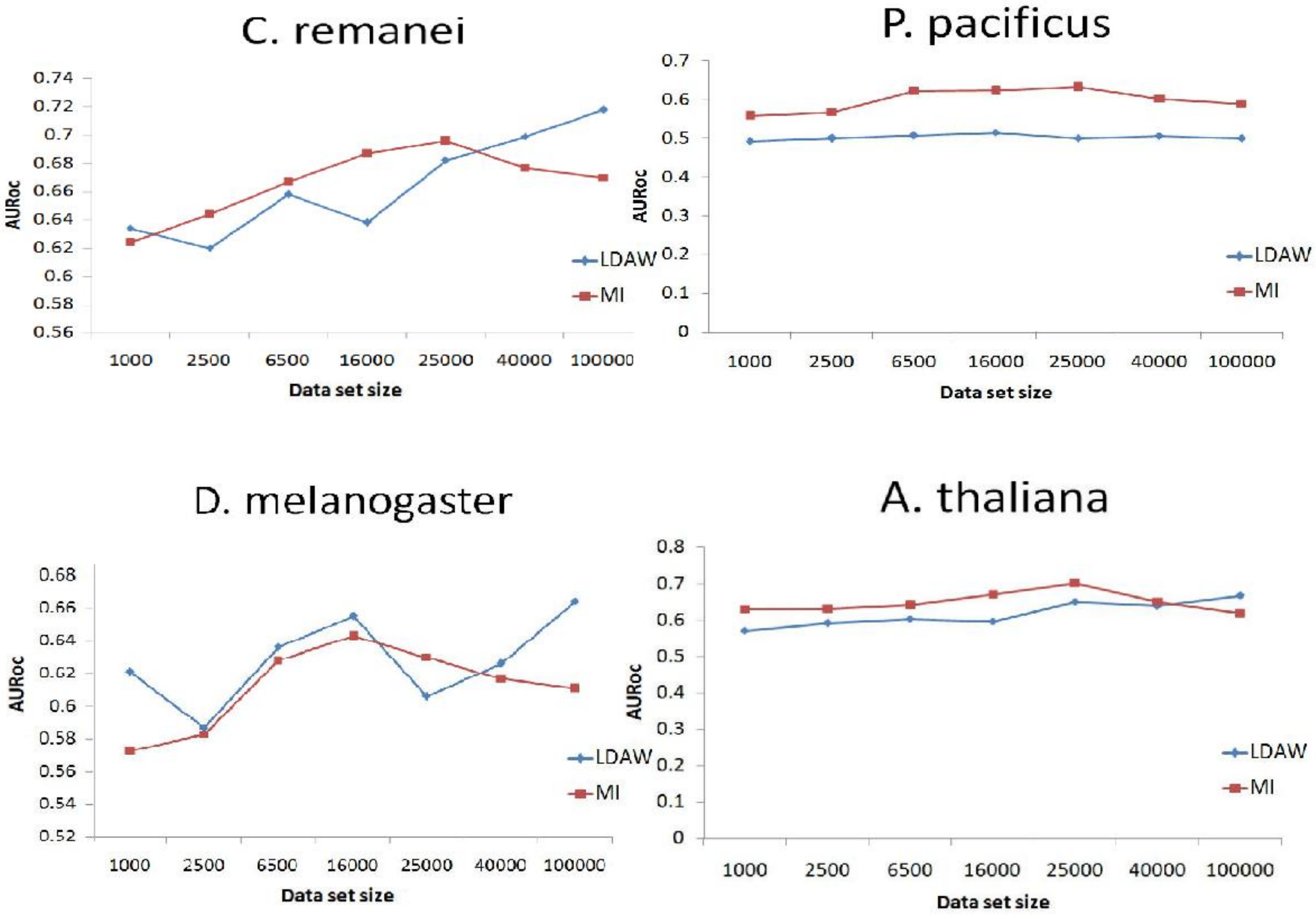


Figure 6.3: *Graphs obtained by plotting best auROC for a given training data set size against training data size. Blue data series corresponds to the LDAW method and Orange corresponds to the MI method.*

C. remanei							
	1000	2500	6500	16000	25000	40000	100000
100	0.544	0.589	0.593	0.597	0.604	0.587	0.58
500	0.605	0.625	0.634	0.638	0.628	0.632	0.606
1000	0.624	0.644	0.667	0.687	0.696	0.677	0.670
2000	0.584	0.597	0.612	0.62	0.653	0.646	0.632
P. pacificus							
	1000	2500	6500	16000	25000	40000	100000
100	0.489	0.532	0.54	0.544	0.568	0.573	0.568
500	0.538	0.547	0.60	0.625	0.628	0.583	0.571
1000	0.558	0.568	0.623	0.611	0.633	0.603	0.59
2000	0.547	0.549	0.564	0.577	0.598	0.581	0.573
D. melanogaster							
	1000	2500	6500	16000	25000	40000	100000
100	0.546	0.550	0.567	0.565	0.574	0.577	0.563
500	0.552	0.558	0.592	0.618	0.587	0.562	0.540
1000	0.573	0.583	0.628	0.643	0.63	0.617	0.611
2000	0.566	0.572	0.592	0.612	0.605	0.587	0.580
A. thaliana							
	1000	2500	6500	16000	25000	40000	100000
100	0.554	0.574	0.581	0.597	0.608	0.588	0.583
500	0.616	0.624	0.639	0.642	0.675	0.636	0.612
1000	0.630	0.632	0.642	0.672	0.703	0.651	0.620
2000	0.592	0.604	0.613	0.612	0.623	0.607	0.587

Table 6.3: *auROC values obtained from MI method in supervised learning framework (DA_t approach) with logistic regression classifier. The target data set sizes used are 1000, 2500, 6500, 16000, 25000, 40000 and 100000, respectively (column names). The number of features used are 100, 500, 1000 and 2000, respectively (row names).*

ted a bar graph showing the auROC values for increasing training data set sizes for both classifiers. Figure 6.4 shows these graphs for *C. remanei, P. pacificus, D. melanogaster* and *A. thaliana*. The red data series corresponds to results from logistic regression classifier and the blue data series corresponds to SVM classifier. Following are the observations made from this graph:

- Firstly, for *C.remanei* SVM classifier performs better than logistic regression classifier except for larger number of topics (features) i.e., for training data set sizes of 1,000, 2,500, 6,500 and 1,00,000 for 200 topics logistic regression gives better results than SVM classifier.

- Secondly, for both *A. thalians* and *D. melanogaster* the logistic regression classifier performs better than SVM classifier in most of the cases.
- Lastly, for *P. pacificus* for all the training data set sizes and number of topics logistic regression classifier performs better than SVM classifier.

From the above observations we can see that the performance of logistic regression classifier is better for majority of the organisms. Thus, we perform the rest of our experiments using the logistic regression classifier only. Note: SVM has not been tuned. It has been used with a definite parameters (meaning a linear kernel).

6.1.3 Study of Different LDA Topic Modeling Settings

In the previous section, we studied the effect of using different classifier. An important aim of this work is also to study the effect of varying the number of LDA features on classification performance. With this aim in mind, we performed our next set of experiments with different number of LDA topics and different k-mer sizes. Results from each of these experiments are presented in the subsections that follow.

The Effect of Increasing the Number of LDA Topics

We varied the number of LDA topics for each set of experiments that we performed. In this section, we studied the effect of varying LDA topics on LDAW and LDAD methods. We begin our discussion with LDAW method and then we move to LDAD method.

The results obtained from LDAW approach for in supervised learning framework using logistic regression classifier are summarized in Table 6.1. Based on this table we constructed a graph showing the number of topics which give best auROC values against the training data set sizes for *C. remanei, D. melanogaster* and *A. thaliana*. As mentioned in Section 6.1.1, the results for *P. pacificus* do not follow the patterns observed for the other three

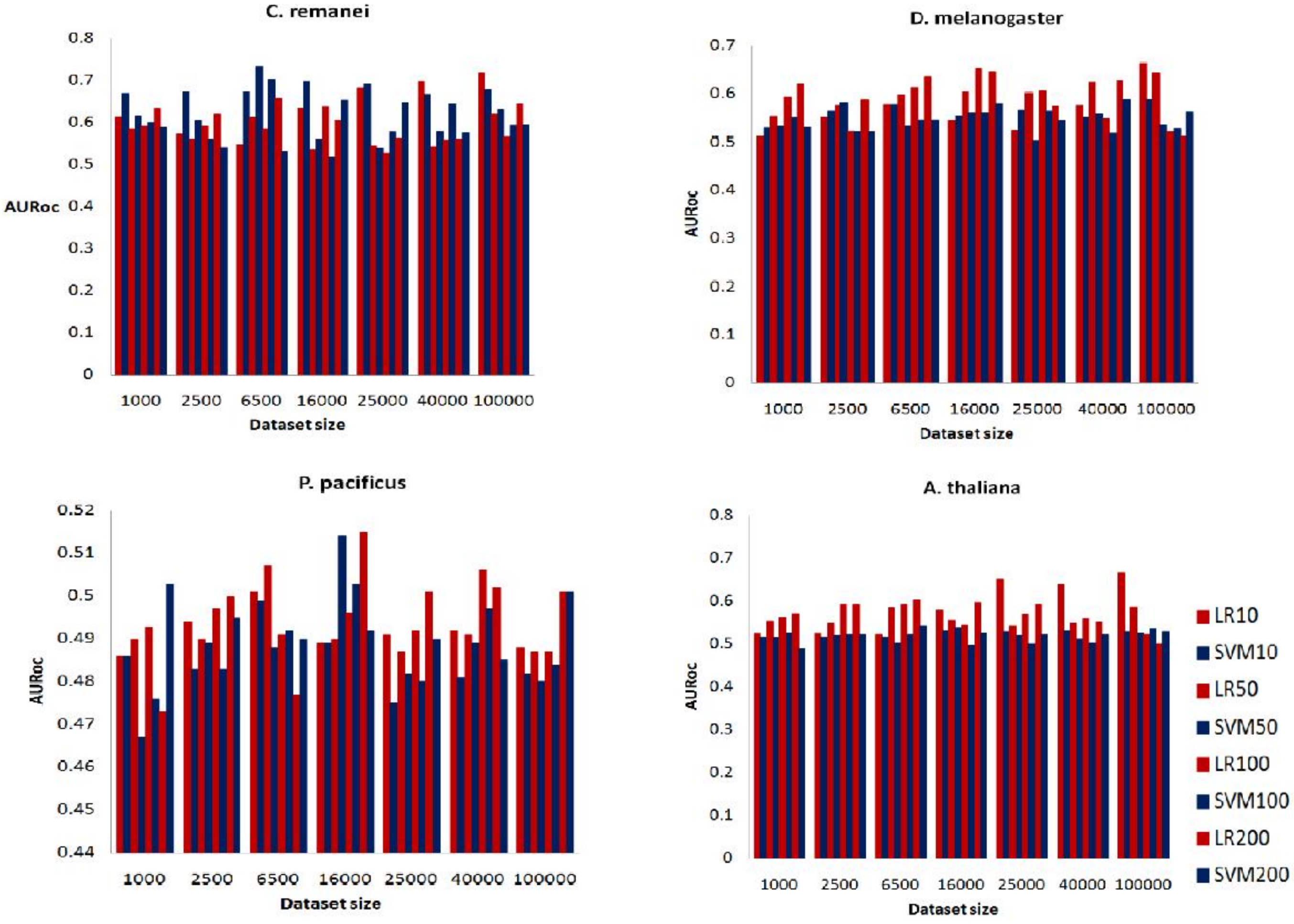


Figure 6.4: *Graphs obtained by plotting auROC values against training data size. Each data series corresponds to a different number of LDA topics (LR10 corresponds to LR classifier with 10 topics). Results for Logistic Regression classifier are shown in red and for Support Vector Machine classifier are shown in blue.*

C. remanei							
	1000	2500	6500	16000	25000	40000	100000
10	0.669	0.674	0.675	0.698	0.692	0.667	0.68
50	0.616	0.605	0.734	0.561	0.54	0.58	0.631
100	0.601	0.56	0.703	0.518	0.578	0.645	0.594
200	0.589	0.541	0.533	0.655	0.648	0.577	0.596
P. pacificus							
	1000	2500	6500	16000	25000	40000	100000
10	0.486	0.483	0.499	0.489	0.475	0.481	0.482
50	0.467	0.489	0.488	0.514	0.482	0.489	0.48
100	0.476	0.483	0.492	0.503	0.48	0.497	0.484
200	0.503	0.495	0.49	0.492	0.49	0.485	0.501
D. melanogaster							
	1000	2500	6500	16000	25000	40000	100000
10	0.53	0.566	0.579	0.555	0.569	0.552	0.588
50	0.534	0.583	0.535	0.562	0.505	0.56	0.537
100	0.551	0.521	0.546	0.562	0.566	0.517	0.572
200	0.531	0.521	0.545	0.582	0.544	0.588	0.563
A. thaliana							
	1000	2500	6500	16000	25000	40000	100000
10	0.517	0.518	0.517	0.533	0.53	0.532	0.529
50	0.516	0.52	0.505	0.538	0.522	0.512	0.526
100	0.527	0.523	0.524	0.498	0.501	0.505	0.536
200	0.49	0.524	0.542	0.527	0.523	0.524	0.529

Table 6.4: *auROC values obtained from LDAW method in supervised learning framework with SVM classifier. The target data set sizes used are 1000, 2500, 6500, 16000, 25000, 40000 and 100000, respectively (column names). The number of LDA topics used are 10, 50, 100 and 200. respectively (row names).*

organisms for LDAW method. Therefore, we did not plot the result for *P. pacificus*. This graph is shown in Figure 6.5. The following observations are made based on this graph:

- For training data set sizes of 1,000, 2,500 and 6,500 the best auROC value is obtained for 200 topics.
- For training data set sizes of 16,000 the best auROC value is obtained for 100 topics.
- For training data set sizes of 25,000, 40,000 and 100,000 the best AUROC is obtained for 50 or 10 topics.

The best auROC values for small number of features with large training set is easy to explain for biological data. Every sequence in our dataset is 141bp long. About 100 - 500 k-mers are enough to cover all the possible consensus sequences in a given sequence collection. Thus, we can classify only using small number of features even for a large datasets. However, the best performance obtained for small number of topics with larger number of features is counter intuitive and only some kind of over fitting can explain these results. We need to investigation deeper to explain these results. We can use a denser LDA topic scale, with steps of 10 topics to understand this behavior.

The results obtained from LDAD approach in supervised learning framework approach using logistic regression classifier are summarized in Table 6.2. Based on this table we constructed a graph showing the number of topics which give the best auROC value against the training data set sizes for all the organisms. This graph is shown in Figure 6.5. We can clearly see from this graph that the best performance is obtained from 200 topics for almost all sizes of training data sets and all organisms. There are only few exceptions for smaller data, set sizes where performance with less number of topics is better. Thus, we can conclude that the performance of LDAD is better for large number of topics than for small number of topics.

The Effect of Increasing the Size of k-mers

In this section, we studied the effect of increasing the size of k-mer and effect of using combination of k-mers as features. All the results we presented so far were using a combination of 6-mers, 7-mers and 8-mers features (6+7+8 mers). In order to perform this study, we conducted two sets of experiments. In the first set of experiments we used only 6-mers features and in the second set of experiments we used only 8-mers features. We used the logistic regression classifier and the DA_t approach for these experiments. We compared the results obtained from these experiments with the results in Table 6.1, where we have also used logistic regression classifier and DA_t approach but with the combination of 6-mers, 7-mers and 8-mers. Thus, for each of these experiments the classifier, approach, test data,

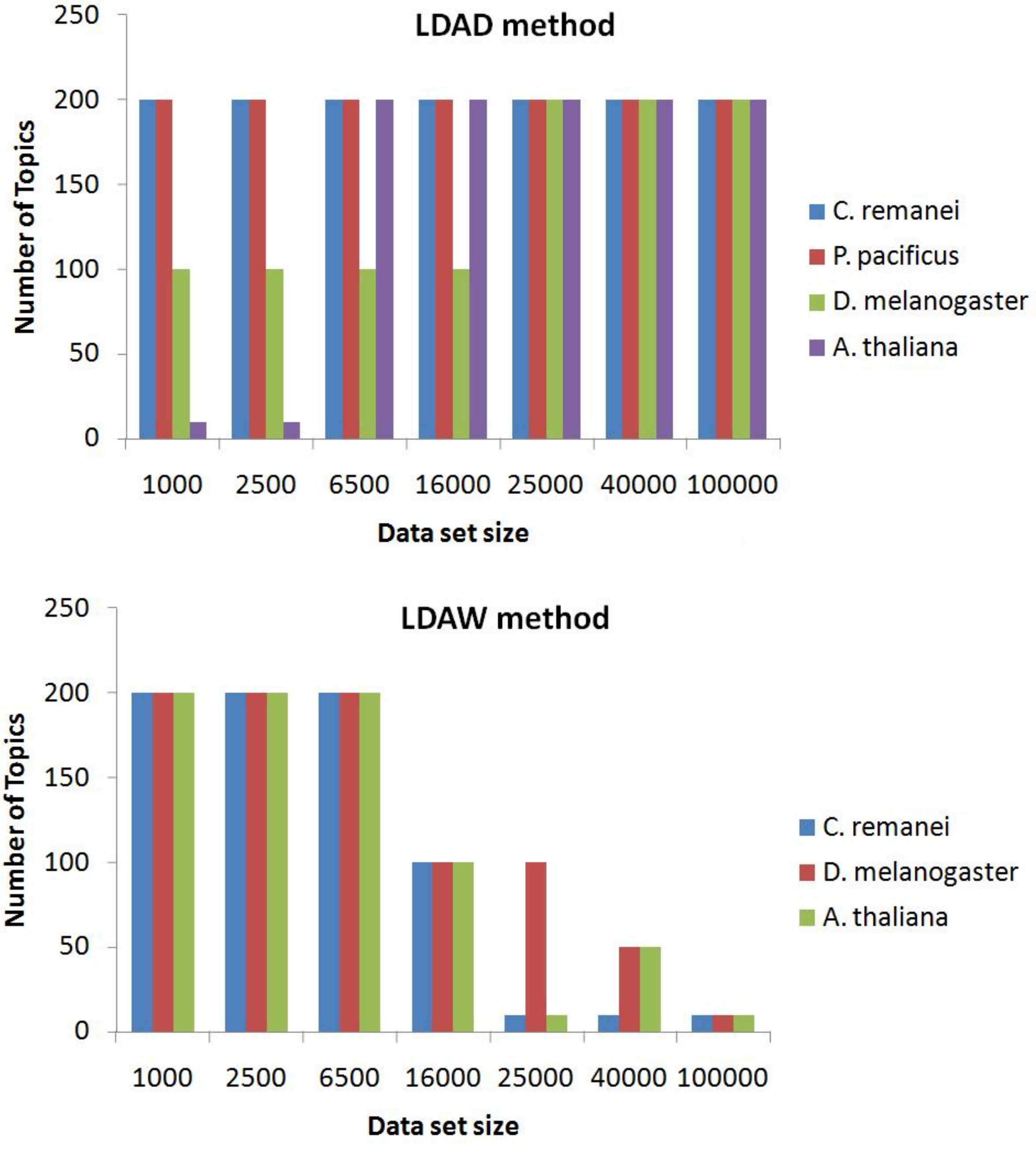


Figure 6.5: *Graph obtained by plotting the number of topics that give the best auROC against the training data size for LDAW method and LDAD method. Each data bar color corresponds to a different organism.*

train data and number of LDA topics used are the same, only the type of features are changed. The results which we obtained from 6-mers only features and 8-mers only features are summarized in Tables 6.5 and 6.6, respectively. We did not presented the results for *P.pacificus* for the reasons discussed in the above sections.

In order to further analyze these results we plotted graphs for auROC values obtained for 6-mers only, 8-mers only and combination of k-mers against the increasing size of training data sets. We will study the graphs obtained from *C. remanei*, *D. melanogaster* and *A.thaliana*. The graphs obtained for *C. remanei* for topic count 10, 50, 100 and 200 are shown in Figure 6.6. We plotted similar graphs for *D. melanogaster*, which are shown in Figure 6.7, each graph corresponding to topic count of 10, 50, 100 and 200. Finally, the same type of graphs were also plotted for *A. thaliana*. These graphs are shown in Figure 6.8, each corresponding to topic count of 10, 50, 100 and 200. Following are the observations made based on these graphs:

- The maroon data series representing 6-mers only features is lower than the other two data series in nearly all graphs. This shows that performance of 6-mers only is worst as compared to both 6+7+8 mers features and 8-mers only features. Since, 6-mers only features gave poor results as compared to 8-mers only features we can conclude that the performance of LDAW method would improve with the increase in the size of k-mers used for obtaining LDAW features, although the improvement is not very drastic. This can be explained by the fact that the information captured by a larger k-mer is more as compared to the information captured by a smaller k-mer. An important point to note here is that, after a threshold the performance will degrade with the increase in the k-mer size.

- The blue data series representing 6+7+8 mers features is above the green data series representing 8-mers features for smaller data sets. However, for larger data sets green data series is above the blue data series. Therefore, we can say that for smaller training data sets 6+7+8 mers performs better, whereas for larger training data sets 8-mers only features perform better. Better performance of 8-mers only features as compared to 6+7+8 mers can be explained by the fact that number of features in 6+7+8 mers combination are much more than 8-mers only features. Large number of features lead to overfitting, thereby decreasing the performance of 6+7+8 mers features.

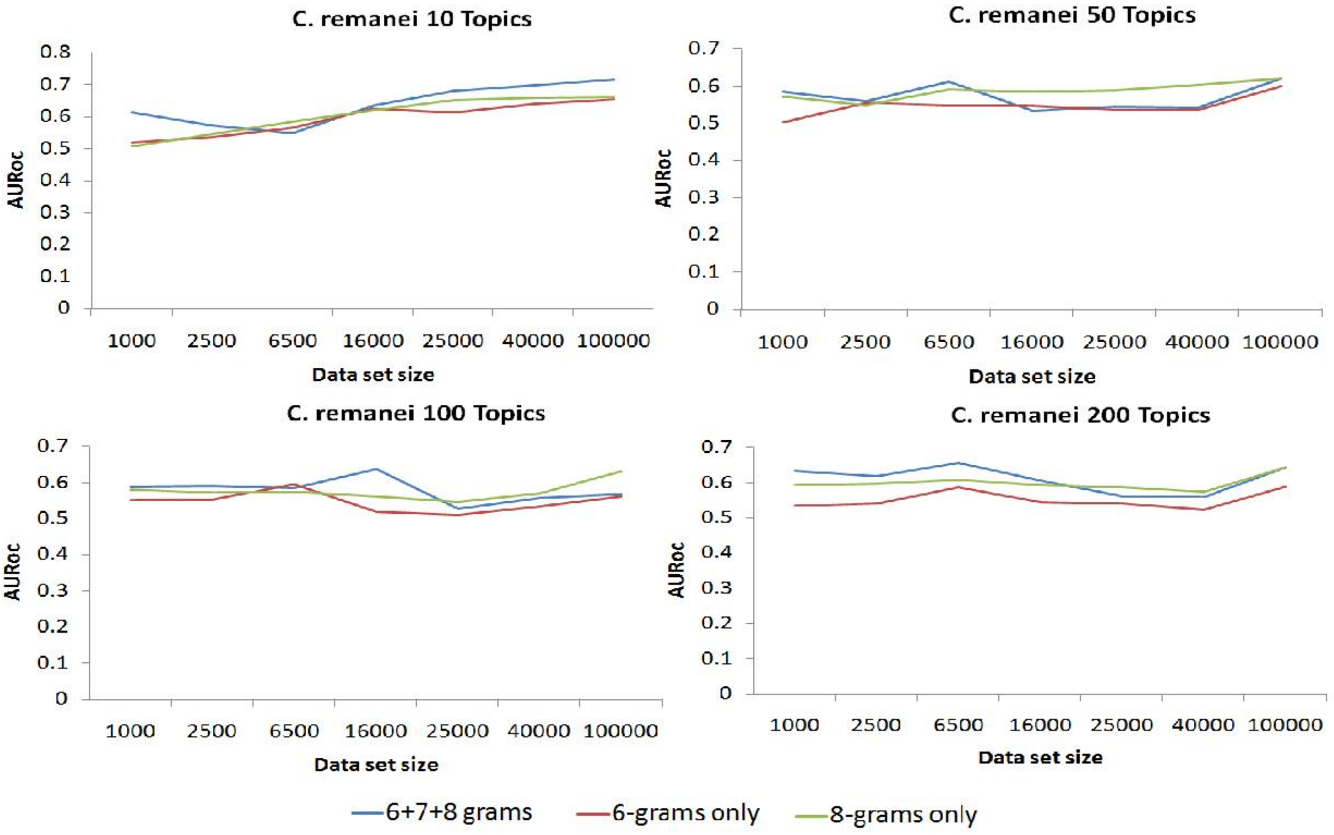


Figure 6.6: *Graphs obtained by plotting the auROC against the training data size for LDAW method with 10, 50, 100 and 200 LDA topics in supervised learning framework for C.remanei. Data series blue corresponds to features obtained from combination of 6-mers, 7-mers and 8-mers, data series green corresponds to features obtained from 8-mers only and data series maroon corresponds to features obtained from 6-mers only.*

6.1.4 The Effect of Increasing Labeled Data in Training Dataset

In this section we will analyze the effect of increasing the amount of labeled data in the Training Dataset. We used seven labeled data sizes as 1,000, 2,500, 6,500, 16,000, 25,000,

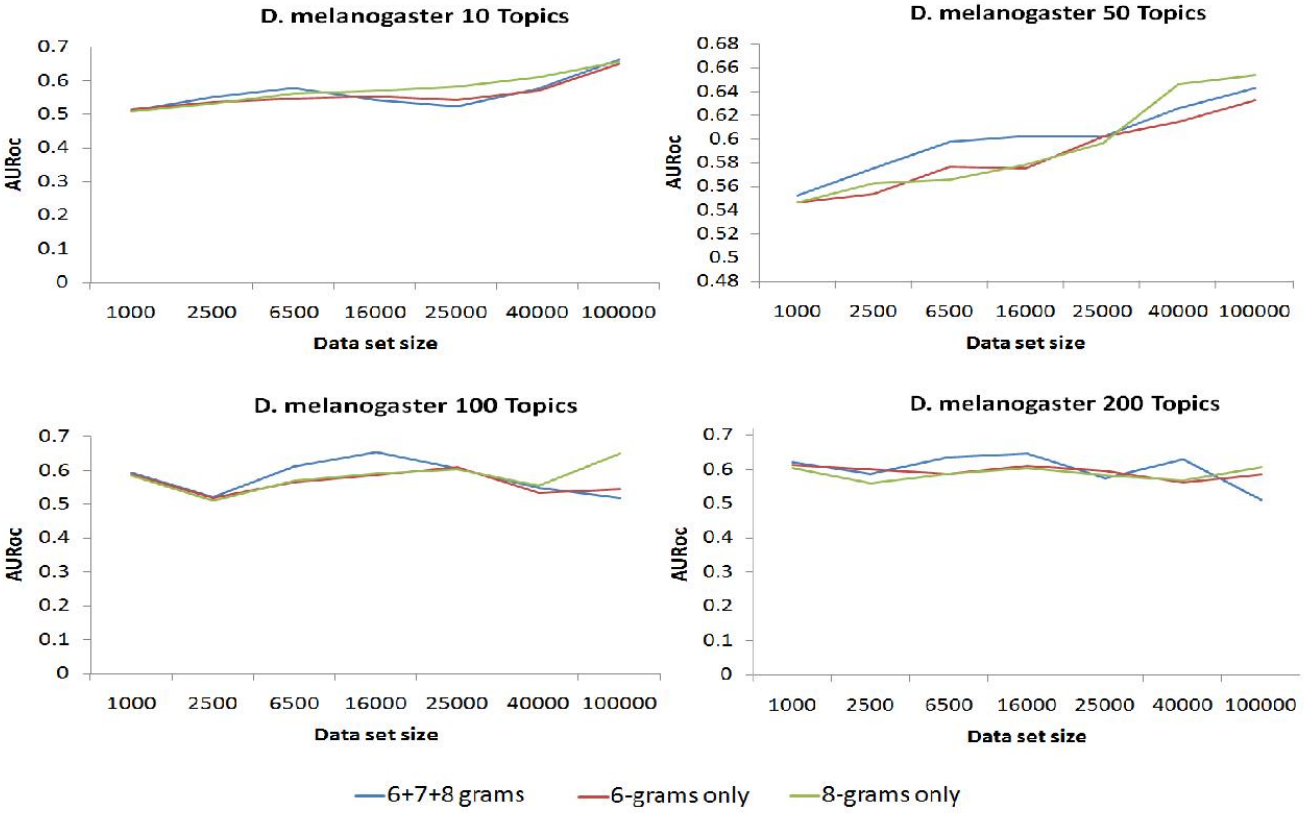


Figure 6.7: *Graphs obtained by plotting the auROC against the training data size for LDAW method with 10, 50, 100 and 200 LDA topics in supervised learning framework for D. melanogaster. Data series blue corresponds to features obtained from combination of 6-mers, 7-mers and 8-mers, data series green corresponds to features obtained from 8-mers only and data series maroon corresponds to features obtained from 6-mers only.*

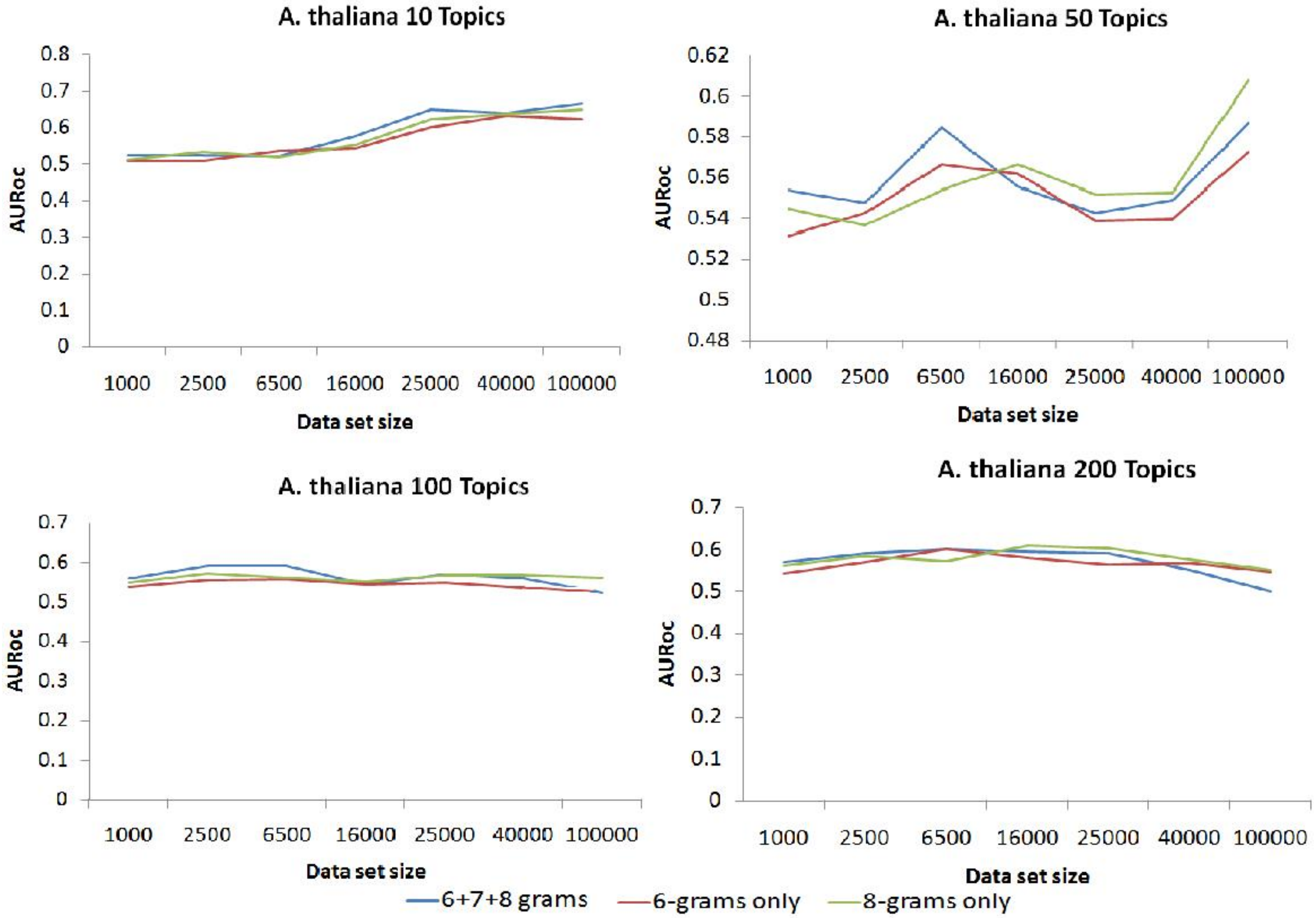


Figure 6.8: *Graphs obtained by plotting the auROC against the training data size for LDAW method with 10, 50, 100 and 200 LDA topics in supervised learning framework for A. thaliana. Data series blue corresponds to features obtained from combination of 6-mers, 7-mers and 8-mers, data series green corresponds to features obtained from 8-mers only and data series maroon corresponds to features obtained from 6-mers only.*

C. remanei							
	1000	2500	6500	16000	25000	40000	100000
10	0.52	0.538	0.566	0.625	0.615	0.64	0.655
50	0.504	0.556	0.55	0.548	0.538	0.537	0.601
100	0.553	0.554	0.597	0.521	0.511	0.535	0.563
200	0.535	0.541	0.58	0.546	0.542	0.524	0.591
D. melanogaster							
	1000	2500	6500	16000	25000	40000	100000
10	0.516	0.537	0.548	0.655	0.644	0.672	0.65
50	0.547	0.554	0.577	0.626	0.602	0.615	0.633
100	0.589	0.601	0.566	0.588	0.61	0.535	0.546
200	0.613	0.60	0.587	0.611	0.597	0.563	0.586
A. thaliana							
	1000	2500	6500	16000	25000	40000	100000
10	0.511	0.51	0.536	0.545	0.601	0.634	0.624
50	0.532	0.543	0.567	0.562	0.539	0.54	0.573
100	0.539	0.556	0.558	0.545	0.55	0.537	0.534
200	0.543	0.571	0.603	0.582	0.564	0.568	0.548

Table 6.5: *auROC values obtained from using only 6-mers features with LDAW method in supervised learning framework with logistic regression classifier. The target data set sizes used are 1000, 2500, 6500, 16000, 25000, 40000 and 100000, respectively (column names). The number of topic used are 10, 50, 100 and 200, respectively (row names).*

40,000 and 1,00,000 for training the classifier. We will see the effect of increasing labeled data size for both LDAW method and LDAD methods. Let us first look at the results for LDAW method.

Table 6.1 summarizes the results of LDAW method in supervised learning framework and using logistic regression classifier. The best auROC value in each of the row is highlighted in the table. When we look at the highlighted auROC values we can see that for topic count of 10 and 50 the best performance is obtained for training data set size of 100,000. However, for topic count of 100 and 200 the best performance is obtained for data set sizes of 6,500 or 16,000. Therefore, we can say that for small number of features the performance improves with larger training data set and for large number of features the performance is best with intermediate training data set size. If the data sets are small they cannot train the classifier

C. remanei							
	1000	2500	6500	16000	25000	40000	100000
10	0.509	0.547	0.586	0.622	0.653	0.659	0.663
50	0.573	0.55	0.592	0.586	0.591	0.605	0.621
100	0.581	0.573	0.575	0.562	0.548	0.571	0.632
200	0.594	0.598	0.608	0.594	0.587	0.576	0.645
D. melanogaster							
	1000	2500	6500	16000	25000	40000	100000
10	0.509	0.534	0.563	0.571	0.583	0.612	0.658
50	0.547	0.563	0.566	0.579	0.597	0.647	0.654
100	0.586	0.559	0.572	0.592	0.605	0.556	0.651
200	0.604	0.56	0.587	0.605	0.583	0.568	0.607
A. thaliana							
	1000	2500	6500	16000	25000	40000	100000
10	0.513	0.515	0.521	0.553	0.623	0.638	0.649
50	0.545	0.537	0.554	0.567	0.552	0.553	0.608
100	0.55	0.574	0.563	0.551	0.568	0.568	0.562
200	0.563	0.585	0.573	0.612	0.604	0.578	0.551

Table 6.6: *auROC values obtained from using only 8-mers features with LDAW method in supervised learning framework with logistic regression classifier. The target data set sizes used are 1000, 2500, 6500, 16000, 25000, 40000 and 100000, respectively (column names). The number of topic used are 10, 50, 100 and 200, respectively (row names).*

well when using a small number of features. Moreover, when we increase size of training data set the classifier can perform better even for small number of features, which explains why we obtained best auROC values for topic count of 10 and 50 with larger training data set sizes. Furthermore, for large training data set and large number of features the performance drop can be explained by over fitting.

Table 6.2 summarizes the results of LDAD method with supervised learning framework using logistic regression classifier. Again, we highlighted the best auROC value in each row. Unlike, the LDAW method, for the LDAD method the best auROC values are obtained for data sizes of 6,500, 16,000 or 25,000. Consequently, we can say that LDAD method performs best for intermediate sizes of training data set irrespective of the number of features used.

6.2 Domain Adaptation Framework Results

We analyze our results in terms of performance for different domain adaptation settings. We have already discussed about the experimental setup used in this framework and domain adaptation algorithms in Section 5.2 and Section 4.3, respectively. One of the main goals of this thesis is to study the performance of our approach for different domain adaptation algorithms with an increasing evolutionary distance between the source and the target domain. We have organized this section in two parts: in Section 6.2.1, we study the effect of increasing the evolutionary distance between the source and target species; in Section 6.2.2, we compare weighted domain adaptation approaches with target only and source only approaches.

6.2.1 The Effect of Increasing Source and Target Species Evolutionary Distance

We performed experiments using four domain adaptation approaches,DA_s, DA_{s+t} DA_{2s+t} and DA_{s+2t}, as discussed in Section 4.3. For these set of experiments we used both source and target domain training sequences for obtaining a LDA model. As opposed to DA_t, where we used training documents from target domain to obtain LDA model. We will analyze the effect of increasing evolutionary distance on performance of each of these approaches. We will begin with DA_s approach, then we will see the results of DA_{s+t} approach and finally we will discuss results from DA_{2s+t} and DA_{s+2t} approach.

Effect on DA_s

The results obtained from DA_s are summarizes in Table 6.7. These results are obtained using Logistic regression classifier. In order to analyze the effect of increasing evolutionary distance on performance we plotted the graph of auROC values for each of the target organism against the size of the training data set. Graphs shown in Figure 6.9 is for DA_s approach and for a topic count of 10, 50, 100 and 200. As you can see each data series represents a target species. It is clear from all four graphs that best auROC values are obtained from *C. remanei*, which is closest to our source species, *C. elegans*. The auROC for *D. melanogaster* and *A. thaliana* are very close, but in most of the cases *D. melanogaster* gave better results than *A. thaliana* and we know that *D. melanogaster* is closer to *C. elegans* than *A. thaliana*. Lastly, *P. pacificus* gave lowest auROC values. Exceptionally bad values of *P. pacificus* can be attributed to low quality of data. We can ignore the results from *P. pacificus* as it did not perform well even for DA_t approach, which was using only target data for training the classifier. We can easily conclude from the results of other three species that the performance of DA_s algorithm is better when the source and target species have smaller evolutionary distances.

Effect on DA_{s+t}

The results obtained from DA_{s+t} are summarizes in Table 6.8. We plotted graphs similar to what we plotted for DA_s, as discussed in the above section. These graphs are shown in Figure 6.10, which summarizes the results for a topic count of 10, 50, 100 and 200. Again, for all the four graphs we can see that best auROC values are obtained for *C. remanei*, then for *D. melanogaster*, then for *A. thaliana* and then *P.pacificus*. We ignore the results from *P.pacificus* for the same reason as mentioned in above section. We can say that the performance of DA_{s+t} approach gives better results when the source and target species have smaller evolutionary distances.

Effect on DA_{2s+t}

The results obtained from DA_{2s+t} are summarizes in Table 6.10. The graph plotted from the results of DA_{2s+t} algorithm are shown in Figure 6.11, which show results for a topic count of 10, 50, 100 and 200. These graphs are similar to the graphs plotted for DA_s and DA_{s+t}. Again, for all the four graphs we can see that best auROC values are obtained for *C. remanei*, then for *D. melanogaster*, then for *A. thaliana* and then *P.pacificus*. We ignore the results from *P.pacificus* for the same reason as mentioned in above section. We can say that the performance of DA_{2s+t} approach gives better results when the source and target species have smaller evolutionary distances.

Effect on DA_{s+2t}

The results obtained from DA_{s+2t} are summarizes in Table 6.9. The graph plotted from the results of DA_{s+2t} algorithm are shown in Figure 6.12. They are plotted in the similar way as we plotted the graphs in the above sections. However, the results from these graph are not similar to the results for the other two approaches. The data series for all the three species *C. remanei*, *D. melanogaster* and *A. thaliana* are very close to each other and there are few instances where either *D. melanogaster* or *A. thaliana* performs better than *C. remanei*. This can be explained by the fact that here we are using higher weight for the target instances that the source instances. Therefore, the effect of instances from source domain is reduced, which resulted in better results for *C. remanei* in case of DA_s and DA_{2s+t} approaches. Again, we are ignoring the results from *P. pacificus* for the reason given in above sections. We can thus conclude that DA_{s+2t} does not necessarily perform better for species with smaller evolutionary distance to the source domain, as in this approach the weight of target instances is higher and, therefore, the results are less effected by the source instances.

6.2.2 The Effect of Using Weighted Combination of Source and Target as Training Data

In this section, we will compare the performance of all DA_t, DA_s, DA_{s+t}, DA_{2s+t} and DA_{s+2t} approaches. This study will shed light on the effect of using different weighted combinations of source and target instances as training data. The results for DA_t, DA_s, DA_{s+t}, DA_{2s+t} and DA_{s+2t} approaches are summarized in Tables 6.1, 6.7, 6.8, 6.9 and 6.10, respectively. The classifier used to obtain all these results was the same, i.e. logistic regression. In order to compare these approaches we plotted a graph of auROC values obtained by using each of these approaches against increasing training data set sizes for each of the organisms, keeping

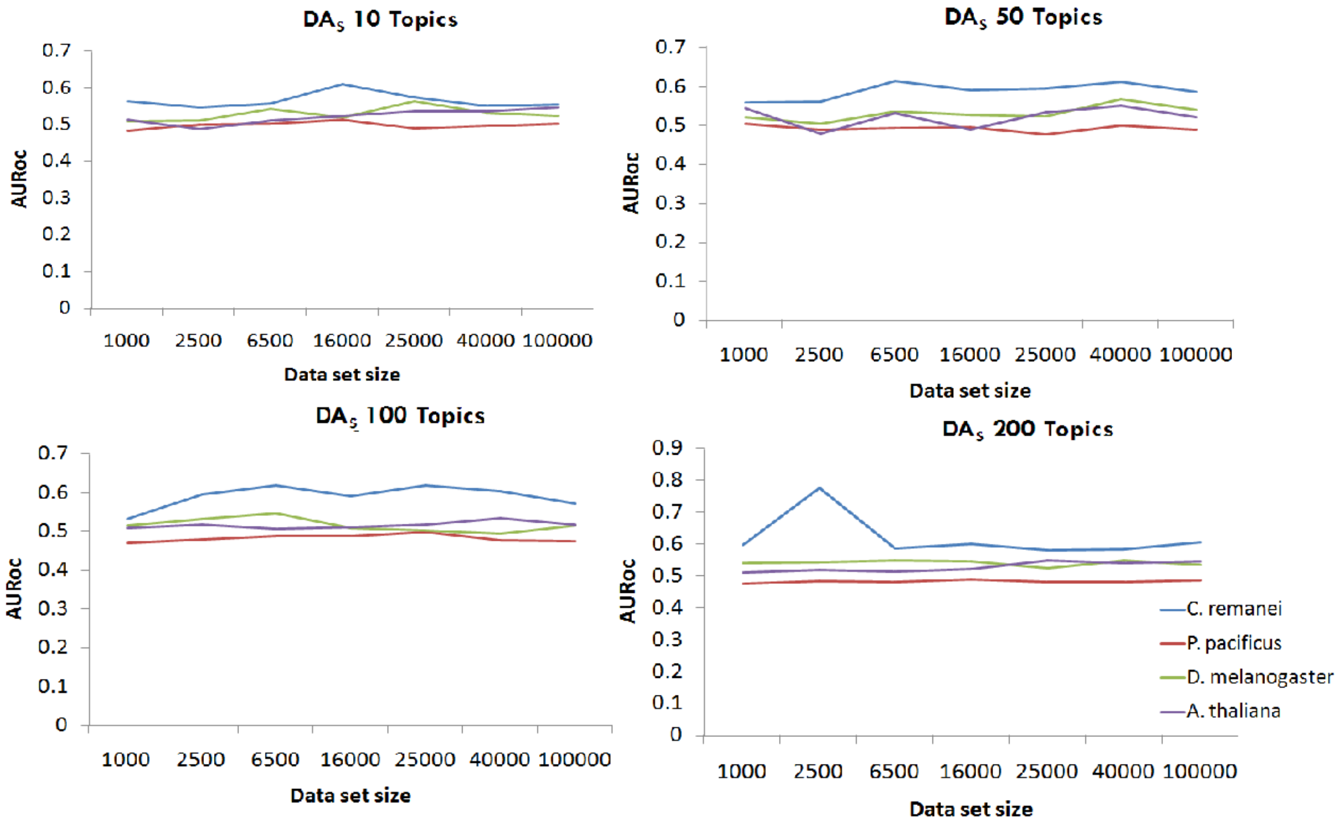


Figure 6.9: *Graphs obtained by plotting the auROC against the training data size for LDAW method with 10, 50, 100 and 200 LDA topics and DA$_s$ approach. Each data series corresponds to a different organism.*

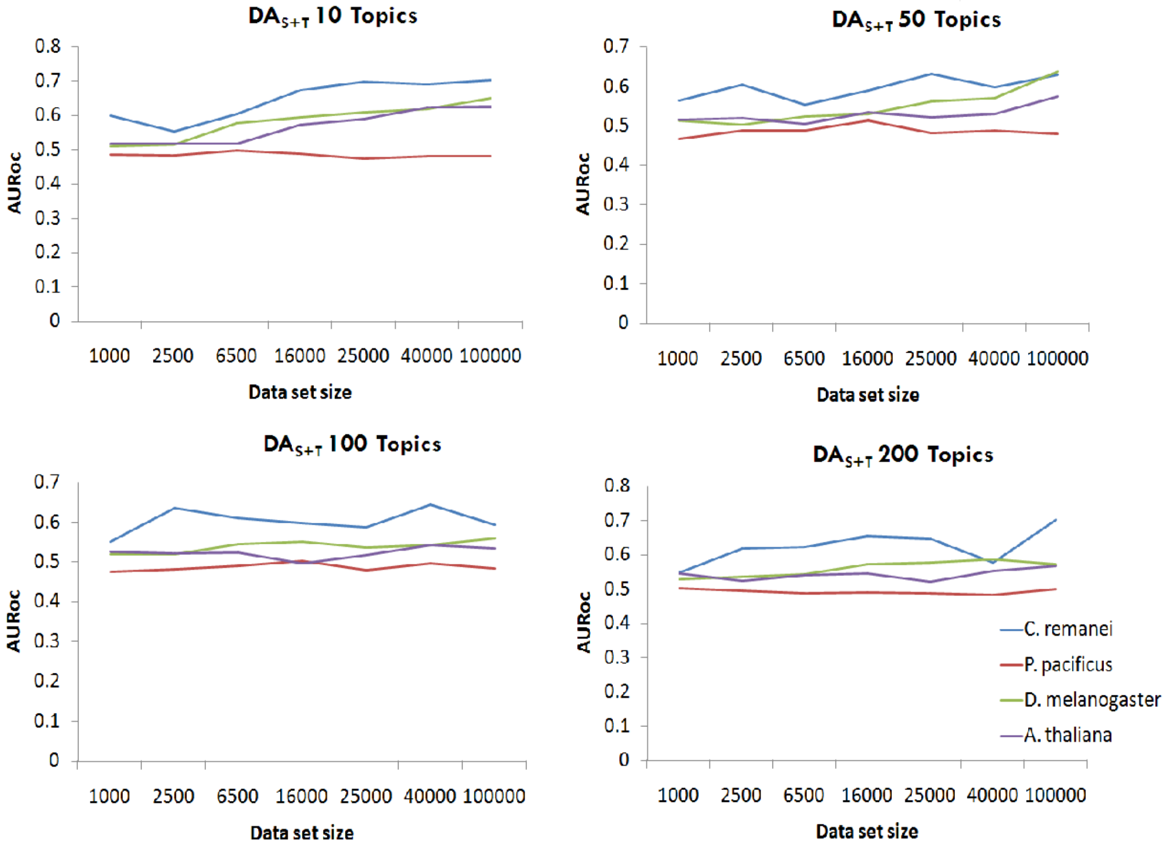


Figure 6.10: *Graphs obtained by plotting the auROC against the training data size for LDAW method with 10, 50, 100 and 200 LDA topics and DA_{s+t} approach. Each data series corresponds to a different organism.*

]

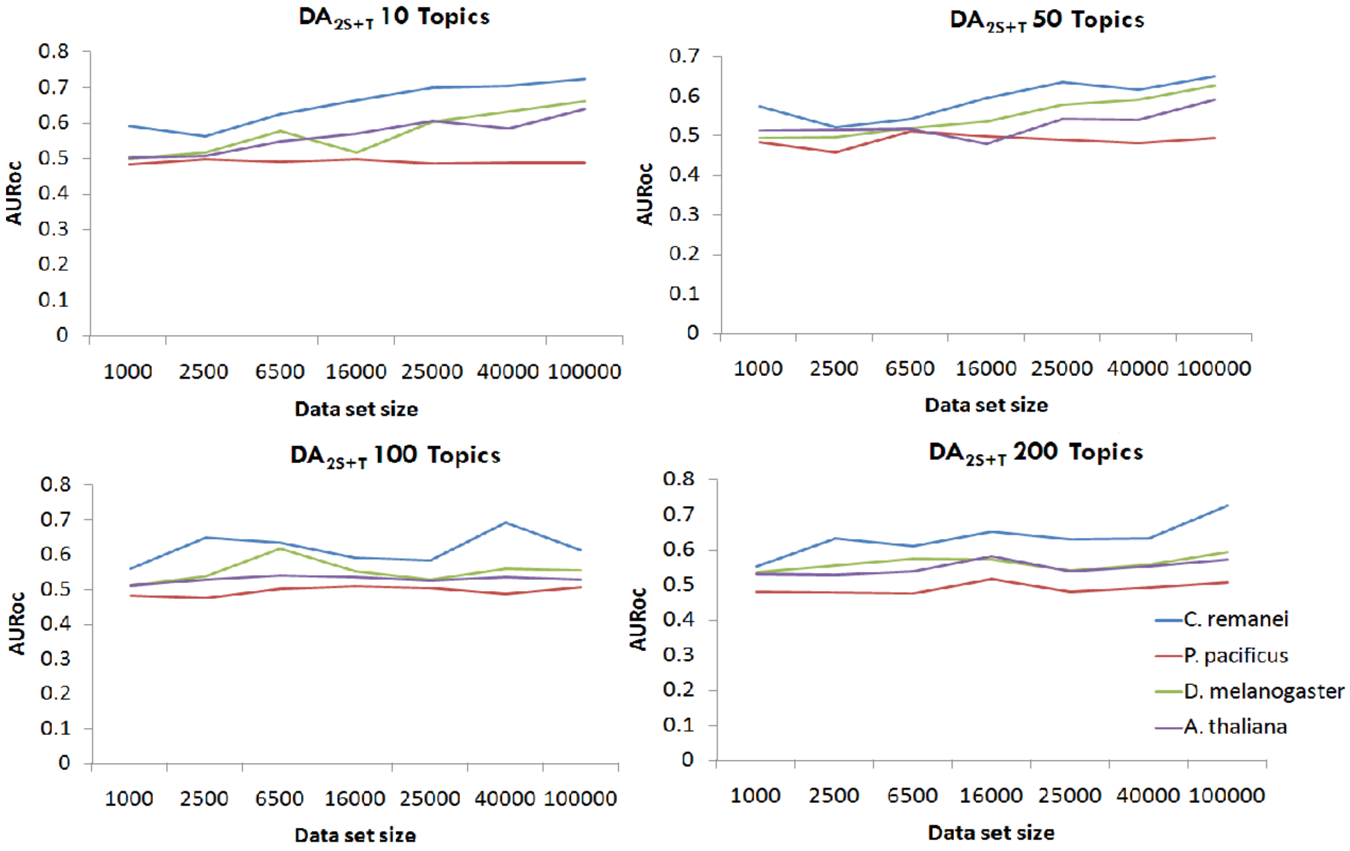


Figure 6.11: *Graphs obtained by plotting the auROC against the training data size for LDAW method with 10, 50, 100 and 200 LDA topics and* DA_{2s+t} *approach. Each data series corresponds to a different organism.*

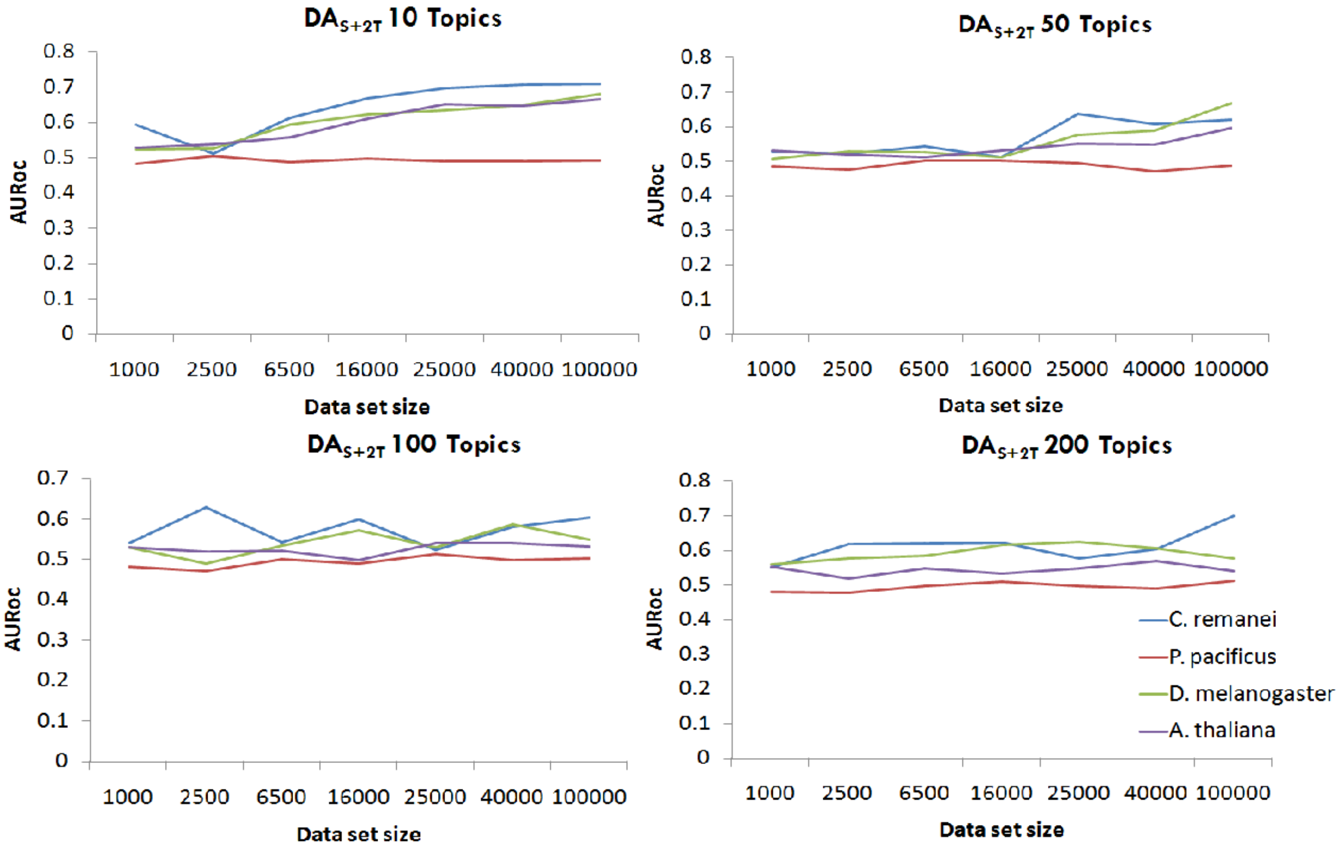


Figure 6.12: *Graphs obtained by plotting the auROC against the training data size for LDAW method with 10, 50, 100 and 200 LDA topics and* DA_{s+2t} *approach. Each data series corresponds to a different organism.*

C. remanei							
	1000	2500	6500	16000	25000	40000	100000
10	0.565	0.547	0.559	0.612	0.576	0.553	0.557
50	0.56	0.562	0.615	0.592	0.596	0.614	0.588
100	0.534	0.597	0.62	0.592	0.619	0.605	0.574
200	0.599	0.777	0.589	0.602	0.582	0.586	0.606
P. pacificus							
	1000	2500	6500	16000	25000	40000	100000
10	0.484	0.502	0.503	0.514	0.492	0.498	0.504
50	0.505	0.49	0.495	0.497	0.478	0.502	0.491
100	0.473	0.48	0.488	0.489	0.5	0.478	0.476
200	0.476	0.486	0.483	0.491	0.481	0.483	0.489
D. melanogaster							
	1000	2500	6500	16000	25000	40000	100000
10	0.509	0.512	0.543	0.521	0.565	0.534	0.524
50	0.522	0.506	0.537	0.529	0.524	0.569	0.542
100	0.516	0.533	0.547	0.509	0.503	0.496	0.516
200	0.541	0.545	0.551	0.548	0.527	0.551	0.537
A. thaliana							
	1000	2500	6500	16000	25000	40000	100000
10	0.514	0.489	0.513	0.524	0.538	0.538	0.548
50	0.546	0.48	0.533	0.492	0.535	0.551	0.523
100	0.51	0.518	0.507	0.512	0.519	0.535	0.518
200	0.513	0.519	0.515	0.522	0.55	0.541	0.548

Table 6.7: *auROC values obtained from LDAW method using DA_s approach with logistic regression classifier. The target data set sizes used are 1000, 2500, 6500, 16000, 25000, 40000 and 100000, respectively (column names). The number of features used are 10, 50, 100 and 200, respectively (row names).*

the number of topics constant. We will begin our discussion with results from *C. remanei*, then we will move to *D. melanogaster* and finally we will discuss results from *A. thaliana*. We have not plotted graphs for *P. pacificus* due to its poor data quality.

The graphs obtained for *C. remanei* for topic sizes 10, 50, 100 and 200 are shown in Figure 6.13. When we look at these graphs it is very difficult to find any trend showing that one of the approaches performed better than the rest. If we look at the graph in Figure 6.13 for 10 topics, we can see that four of the data series are almost overlapping.

C. remanei							
	1000	2500	6500	16000	25000	40000	100000
10	0.599	0.554	0.605	0.675	0.698	0.692	0.704
50	0.564	0.605	0.554	0.59	0.632	0.598	0.631
100	0.553	0.636	0.612	0.598	0.587	0.645	0.594
200	0.548	0.62	0.623	0.655	0.648	0.577	0.703
P. pacificus							
	1000	2500	6500	16000	25000	40000	100000
10	0.486	0.483	0.499	0.489	0.475	0.481	0.482
50	0.467	0.489	0.488	0.514	0.482	0.489	0.48
100	0.476	0.483	0.492	0.503	0.48	0.497	0.484
200	0.503	0.495	0.49	0.492	0.49	0.485	0.501
D. melanogaster							
	1000	2500	6500	16000	25000	40000	100000
10	0.51	0.516	0.579	0.595	0.609	0.62	0.651
50	0.514	0.503	0.525	0.532	0.562	0.57	0.639
100	0.521	0.521	0.546	0.552	0.537	0.544	0.56
200	0.531	0.537	0.545	0.572	0.578	0.588	0.573
A. thaliana							
	1000	2500	6500	16000	25000	40000	100000
10	0.517	0.518	0.517	0.573	0.591	0.623	0.625
50	0.516	0.52	0.505	0.536	0.522	0.532	0.576
100	0.527	0.523	0.524	0.498	0.519	0.543	0.536
200	0.547	0.524	0.542	0.547	0.523	0.554	0.569

Table 6.8: *auROC values obtained from LDAW method using DA_{s+t} approach with logistic regression classifier. The target data set sizes used are 1000, 2500, 6500, 16000, 25000, 40000 and 100000, respectively (column names). The number of features used are 10, 50, 100 and 200, respectively (row names).*

Since, *C. remanei* is very close the source domain species *C. elegans* we can assume that the distribution of both the species is more or less the same. Thus, when we compared the performance of DA_s and DA_t we did not find one approach better than the other. Similarly, performance of DA_{s+t}, DA_{2s+t} and DA_{s+2t} did not show much differences.

The graphs obtained for *D. melanogaster* for topic sizes 10, 50, 100 and 200 are shown in Figure 6.14. From all the four graphs it can be observed that the performance of DA_s is worst and performance of DA_t is the best among all the four approaches. Further, when we

C. remanei							
	1000	2500	6500	16000	25000	40000	100000
10	0.594	0.513	0.615	0.669	0.698	0.707	0.711
50	0.53	0.523	0.544	0.512	0.638	0.609	0.621
100	0.542	0.629	0.544	0.601	0.525	0.581	0.604
200	0.552	0.62	0.621	0.624	0.579	0.605	0.701
P. pacificus							
	1000	2500	6500	16000	25000	40000	100000
10	0.483	0.505	0.488	0.499	0.491	0.492	0.493
50	0.486	0.476	0.503	0.504	0.496	0.473	0.488
100	0.483	0.473	0.501	0.492	0.514	0.5	0.503
200	0.482	0.48	0.499	0.511	0.498	0.492	0.513
D. melanogaster							
	1000	2500	6500	16000	25000	40000	100000
10	0.525.	0.528	0.596	0.623	0.636	0.65	0.681
50	0.507	0.531	0.528	0.514	0.578	0.589	0.67
100	0.53	0.491	0.536	0.574	0.532	0.588	0.55
200	0.56	0.578	0.585	0.616	0.625	0.608	0.579
A. thaliana							
	1000	2500	6500	16000	25000	40000	100000
10	0.531	0.539	0.558	0.611	0.652	0.647	0.666
50	0.533	0.52	0.513	0.533	0.552	0.548	0.598
100	0.53	0.521	0.522	0.5	0.541	0.542	0.534
200	0.555	0.521	0.55	0.535	0.548	0.571	0.543

Table 6.9: *auROC values obtained from LDAW method using DA_{s+2t} approach with logistic regression classifier. The target data set sizes used are 1000, 2500, 6500, 16000, 25000, 40000 and 100000, respectively (column names). The number of features used are 10, 50, 100 and 200, respectively (row names).*

compare DA_{s+2t} and DA_{s+2t} we see that DA_{s+2t} gives better results than DA_{2s+t}. Since, the evolutionary distance between *D. melanogaster* and the source species is large, DA_s approach does not give very good results. For same reason performance of DA_{2s+t} is not better than DA_{s+2t}. Giving more weight to source domain instances will only result in worsening of classification performance.The performance of DA_{s+t} is closer to DA_{2s+t} in some cases and in some cases it is closer to DA_{s+2t}.

The graphs obtained for *A. thaliana* for topic sizes 10, 50, 100 and 200 are shown in

C. remanei							
	1000	2500	6500	16000	25000	40000	100000
10	0.592	0.563	0.625	0.664	0.7	0.705	0.725
50	0.575	0.523	0.544	0.596	0.636	0.617	0.652
100	0.562	0.650	0.635	0.592	0.585	0.694	0.614
200	0.555	0.633	0.611	0.652	0.630	0.634	0.728
P. pacificus							
	1000	2500	6500	16000	25000	40000	100000
10	0.484	0.499	0.492	0.498	0.487	0.489	0.488
50	0.485	0.459	0.511	0.499	0.491	0.483	0.496
100	0.484	0.477	0.503	0.511	0.506	0.489	0.508
200	0.482	0.479	0.477	0.518	0.482	0.493	0.509
D. melanogaster							
	1000	2500	6500	16000	25000	40000	100000
10	0.502	0.517	0.579	0.517	0.605	0.634	0.662
50	0.495	0.497	0.521	0.537	0.58	0.592	0.627
100	0.514	0.539	0.62	0.554	0.53	0.56	0.556
200	0.536	0.557	0.575	0.572	0.542	0.558	0.596
A. thaliana							
	1000	2500	6500	16000	25000	40000	100000
10	0.504	0.507	0.548	0.571	0.607	0.586	0.641
50	0.515	0.516	0.519	0.481	0.543	0.541	0.593
100	0.512	0.53	0.542	0.538	0.527	0.536	0.531
200	0.533	0.531	0.539	0.582	0.539	0.554	0.574

Table 6.10: *auROC values obtained from LDAW method using DA_{2s+t} approach with logistic regression classifier. The target data set sizes used are 1000, 2500, 6500, 16000, 25000, 40000 and 100000, respectively (column names). The number of features used are 10, 50, 100 and 200, respectively (row names).*

Figure 6.15. The observations made from *A. thaliana* are similar to the observations for *D. melanogaster*. Again, the worst performance is obtained for DA_s approach and in most of the cases DA_{s+2t} performed better than DA_{2s+t}. We can conclude that as the evolutionary distance between *A. thaliana* is very large from the source domain, we did not obtain good results for DA_s approach and assigning higher weight to the source instances in DA_{2s+t} did not help to improve classification performance. The performance of DA_{s+t} is sometimes close to DA_{2s+t} and sometimes close to DA_{s+2t}.

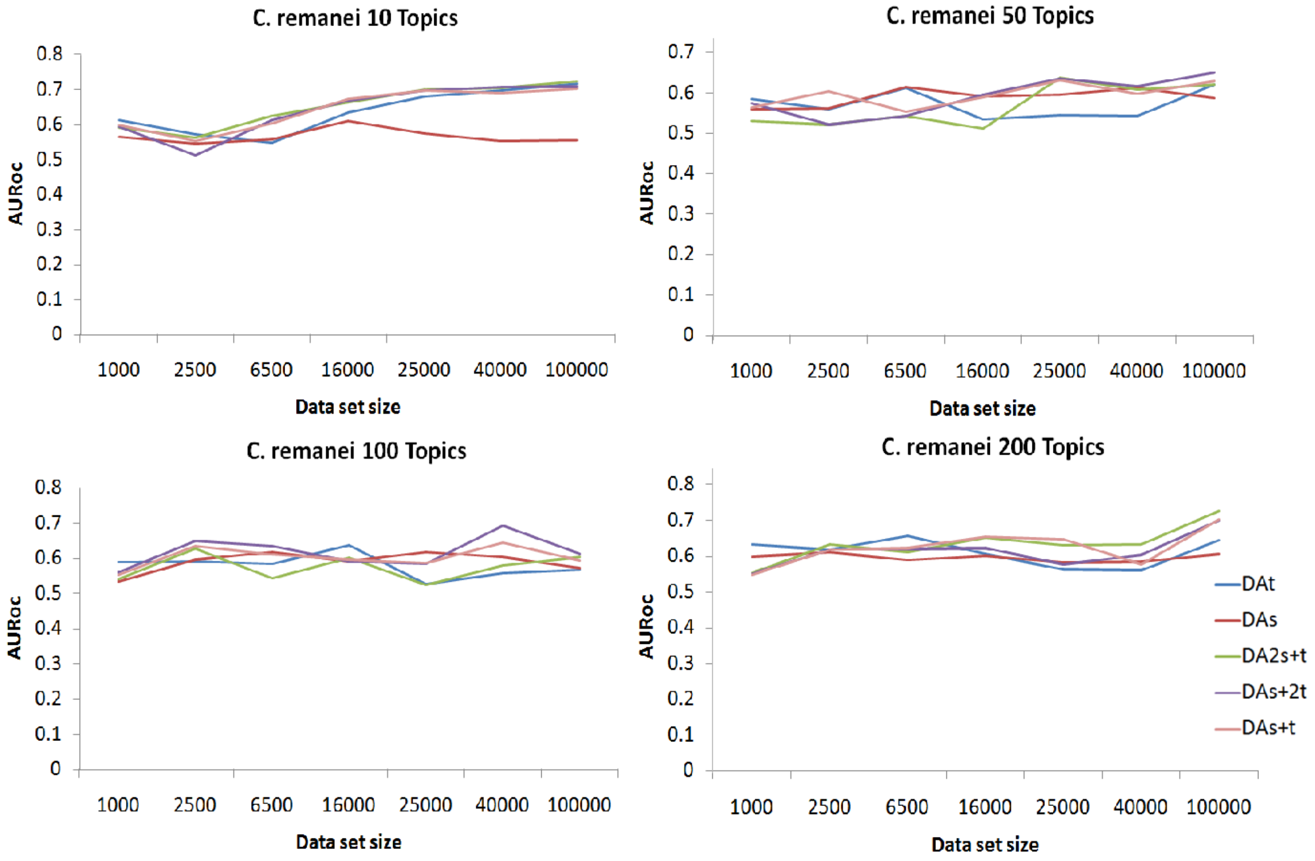


Figure 6.13: *Graphs obtained by plotting the auROC against the training data size for LDAW method with 10, 50, 100 and 200 LDA topics for C. remanei. Data series blue is for DA_t approach, maroon is for DA_s approach, pink is for DA_{s+t}, green is for DA_{2s+t} approach and purple is for DA_{s+2t} approach.*

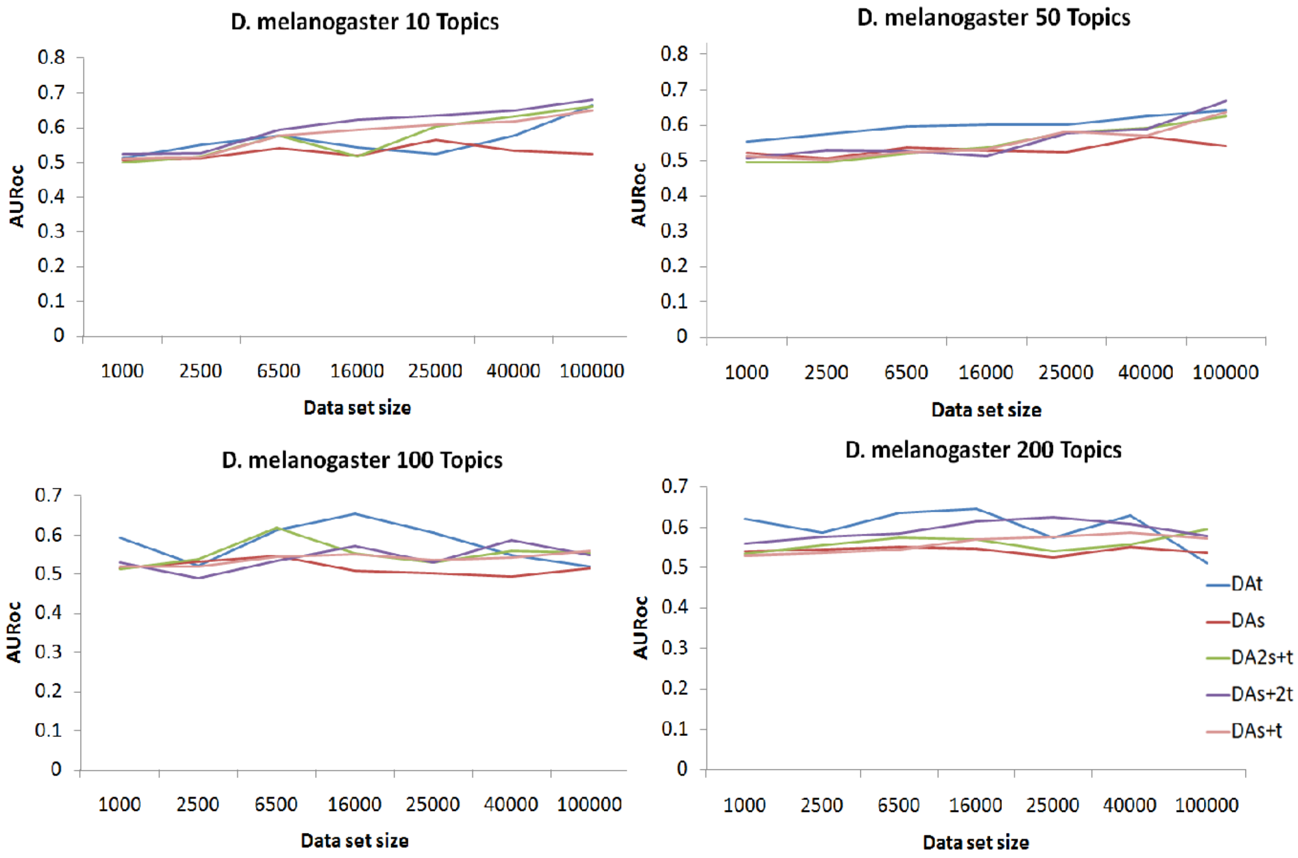


Figure 6.14: *Graphs obtained by plotting the auROC against the training data size for LDAW method with 10, 50, 100 and 200 LDA topics for D. melanogaster. Data series blue is for DA_t approach, maroon is for DA_s approach, pink is for DA_{s+t}, green is for DA_{2s+t} approach and purple is for DA_{s+2t} approach.*

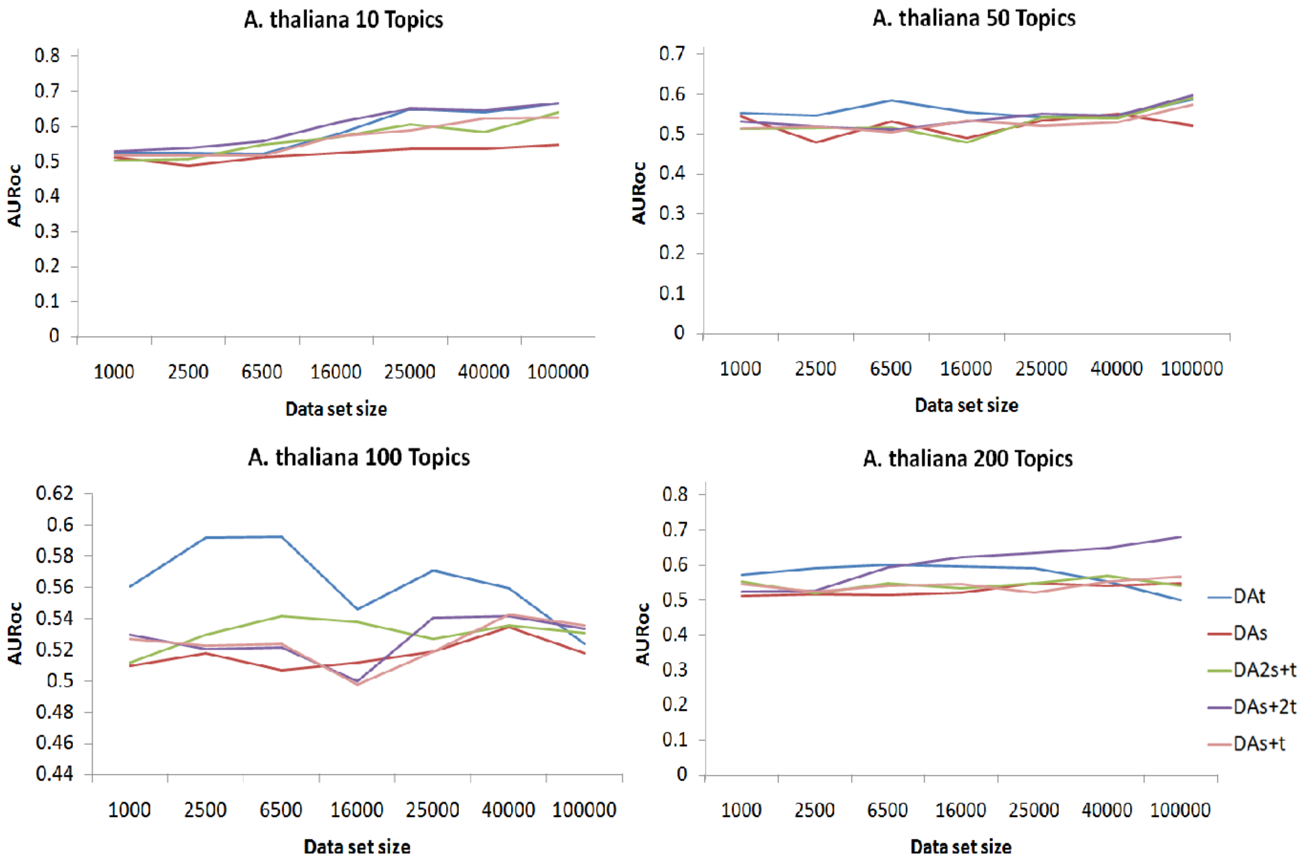


Figure 6.15: *Graphs obtained by plotting the auROC against the training data size for LDAW method with 10, 50, 100 and 200 LDA topics for A. thaliana. Data series blue is for DA_t approach, maroon is for DA_s approach, pink is for DA_{s+t}, green is for DA_{2s+t} approach and purple is for DA_{s+2t} approach.*

Chapter 7

Conclusion and Future Work

In this chapter, we being by presenting the conclusions of our work by answering the questions which we raised in Chapter 5 based on the results which we obtained in Chapter 6. Finally we showcase some related problems that we would like to address in future work.

7.1 Conclusion

In Chapter 5, we listed several settings for which we wanted to study the effectiveness of LDA as a dimensionality reduction tool for biological sequences. The following are insights we gained after thoroughly investigating the results of our experiments.

- We compared the performance of LDAW and LDAD methods in the first part of Section 6.1.1. We can conclude from these results that LDAW approach gives better performance as compared to LDAD approach for all the organisms irrespective of the training data sizes and number of LDA topics used.

- We compared the performance of LDAW approach with the traditional dimensionality reduction approach called MI method in the second part of Section 6.1.1. Investigation of these results suggest that the LDAW method is a better method that the MI method for larger data sets. Also, we found that for the MI method the classification performance increases with the increase in the number of features selected, however, it decreases after a threshold on the number of features selected is reached.

- We compared the performance of LDAW approach with logistic regression classifier and SVM classifier in Section 6.1.2. We found that the classifier performance depends on the organisms, i.e. the type of data. For *C.remanei*, SVM classifier performance is better than the performance of the logistic regression classifier, except for larger number of topics. Only for training data set sizes of 1,000, 2,500, 6,500 and 1,00,000, when we take words from 200 topics, logistic regression gives better results than SVM classifier. Contrarily, for both *A. thalians and D. melanogaster*, the logistic regression classifier performs better than SVM classifier in most of the cases. And for *P. pacificus* for all the training data set sizes and number of topics logistic regression classifier performed better than SVM classifier.

- We studies the effect of increasing number of LDA topics on classification performance for both LDAW and LDAD method in the first part of Section 6.1.3. Based on the results which we examined in this section we concluded that the performance of LDAW is better for large number of topics when the train data set is small, whereas the performance of LDAW is better for small number of topics when the train data set is large. Nonetheless, the performance of LDAD is better for large number of topics for almost all the sizes of train data set.

- We reviewed the effect of increasing the size of k-mer and effect of using combination of k-mers as features in the second part of Section 6.1.3. After analyzing the results from this section we concluded that the performance of LDAW method would improve with the increase in the size of k-mers used for obtaining LDAW features. Also, for smaller training data sets 6+7+8 mer features perform better, whereas for larger training data sets 8-mers only features perform better.

- We studied the effect of increasing the labeled data for both LDAW and LDAD methods in Section 6.1.4. We observed that for the LDAD method, for a small number of features the performance improves with a larger training data set and for large number of features the performance is best with intermediate training data set sizes. On the other hand, for LDAD method the best auROC values are obtained for data sizes of 6,500, 16,000 or 25,000 and it is independent of the number of features.

- We investigated the effect of increasing source and target species evolutionary distance for TL_s, TL_{s+2t} and TL_{2s+t} approaches in Section 6.2.1. Based on the results which we analyzed in this section we can conclude that for TL_s algorithm performs better when the source and target species have smaller evolutionary distance. TL_{2s+t} gives better results when the source and target species have smaller evolutionary distance. TL_{s+2t} does not perform better for species with smaller evolutionary distance to the source domain, as in this approach the weight of target instances is more, the effect of source instances is suppressed.

- We studied the effect of using weighted combination of source and target as training data in Section 6.2.2. The analysis of the results obtained in this section suggested that TL_s and TL_{2s+t} are better only when the evolutionary distance between source and target species are small.

Use a variable number of words in each topic (e.g., based on a threshold on the word probability).

7.2 Future Work

This section showcases several related problems that we would like to address in future work. They are briefly described in what follows:

- We used a combination of k-mers (6-mers, 7-mers and 8-mers) for our experiments. It is possible that a shorter k-mer may be a subsequence of a longer k-mer. For example, a 6-mer sequence *aaggtt* is a subsequence of a 8-mer sequence *caaggtta*. The presence of such sequences leads to redundant features. In future work we would like to keep only the longer k-mer and remove all shorter overlapping sequence.

- In this study, we considered LDA topic counts of 10, 50, 100 and 200. We observed that for LDAW, for smaller datasets 100 and 200 topics gave better results than 10 and 50 topics. These results were counter intuitive led us to believe that they are the effect of over fitting. Thus, we would like to use a denser topic scale (e.g., generate all topic models between 10 and 1000, with a step of 10) to understand better the variation of the performance with the number of topics, for both LDAW and LDAD.

- We used top 10 words from each topic as our features in LDAW. However, in the future we would like to use a variable number of words in each topic based on a threshold on the word probabilities.

- In this study, we used only *C. elegans* as our source domain. However, as a part of future work we can use a combination of several source data sets. We have training data from five organisms and only one was used as source domain. We can combine data from two or more organisms to form the source domain and see how it effects the classification performance.

- In this work, we studied mRNA splicing classification task. However, we can apply the LDAW or LDAD methods or obtain features for other biological problems. Once such biological problem which uses machine learning is motif-based machine learning approach for identifying intergenic regulatory elements is described by Bahirwani [2010].

Bibliography

U. Abel, E. Laurence, and B. Ewan. Comparative genomics: Genome-wide analysis in metazoan eukaryotes. *Nat Rev Genet*, 4(4):251–262, April 2003.

M. A. Aizerman, E. A. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. In *Automation and Remote Control*, pages 821–837, 1964.

A. Arnold, R. Nallapati, and W. W. Cohen. A comparative study of methods for transductive transfer learning. In *ICDM Workshop on Mining and Management of Biological Data*, 2007.

A. Arnold, R. Nallapati, and W. W. Cohen. Exploiting feature hierarchy for transfer learning in named entity recognition. In *In ACL:HLT 08*, 2008.

V. Bahirwani. Exploring transcription patterns and regulatory motifs in arabidopsis thaliana. Master's thesis, Kansas State University, 2010.

D. Blei, Y. Andrew, and J. Michael. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, January 2003.

D. Blei, K. Franks, M. Jordan, and I. Mian. Statistical modeling of biomedical corpora: mining the caenorhabditis genetic center bibliography for genes related to life span. *BMC Bioinformatics*, 7, May 2006.

B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152. ACM Press, 1992.

J. Boyd-Graber, D. M. Blei, and X. Zhu. A topic model for word sense disambiguation. In *Empirical Methods in Natural Language Processing*, 2007.

CA. Collins and C. Guthrie. Allele-specific genetic interactions between prp8 and RNA active site residues suggest a function for prp8 at the catalytic core of the spliceosome. In *Genes and Dev*, 1999.

C. Cortes and V. Vapnik. Support-vector networks. In *Machine Learning*, pages 273–297, 1995.

K. Crammer, Y. Singer, N. Cristianini, J. Shawe-taylor, and B. Williamson. On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:2001, 2001.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 1 edition, 2000.

H. DaumeIII. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. Association for Computational Linguistics, 2007.

S. Degroeve, B. De Baets, Y. Van De Peer, and P. Rouze. Feature subset selection for splice site prediction. *Bioinformatics*, 18 Suppl 2:S75–S83, October 2002.

B. Douglas. Mechanisms of alternative pre-messenger RNA splicing. *Annual Review of Biochemistry*, 72(1):291–336, 2003.

Imola Fodor. A survey of dimension reduction techniques, 2002. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.8.5098.

J. Guo, G. Xu, X. Cheng, and H. Li. Named entity recognition in query. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 267–274. ACM, 2009.

M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

S. M. Hebsgaard, P. G. Korning, N. Tolstrup, J. Engelbrecht, P. Rouze, and S. Brunak. Splice site prediction in arabidopsis thaliana pre-mRNA by combining local and global sequence information. *Nucleic Acids Res*, 24(17):3439–52+, 1996.

R. Islamaj, L. Getoor, and J. Wilbur. Wj: A feature generation algorithm for sequences with application to splice-site prediction. In *In Proceedings of European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 553–560, 2006.

J. Johnson, J. Castle, P. Garrett-Engele, Z. Kan, P. Loerch, C. Armour, R. Santos, E. Schadt, R. Stoughton, and D. Shoemaker. Genome-wide survey of human alternative pre-mRNA splicing with exon junction microarrays. *Science (New York, N.Y.)*, 302 (5653):2141–2144, December 2003.

D. G. Kleinbaum, M. Klein, and E. R. Pryor. *Logistic regression : a self-learning text*. New York : Springer-Verlag, 1994.

D. Koller and M. Sahami. Toward optimal feature selection. In *ICML*, pages 284–292. Morgan Kaufmann, 1996.

R. Krestel, P. Fankhauser, and W. Nejdl. Latent Dirichlet allocation for tag recommendation. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 61–68. ACM, 2009.

H. Liu and H. Motoda. *Feature extraction, construction and selection : a data mining perspective*, volume SECS 453. Kluwer Academic, 1998.

Z. Marketa and B. Jeremy. *Understanding bioinformatics*. New York : Garland Science, 2008.

C. Mathe, M. Sagot, T. Schiex, and P. Rouze. Current methods of gene prediction, their strengths and weaknesses. *Nucl. Acids Res.*, 30(19):4103–4117, October 2002.

A. McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

A. McCallum and K. Nigam. A comparison of event models for naive Bayes text classification. In *IN AAAI-98 WORKSHOP ON LEARNING FOR TEXT CATEGORIZATION*, pages 41–48. AAAI Press, 1998.

T. Mitchell. *Machine Learning*. McGraw Hill, 1997.

B. Monya. Next-generation sequencing: adjusting to data overload. *Nature Methods*, 7(7): 495–499, July 2010.

TW. Nilsen. RNARNA interactions in the spliceosome: Unraveling the ties that bind. In *Cell*, page 14, 1994.

R. Parimi. LDA based approach for predicting friendship links in live journal social network. Master's thesis, Kansas State University, 2010.

H. Pearson. Genetics: what is a gene? *Nature*, May 2006.

A. Perina, P. Lovato, V. Murino, and M. Bicego. Biologically-aware Latent Dirichlet Allocation (BaLDA) for the classification of expression microarray. In *PRIB'10*, pages 230–241, 2010.

A. Pires-Dasilva and R. J. Sommer. Conservation of the global sex determination gene tra-1 in distantly related nematodes. *Genes and Development*, 18:1198–1208, 2004.

D. L. Riddle, T. E. Blumenthal, B. J. Meyer, and J. R. Priess. *C. elegans II*. Cold Spring Harbor Laboratory Press, 1997.

S. Rogers, M. Girolami, C. Campbell, and R. Breitling. The latent process decomposition of cDNA microarray datasets. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2, 2005.

B. Scholkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* Adaptive Computation and Machine Learning. MIT Press, 12 2002.

B. Scholkopf, K. Tsuda, and J. P. Vert, editors. *Kernel Methods in Computational Biology.* MIT Press, 2004.

B. Scholkopf, K. Tsuda, and J. P. Vert. Kernel methods in genomics and computational biology, 2005.

G. Schweikert, C. Widmer, B. Scholkopf, and G. Ratsch. An empirical analysis of domain adaptation algorithms for genomic sequence analysis. In *Advances in neural information processing systems 21 : 22nd Annual Conference on Neural Information Processing Systems*, June 2009.

W. Scott and W. Gilbert. The evolution of spliceosomal introns: patterns, puzzles and progress. *Nature Reviews Genetics*, 7(3), 2006.

S. Sonnenburg. New methods for splice site recognition. Master's thesis, Humbold-University zu Berli, 2002.

L. D. Stein, Z. Bao, D. Blasiar, T. Blumenthal, M. R. Brent, N. Chen, A. Chinwalla, L. Clarke, C. Clee, A. Coghlan, A. Coulson, P. D'Eustachio, A. Fitch, L. A. Fulton, R. E. Fulton, S. Griffiths-Jones, T. W. Harris, L. W. Hillier, R. Kamath, P. E. Kuwabara, E. R. Mardis, M. A. Marra, T. L. Miner, P. Minx, J. C. Mullikin, R. W. Plumb, J. Roger, J. E. Schein, M. Sohrmann, J. Spieth, J. E. Stajich, C. Wei, D. Willey, R. K. Wilson, R. Durbin, and R. H. Waterston. The genome sequence of caenorhabditis briggsae: a platform for comparative genomics. *PLoS Biol*, 1(2):166–192, November 2003.

M. Steyvers and T. Griffiths. *Latent Semantic Analysis: A Road to Meaning*, chapter Probabilistic topic models. Laurence Erlbaum, 2007.

S. Tasci and T. Gungor. LDA-based keyword selection in text categorization. In *International Symposium on Computer and Information Sciences*, pages 230–235, September 2009.

I. Vukusic, S. Grellscheid, and T. Wiehe. Applying genetic programming to the prediction of alternative mRNA splice variants. *Genomics*, 89(4):471–479, April 2007.

C. Widmer, J. Leiva, Y. Altun, and G. Ratsch. Leveraging sequence classification by taxonomy-based multitask learning. In *Research in Computational Molecular Biology*, volume 6044, chapter 34, pages 522–534. Springer Berlin / Heidelberg, 2010.

G. Yeo and C. Burge. Maximum entropy modeling of short sequence motifs with applications to RNA splicing signals. *J. Comput. Biol*, 11:377–394, 2004.

L. Yu and H. Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *ICML*, pages 856–863, 2003.

M. Zhang. Computational prediction of eukaryotic protein-coding genes. *Nature Reviews Genetics*, 2002.

X. Zhang, K. Heller, I. Hefter, C. Leslie, and L. Chasin. Sequence information for the splicing of human pre-mRNA identified by support vector machine classification. *Genome Res*, 13:2003, 2003a.

X. Zhang, H. Katherine, I. Hefter, C. S. Leslie, and L. A. Chasin. Sequence information for the splicing of human Pre-mRNA identified by support vector machine classification. *Genome Res*, 13:2003, 2003b.

C. Zheng, T. M. Nair, M. Gribskov, Y. S. Kwon, Hai-Ri Li, and Xiang-Dong Fu. A database designed to computationally aid an experimental approach to alternative splicing. In *Pacific Symposium on Biocomputing*, pages 78–88, 2004.

Appendix A

Basic Terminology

- auROC: Short for *area under receiver operating characteristic (ROC) curve.* ROC is a graphical plot of true positive rate, vs. false positive rate for a binary classifier system as its discrimination threshold is varied.

- Bioinformatics: Bioinformatics is the application of information technology and computer science to the field of molecular biology [Marketa and Jeremy, 2008].

- Classifier: Classifier maps input data into defined output categories based on the characteristics of input (also called features).

- Consensus Sequence: Consensus sequence refers to the most common nucleotide or amino acid at a particular position after multiple sequences are aligned [Marketa and Jeremy, 2008].

- Domain adaptation: Domain adaptation is a subproblem of transfer learning where a model trained over a source domain is generalized to perform well on a related target domain, here the two domains data are distributed similarly, but not identically [Arnold et al., 2008].

- Dimensionality Reduction: Given a p-dimensional random variable $\mathbf{x} = (x_1, ..., x_p)^T$, and a lower dimensional representation of it, $\mathbf{s} = (s_1, ..., s_k)^T$ with $k \leq p$, that captures the content in the original data according to some criterion, we say that $\mathbf{s}$ is a representation of $\mathbf{x}$ in a reduced dimensionality space [Fodor, 2002].

- DNA: Short for *deoxyribonucleic acid*. The nucleic acid that is the genetic material determining the makeup of all living cells and many viruses. It consists of two long strands of nucleotides linked together in a structure resembling a ladder twisted into a spiral. In eukaryotic cells, the DNA is contained in the nucleus (where it is bound to proteins known as histones) and in mitochondria and chloroplasts. In the presence of the enzyme DNA polymerase and appropriate nucleotides, DNA can replicate itself (http://www.thefreedictionary.com/DNA).

- Features: Features are the individual measurable heuristic properties of the phenomena being observed.

- Latent Dirichlet Allocation: Latent Dirichlet Allocation (LDA), a generative probabilistic model for collections of discrete data such as text corpora. LDA is a three-level hierarchical Bayesian model, in which each item of a collection is modeled as a finite mixture over an underlying set of topics. Each topic is, in turn, modeled as an infinite mixture over an underlying set of topic probabilities [Blei et al., 2003].

- K-mer: k-mer is a subsequence of a nucleic acid or amino acid sequence of k nucleotides.

- Nucleotide: A compound consisting of a nucleoside linked to a phosphate group. Nucleotides form the basic structural unit of nucleic acids such as DNA.

- Splice site(donor/acceptor): The intron-exon boundary in a gene is splice site. The beginning of an intron is a donor site and end of an intron is acceptor site.

- Supervised learning: Supervised learning is the machine learning task of producing an inferred function, which is called a classifier by making use of training data. The inferred function should predict the correct output value for any valid input object.

- Transfer learning: Transfer learning is a problem in machine learning, where information gained in one learning task is used to improve performance in another related task [Arnold et al., 2008].
- Unsupervised learning: Unsupervised learning is the machine learning task of inferring a hidden structure of unlabeled data.