

AMERICAN UNIVERSITY OF BEIRUT

A MAP REDUCE SEISMIC TEXTURE ANALYSIS
AND BARRICADED BOUNDARY MINORITY LS-SVM
FRAMEWORK FOR MARINE SEISMIC EXPLORATION
DATA

by
HMAYAG KEVORK PARTAMIAN

A thesis
submitted in partial fulfillment of the requirements
for the degree of Master of Science
to the Department of Computational Science
of the Faculty of Arts and Sciences
at the American University of Beirut

Beirut, Lebanon
May, 2014

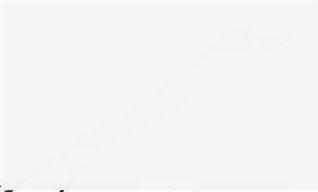
AMERICAN UNIVERSITY OF BEIRUT

A MAP REDUCE SEISMIC TEXTURE ANALYSIS AND
BARRICADED BOUNDARY MINORITY LS-SVM FRAMEWORK
FOR MARINE SEISMIC EXPLORATION DATA


by
HMAYAG KEVORK PARTAMIAN

Approved by:

Dr. Mariette Awad, Assistant Professor
Electrical and Computer Engineering


Advisor

Dr. Nabil Nassif, Professor
Mathematics


Member of Committee

Dr. Mazen Saghir, Associate Professor
Computer Engineering


c/c
Member of Committee

Date of thesis defense: May 9, 2014

AMERICAN UNIVERSITY OF BEIRUT

THESIS, DISSERTATION, PROJECT RELEASE FORM

Student Name: H M A Y A L W E V O R K P A R T A M I A N
Last First Middle

Master's Thesis Master's Project Doctoral Dissertation

I authorize the American University of Beirut to: (a) reproduce hard or electronic copies of my thesis, dissertation, or project; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes.

I authorize the American University of Beirut, **three years after the date of submitting my thesis, dissertation, or project**, to: (a) reproduce hard or electronic copies of it; (b) include such copies in the archives and digital repositories of the University; and (c) make freely available such copies to third parties for research or educational purposes.


Signature

May 16, 2014
Date

ACKNOWLEDGMENTS

I take this opportunity to thank everyone who supported me in finalizing this work. First, I thank Dr. Mariette Awad for her continuous support, patience, and aid to carry out this thesis. I thank my supervisors, Dr. Nabil Nassif and the head of Computational Science department, Dr. Mazen Al Ghoul, and Dr. Mazen Al Saghir for their continuous support. I would like to thank also Dr. MagedaSharafeddin for her continuous support and the help she provided. I also thank all the RASSD project team, Qatar Foundation and AUB for their support and the opportunities of research, education, and the funding they provided.

I also want to thank my amazing mother, ArdemisPartamian, my brother Haig and sister in law Flora, my nephew Kevork, my lovely sister Maral and Family, and my aunt VrejouhiKejakoushian and family, and my two lovely aunts Nouritsa and Jacqueline Partamian and also my uncle AntranikPartamian and family. Finally, I want to thank my great friends who aided me by just being there during the fulfillment of this project especially Jack Aroyan and Daisy Al Jurdi, Nareg Partamian, Vahe Shahbazian, Raffi and Shant Pashabedian, YervantToramian, Abraham and Nayiri Fesdekjian, Taline Chitilian, Serly Bajakezian, Apig Fesdekjian and Dr. Pegor Aynajian and Tamar Boyajian.

Finally, I dedicate this work to my deceased father, KevorkPartamian and Hrag Bekarian.

ABSTRACT

Oil and gas exploration involves different complex and costly procedures. Specially designed vehicles (trucks or ships) send sound waves and collect their reflections using a set of predesigned geometrically distributed sensors. Further analysis is performed to extract the different seismic attributes which help identify the different lithological formations such as oil and gas reservoirs. The analysis also helps identify suitable drilling sites and estimate oil or gas quantity for business men and economists to assess the drilling risks and costs which can reach up to 1Billion dollars. In short, seismic data analysis is a distributed big data analysis by excellence: it involves many complex and computationally expensive operations from massive data acquisition, to data processing and data analysis.

In this thesis, seismic data acquisition, processing and analysis are described to highlight the complexity of the problem. The overall seismic data processing and analysis flow are migrated into a distributed design that uses the Map/Reduce Paradigm. A sample seismic texture analysis is carried out to identify target locations in an oil bearing site where slices of a 3D seismic block data are processed separately to extract window samples and their corresponding Haralick attributes using the Grey Level Co-occurrence Matrix (GLCM). We propose the Barricaded Boundary Minority Oversampling Method (BBMO) which is based on a modification of the least square support vector machine (LS-SVM) since it can be easily distributed due to its equivalent incremental form. BBMO oversamples the minority samples at the boundary in the direction of its closest majority samples to fix the problem of data imbalance caused by the fact that oil bearing sites in a specific field are usually less than the non-bearing sites resulting in imbalance in the seismic exploration data. All operations are described and profiled to find the computationally most expensive in our proposed framework. Experimental results on BBMO performance and computational improvements when multithreading and distributed accelerated computing are employed, motivate follow on work.

TABLE OF CONTENTS

ACKNOWLEDGMENTS	V
ABSTRACT.....	VI
TABLE OF CONTENTS.....	VII
TABLE OF FIGURES.....	X
LIST OF TABLES.....	XIII
1. INTRODUCTION	1
1.1 Thesis Objective	1
1.2 Thesis Organization.....	2
2.METHODS OF SEISMIC DATA ACQUISITION, PROCESSING, AND THEIR ANALYSIS.....	4
2.1 Introduction	4
2.2 Seismic Data Acquisition and Processing	5
2.2.1 <i>Seismic Data Acquisition</i>	6
2.2.2 <i>Seismic Data Processing</i>	11
2.2.3 <i>Seismic Attributes</i>	17
2.3 Seismic Data Analysis.....	28
2.3.1 <i>Support Vector Machines</i>	29
2.3.2 <i>Least Squares Support Vector Machines</i>	33
2.3.3 <i>Distributed SVMs</i>	34
2.4 Distributed Platforms.....	38
2.5 Literature Review on Imbalanced Classification and Other Related Work	41
3. COMPUTATIONAL DESIGNS, FLOWS, ALGORITHMS, AND ANALYSIS.....	45

3.1 Introduction	45
3.2 Seismic Data Processing and Analysis Workflows	46
3.2.1 <i>Sequential Flow of Seismic Data Interpretation</i>	46
3.2.2 <i>Distributed Seismic Data Processing Workflows</i>	49
3.3 Seismic Texture Analysis	52
3.4 Seismic Texture Analysis Pseudocode and Complexity Study	56
3.5 Map/Reduce Based Design of Seismic Texture Analysis	59
3.6 Barricaded Boundary Minority Oversampling Method.....	63
3.6.1 <i>Nomenclature</i>	64
3.6.2 <i>BBMO formulation</i>	64
3.6.3 <i>BBMO and LS-SVM</i>	71
3.6.4 <i>Complexity Analysis</i>	75
4.EXPERIMENTAL RESULTS	76
4.1 Introduction	76
4.2 Experiments with BBMO	77
4.2.1 <i>Performance Metrics</i>	77
4.2.2 <i>Imbalanced Data Description</i>	78
4.2.3 <i>Experimental Results using BBMO</i>	79
4.3 Experiments on Seismic Textures	83
4.3.1 <i>Performance of the Classifiers on the Seismic Data</i>	85
4.3.2 <i>Volume Estimation and Visualization</i>	86
4.4 Profiling of the Seismic Data Analysis Algorithm	87
4.5 Experimental Results of the Seismic Application on Distributed Matlab ..	90
4.6 Seismic Analysis on RASSD with Acceleration	92

5.CONCLUSION AND FUTURE WORK	94
REFERENCES	96

TABLE OF FIGURES

Figure 1 Common types of traps	7
Figure 2 The seismic reflection method	8
Figure 3 Marine seismic data acquisition	9
Figure 4 2D and 3D seismic data acquisition	10
Figure 5 Seismic data processing	12
Figure 6 De-multiplexing seismic data	12
Figure 7 From seismic traces to seismic sections	13
Figure 8 De-convolution increases temporal resolution of seismic readings	14
Figure 9 Common midpoint (CMP) sorting	15
Figure 10 Velocity model	15
Figure 11 Normal Move-Out (NMO) correction	16
Figure 12 Migrated Image	17
Figure 13 Different attributes computed using seismic sections	21
Figure 14 Angular relationship between two neighboring cells in an image	23
Figure 15 Calculating the GLCM matrix of a 4x4 sample image	24
Figure 16 Windowing of images	28
Figure 17 Basic flow of seismic multi-attribute analysis	29
Figure 18 Learning the separating hyperplane of SVMs	30
Figure 19 Linearly separable and non-separable cases	31
Figure 20 Non-linear boundaries are transformed into a linear one using kernel method	32
Figure 21 Distributed LS-SVM	34
Figure 22 The cascaded SVM	37
Figure 23 The Hadoop Map/Reduce architecture	39
Figure 24 Overview of the general system architecture of the RASSD	41

Figure 25 Overview of the general system flow of seismic data processing	46
Figure 26 Attribute Extraction Stage	47
Figure 27 Classification stage	48
Figure 28 Map/Reduce input and output	50
Figure 29 Map Reduce based architecture of the preprocessing stage (PPS) and the attribute extraction stage (AES)	50
Figure 30 Map/Reduce based architecture of seismic attribute data processing (AMP) and the learning and prediction stages (LPS).	51
Figure 31 Seismic slices from the 2D volume taken as images	53
Figure 32 Texture attribute computation flow	54
Figure 33 Labeling strategy	54
Figure 34 Labeling examples	55
Figure 35 Learning and prediction flow using texture attributes	56
Figure 36 Distributed Map/Reduce design of the learning phase	60
Figure 37 Distributed Map/Reduce design of the prediction phase	62
Figure 38 Volume estimation and visualization	63
Figure 39 Illustrative extraction of the boundary samples of the linearly separable case and the formation of the barricade Z by calculating the weighted means of all the boundary samples	65
Figure 40 BBMO-1 pushes the hyper-plane away from the minority by adding the Barricade	68
Figure 41 BBMO-2 oversamples in the direction of the closest majority	69
Figure 42 The way SMOTE modifies the minority class distribution when synthetic data are added	70
Figure 43 The way BBMO-2 modifies the minority class distribution when synthetic data are added	70
Figure 44 Preparing the BBMO matrices to compute the weighted means implicitly using LS-SVM	74
Figure 45 The confusion matrix and the evaluation matrix	77

Figure 46 Average error in performance metrics when LS-SVM is compared with SVM with and without BBMO-2	81
Figure 47 Metric value variation when number of synthetic data is varied	82
Figure 48 Types of textures in a seismic slice	84
Figure 49 Seismic texture attribute values at the considered windows in figure 48	84
Figure 50 The energy attribute computed in 3D fashion using OpenDtect Software	85
Figure 51 Performance of LS-SVM with BBMO	86
Figure 52 3D seismic prediction and volume reconstruction	87
Figure 53 Profiling of the learning phase	88
Figure 54 Profiling of the computationally most expensive operation of the learning phase	89
Figure 55 Profiling of the prediction phase	89
Figure 56 Computational cost of the prediction phase as data size increase: serial versus multithreaded	91
Figure 57 Speed-up due to multithreading	91
Figure 58 Haralick and GLCM operations' speedups	92

LIST OF TABLES

1.	Types of marine seismic surveys.....	10
2.	Computational complexity of seismic texture analysis.....	58
3.	Description of datasets.....	78
4.	Performance of SVMs compared with BBMO-1.....	79
5.	Performance of SVMs compared with BBMO-2.....	80
6.	Comparison of BBMO-2 against SMOTE.....	82

CHAPTER I

INTRODUCTION

1.1 Thesis Objective

Complex and large-scale data analysis have captured the attention of scientists and researchers to find solutions to the high computational costs and reduce their processing times. In particular, large-scale seismic data analysis involves many complex operations each of which has diversely expanded and researched to include different methods increasing the computational cost of the whole application. Different problems of seismic data analysis are addressed and solved. The main objectives of this thesis are:

1. An overview of marine seismic data acquisition, processing and analysis is presented highlighting different methods employed during the three different stages.
2. An overview of seismic texture analysis using the Haralick attributes that employ the Gray Level Co-occurrence matrices to extract four features that can aid in the identification of oil bearing sites
3. Our proposed Barricaded Boundary Minority oversampling technique to remove the bias of LS-SVM with imbalanced datasets
4. Description of the methods used during learning and prediction of oil bearing sites in 3D seismic data

5. Extension of the method into a distributed Map/Reduce model after profiling the different tasks in the algorithm in different distributed platforms

1.2 Thesis Organization

The thesis is organized as follows

Chapter 2 describes an overview of seismic data analysis. The raw data captured from the sensors undergoes many different processing techniques until the data is ready to be migrated into the 3D volume. Marine seismic method is described and all the processing steps are briefly described. Seismic texture analysis is described in detail. Support vector machines which are the classifiers used in this thesis along with their distributed forms are also described in detail. An overview of the map reduce framework and the RASSD frameworks is provided. Finally, a literature review on imbalanced datasets is discussed.

Chapter 3 discusses the serial and distributed forms of the different flows in the method. The overall seismic method described in chapter 2 is imported into a distributed design that uses the Map/Reduce paradigm. An example flow that includes extraction and analysis of seismic texture attributes is presented and analyzed based on the design proposed. Finally, our proposed Barricaded Boundary Minority Oversampling technique which oversamples the minority at the boundary in the direction of the closest majority samples to remove the bias of LS-SVM. A detailed formulation and analysis is presented.

Chapter 4 gives the experimental setup, the metrics used, and the different experiments performed on the different datasets. First, experimental results of the BBMO are presented and analyzed. Next, seismic texture analysis is performed serially, in a distributed fashion, with multithreading, and with map reduce design over RASSD. Computational speedups and accuracy results are reported.

Chapter 5 provides a conclusion and highlights the possible extensions of the discussed topics.

CHAPTER II

METHODS OF SEISMIC DATA ACQUISITION, PROCESSING, AND THEIR ANALYSIS

2.1 Introduction

The exploration of oil and gas involves different complex and costly procedures that encompass the joint efforts of geologists, geophysicists, engineers and businessmen. Specially designed vehicles (trucks or ships) send sound waves and collect their reflections using a set of predesigned geometrically distributed sensors. Further analysis is performed to extract the different seismic attributes which help identify the different lithological formations such as oil and gas reservoirs. The analysis also helps identify suitable drilling sites and estimate oil or gas quantity for business men and economists to assess the risks and the costs of drilling (up to 1Billion dollars [1]).

In addition to the complex processing steps involved in seismic analysis, the amount of data collected and involved during processing is huge (ranging between 100GB to few TB [2]) which makes the problem computationally too expensive especially in 4D analysis and monitoring where real-time analysis is crucial. On the other hand, many distributed methods and paradigms such as the Hadoop Map/Reduce have proved to reduce the computational costs of large data analysis by parallelizing the tasks among the different nodes in the cloud.

The main purpose of this chapter is to describe the techniques and methods used during marine seismic data acquisition, data processing, and data interpretation. We discuss marine seismic data acquisition methods highlighting the difference between

2D, 3D, and 4D seismic data. Next, seismic data processing is discussed where the data is reconditioned first and migrated later to form the 3D seismic volumetric data. An overview of different data interpretation techniques that extract the needed features or attributes from the seismic data is presented. Among the seismic attributes and their different classifications, we discuss in detail texture attributes which employ the grey co-occurrence matrix (GLCM) to compute the Haralick texture attributes. An outline of Support Vector Machines (SVM) and Least Squares Support Vector Machines (LS-SVM) is provided along with their distributed forms. The Hadoop Map Reduce and the RASSD platforms are briefly reviewed. Finally, imbalanced datasets and the different techniques used to overcome the problem of imbalance are reviewed.

The following nomenclature is used for this section:

θ :GLCMrelationship angle	d_{i1} :classifier matrix-matrix multiplication term
$C_{\theta,n}$:GLCM horizontal matrix ($\theta=0$)	d_{i2} :classifier matrix-vector multiplication term
V_{ij} :symmetric GLCM elements	L_i :label matrix of each window
P_{ij} :normalized GLCM elements	ω : hyperplane weight vector
N_g :number gray tones	b :hyperplane bias
X_i : matrix containing the window samples on node i	C :the penalty term
e :vector of ones	ξ :soft margin slack variables
D_i :diagonal matrix containing the labels of the windows on node i	I_0 :identity matrix of size $m+1$ whose last diagonal value is 0
E_i :adjusted sample matrix on node i	B :testing data matrix
	Y :predicted label
	α :Lagrange multipliers

2.2 Seismic Data Acquisition and Processing

In this section, an overview of the seismic method is represented. Oil trapped beneath the earth surface is first discovered using the seismic acquisition method where

acoustic waves are sent into the earth's crust and their response signals are gathered by special sensors. The collected data undergoes a series of processes to recondition the data and migrate it to form a 3D model of the explored site. The data is processed to extract different attributes which are in turn used for seismic event analysis such as oil bearing sites.

2.2.1 Seismic Data Acquisition

Oil and gas are fluids of organic origin that were cooked under high pressure and over millions of years underneath the earth's surface [3]. These hydrocarbon fluids are stored in porous rocks [4] forming an oil trap (geologic environments that allow for significant amounts of oil or natural gas to accumulate). Oil traps have two basic components: a permeable reservoir rock to hold the oil and natural gas, and a cap rock that is impermeable and traps the accumulated material. These traps exist under the earth's crust both on land and under the oceans and seas. Naturally and over millions of years, different types of traps have formed as shown in Figure 1.

Many techniques have been employed to discover and characterize these traps, the most popular of which is the seismic method. Whether conducted on land or in marine environments, the method is the same except for some details and the equipment used during acquisition. In this paper, the marine seismic method is described and analyzed.

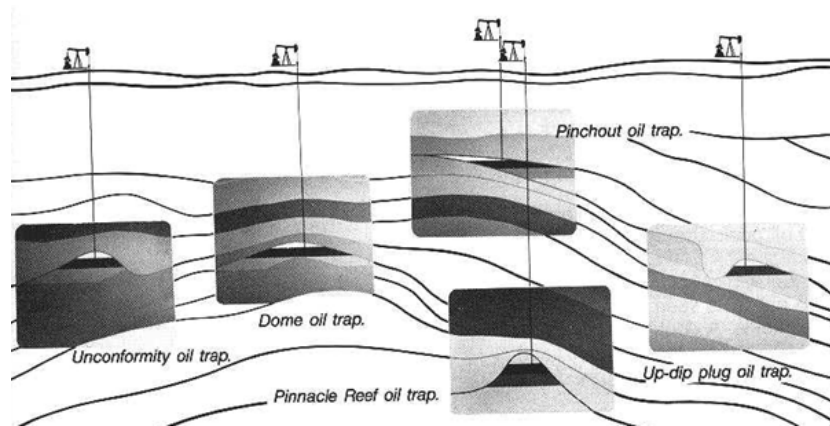


Figure 1 Common types of traps [5]

The marine seismic method employs high energy, low frequency sound waves that can penetrate more than 6km below the sea floor using an acoustic sound source such as Vibroseis, hand gun, or dynamite. The generated acoustic waves travel through the various layers underneath the surface and bounce back to the receivers, geophones or hydrophones, which measure the strength and return time of each wave [6]. The sound wave propagates into the different layers governed by the laws of reflection and refraction and return to the sensors in three forms: compressional wave (P-wave), shear wave (S-wave) and the ghost wave as shown in Figure 2.

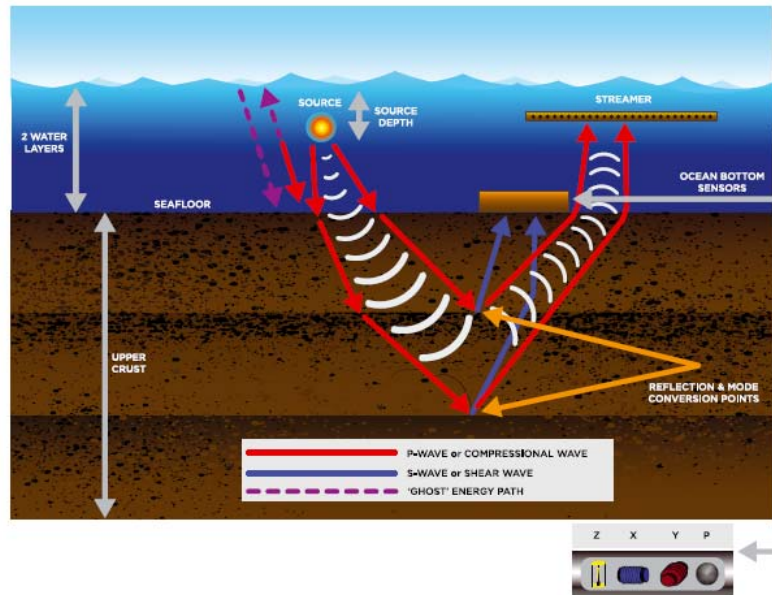


Figure 2The seismic reflection method [7]

The marine seismic method employs one or two sets of underwater equipment directly connected behind the marine seismic vessel (Figure 3). One set to generate sound waves and another composed of one or several long cables called streamers, each containing several hundred evenly spaced receivers that record the reflected signals [7].

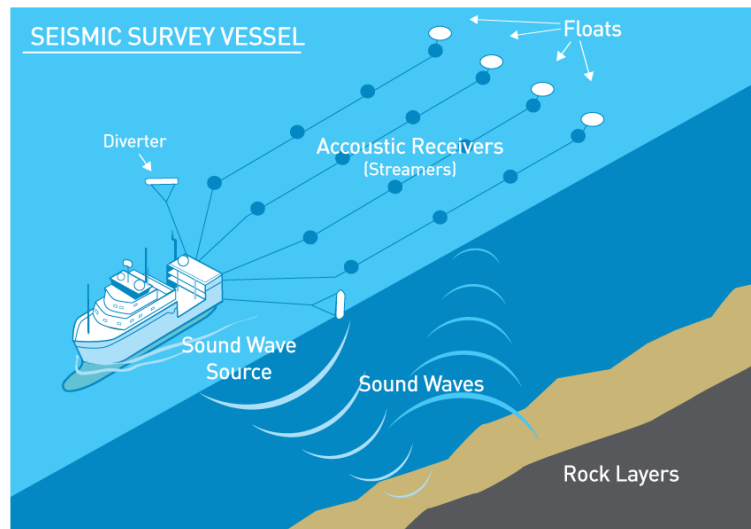


Figure 3 Marine seismic data acquisition [8]

Based on the set-up used, seismic surveys may be two dimensional (2D), three-dimensional (3D), or 4 dimensional (4D). 2-D surveys use one sound source and one set of receivers. These surveys are usually conducted along a grid with parallel lines that are up to five kilometers apart. The acquired survey data can then be analyzed to produce a set of 2D images. With 2D acquisition, a general picture of the geological characteristics of an area is produced indicating different types and sizes of structures present. To get more detailed information about the present geological features, 3-D surveys are conducted. Two or more sound sources and multiple sets of receivers (Figure 4) produce more accurate volumetric data per survey vessel sail line resulting in more accurate models of the exploration site.

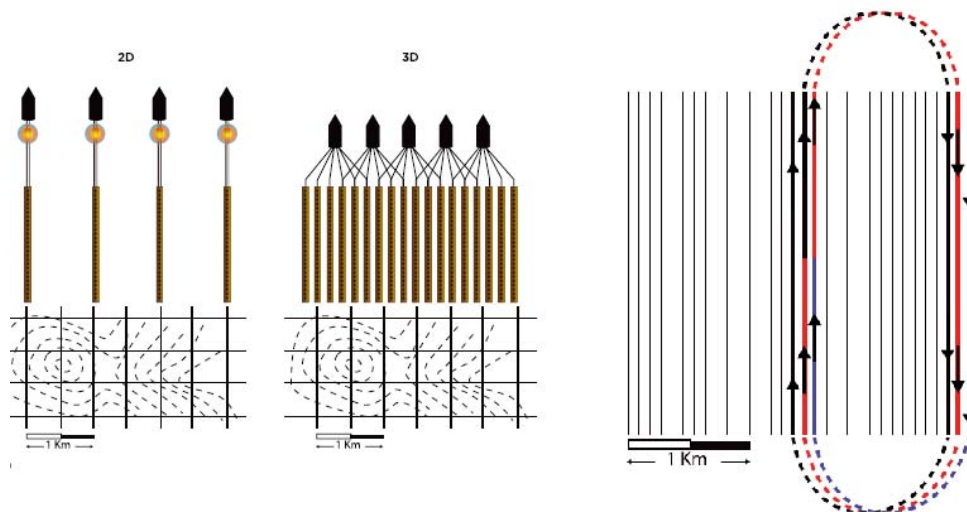


Figure 42Dand 3D seismic data acquisition [7]

3D surveys are typically acquired along a racetrack pattern (as shown in figure 4) and may take many months to complete. Powerful computers are required to process the large volume of the data and produce a 3D image of the subsurface. A 3D seismic data acquisition project for a 500 km² area takes about 8 months [9].

Table 1.Types of Marine Seismic Surveys

Survey/ Data Produced	Setup	Amount of Data	Time Taken to Process	Purpose
2D	-1 sound Source -1 streamer -Equidistant Sensors	Few Gb	Few Days	General View Inaccurate
3D	-Many Streamers -Sensors organized in a grid	100GB -1.5 TB	Few Months	Detailed View Accurate
4D	Underground Sensors	Few GB /TB per unit time	Pseudo Real Time	Accurate Monitoring Real Time

4D seismic surveys are 3D surveys carried out at different times over the producing life time of a field to assess how the hydrocarbons are moving through the reservoirs as production proceeds. In this case, the hydrophones are placed at the sea floor connected to the processing vessel/structure (z, y, x, and p in figure 2) [10]. The on-board high performance computers allow the monitoring of the oil sites in a pseudo-real time fashion. Table 1 compares the different methods in terms of time taken, equipment used, acquisition times and applications.

Higher resolution images can be obtained by using sparkers, boomers, or chirp profilers instead of hydrophones [7]. For more detailed imaging, wide azimuth imaging [10] is used especially in sites with complex geological environments. Other parameters such as the currents, tides, temperature, salinity of water, and weather condition, are also recorded to correct the seismic readings. The data saved in the SEG-Y format contains the raw and some processed data in multiplexed channel sequential order per recording cycle. The data are recorded at the same time at consecutive channels [11].

2.2.2 Seismic Data Processing

The raw seismic data cannot be directly used for analysis. Yilmaz [12] illustrated in detail the different processing techniques applied on seismic data (Figure 5). The data is reconditioned first to increase the signal to noise (SNR) ratio and later migrated to create a geographical model of the subsurface under study.

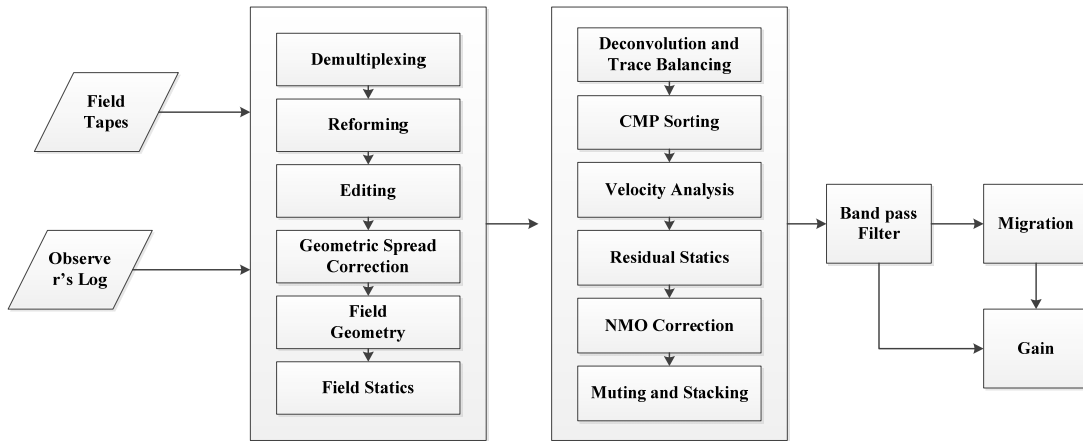


Figure 5 Seismic data processing [12]

The multiplexed data samples from a recording cycle are first de-multiplexed by storing the data samples in consecutive order in memory, saving all the samples in one channel followed by those of the next channel as shown in figure 6. This process can be done by matrix transposition.

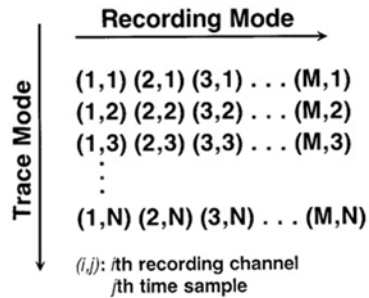


Figure 6 De-multiplexing seismic data [12]

Next, the data is edited to remove the dead traces, noise traces, switch polarity on reversed traces and cut unwanted signals. Gain recovery is also applied. This process is like turning up the volume to account for seismic attenuation. Automatic gain control

can also be used to equalize the amplitude along the trace. Finally, field geometry is incorporated with the seismic data. Coordinates of shot and receiver gathers are added to the headers of the seismic data. Seismic signals can be considered as a sampled signal wave.

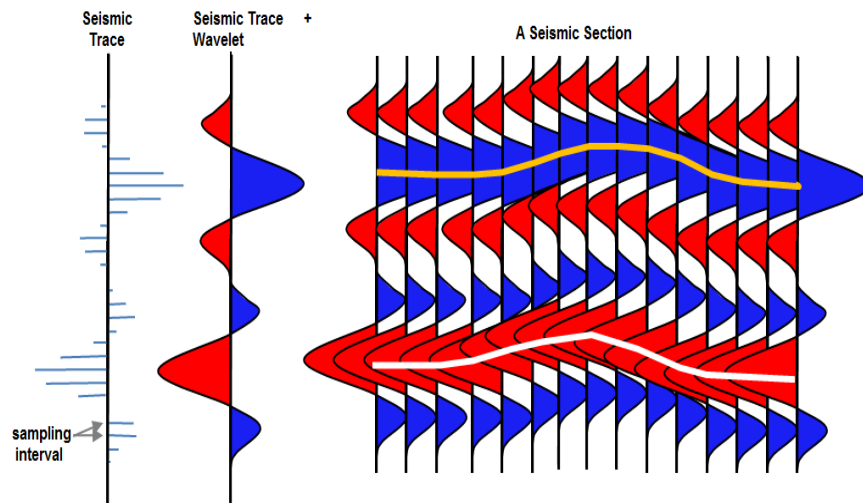


Figure 7 From seismic traces to seismic sections [13]

The waves are incorporated with a wavelet to produce a continuous wave that can be processed (Figure 7). When the waves are assembled based on their correct geographical locations, a seismic section is formed showing the lithological structure of the subsurface. But first, the wave undergoes de-convolution to improve the SNR further. Seismic de-convolution improves the temporal resolution of the data by compressing the wavelet of the seismic trace to a spike using a wiener filter. This technique broadens the spectrum of seismic data to obtain more high-frequency energy

in the signal. Figure 8 (right) shows more detail due to de-convolution when compared to the original image (Figure 8 (left)).

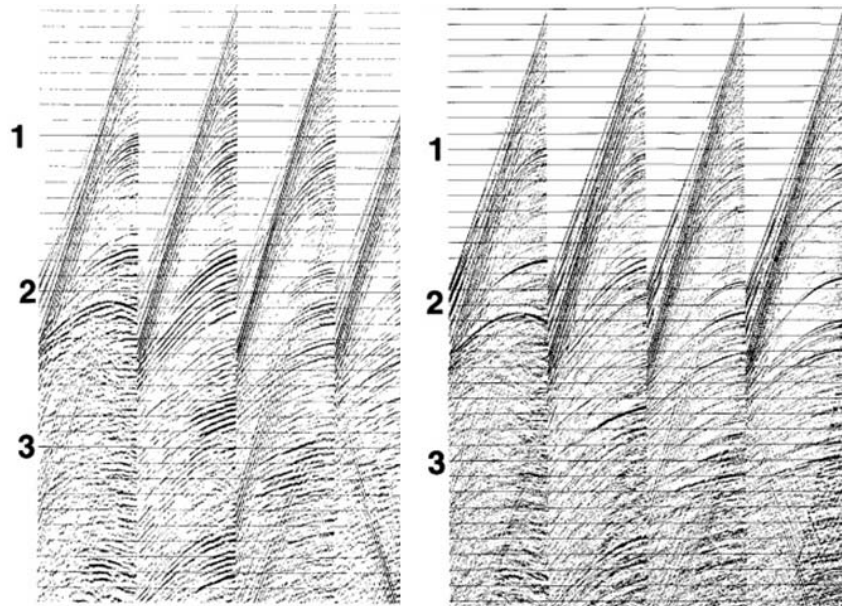


Figure 8 De-convolution increases temporal resolution of seismic readings [12]

Since both signal and noise are boosted, data is also filtered using a wide band filter. The data is then transformed from shot-receiver coordinates to midpoint-offset coordinates by using the field geometry information. With Common Midpoint Sorting (CMP) (Figure 9), each trace is assigned to the midpoint of the shot and receiver and all the ones with the common midpoint are grouped together. CMP stacking attenuates coherent noise such as multiples, guided waves, and ground roll.

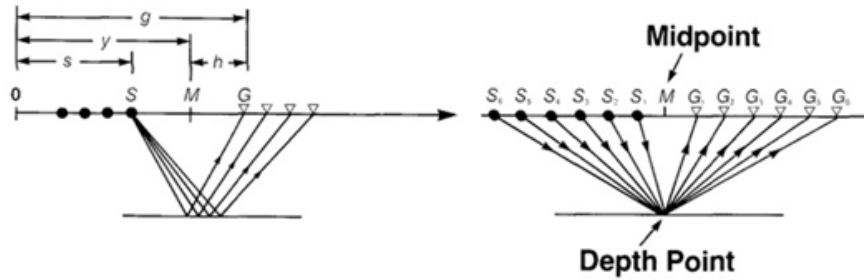


Figure 9 Common midpoint (CMP) sorting [12]

On the other hand, multi-fold coverage with non-zero offset recording yields velocity information about the subsurface. Velocity analysis is a crucial phase in most imaging projects since it aids during the migration to the depth domain. A study performed on selected CMP gathers or groups of gathers yields the velocity spectrum that represents hyperbolic trajectories governed by velocity, offset, and travel-time.

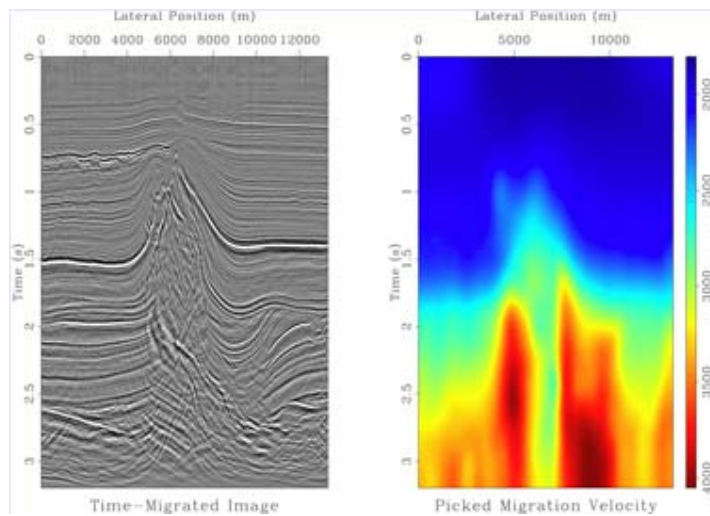


Figure 10 Velocity model [12]

These velocity models are then spatially interpolated across the entire profile to supply a velocity function for each CMP gather in the profile (Figure 10). Velocity model is also used for normal move-out correction to flatten the events across the offset range. The resulting distortions are muted before stacking. Residual static correction is applied on the NMO corrected (Figure 11) stacked data to improve stacking quality. Predictive de-convolution is used to suppress reverberations and whiten the spectrum. Time variant band pass filtering is applied to suppress noisy frequency bands and finally a gain is applied to bring up weak reflections.

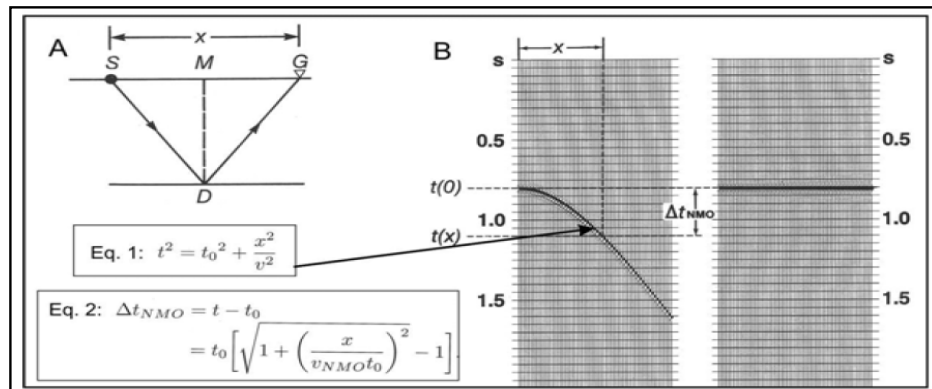


Figure 11 Normal Move-Out (NMO) correction [12]

Dipping events in the seismic data are mapped to their subsurface position by migration of the stacked section using the medium velocity to produce a seismic 2D image or slice as shown in figure 12. Migration also collapses reflections. There are two types of Migration: pre-stack time migration and pre-stack depth migration. Kirchhoff time migration (PKTM) is one of the most popular migration algorithms [14].

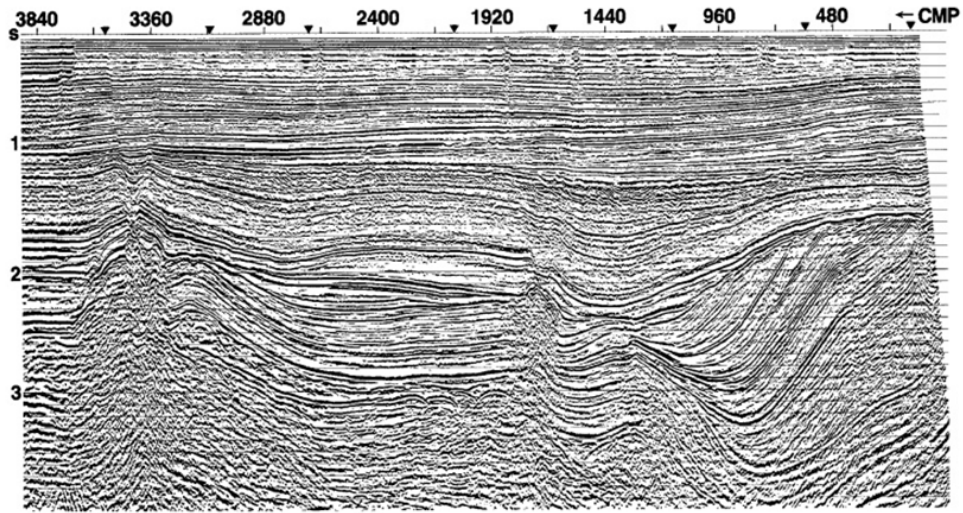


Figure 12 Migrated Image [12]

2.2.3 Seismic Attributes

From the processed seismic data, a plethora of seismic attributes can be extracted. Seismic attributes are quantified specific data characteristics which represent subsets of total information and can be computed from pre-stack or from post-stack data, before or after time migration. Ben Summerfield highlighted that it was possible to predict fluid content, porosity and faces changes from reflection character changes such as bright spots [15] that became popular in the 1970's [16][17]. Taner and Sheriff introduced five attributes: instantaneous amplitude, instantaneous phase, instantaneous polarity, instantaneous frequency, and weighted average frequency [18]. New geologically non-significant attributes such as zero-crossing frequency, average amplitude, dominant frequency and many others, revealed meaningful patterns in the seismic data. Several studies related complex trace seismic attributes to Fourier spectral averages, which yielded clues to wavelet properties and led to "response attributes" [19].

Ostrander used pre-stack attributes and proved them to be efficient direct hydrocarbon indicators [20]. In the mid 1990's, 3D seismic attributes became popular which helped bring new advances into the attribute analysis field [21]. Other multi-dimensional attributes soon followed, such as parallelism and divergence [22][23]. Seismic Inversion [24-27] and Attribute versus Offset (AVO) [28-30] (which is based on the Zoeppritz equations [31]) are two other techniques commonly used in seismic data analysis as well. They employ both pre-stack and post-stack data to compute diverse attributes such as the Acoustic Impedance (AI), and the Elastic Impedance (EI) [32]. Simultaneous Pre-stack Inversion uses the pre-stack CMP gathers to compute impedance and density [33]. Seismic and well-log data have to be tied together by using interpreted seismic horizons before inversion [34]. 3D versions of dip and azimuth were also defined and studied in [35].

2.2.3.1 Seismic Attribute Classification

Hundreds of seismic attributes have been created and are classified in many ways [36-38]: attributes derived from Fourier analysis, complex trace analysis, time-frequency analysis, wavelet transforms, principal components, AVO attributes, and texture attributes to name a few.

Pre-Stack Attributes

Azimuth and offset related information can be derived from the image gathers traces or the CMP. Although computing these attributes is usually computationally expensive, they can help understand fluid content and fracture orientations. AVO attributes and velocity are examples of pre-stack attributes.

Post-Stack Attributes

Post-stack attributes are the ones computed after stacking the seismic data which is an averaging process to eliminate offset and azimuth information. The migrated data will still maintain the time relationships and all the temporal variables such as frequency. Such attributes are used to observe large amounts of data in initial explorations.

Attributes may be further classified by their computational characteristics:

Instantaneous Attributes

Instantaneous attributes are calculated sample by sample, and represent instantaneous variations of various parameters. Complex traces may be used to compute instantaneous values of attributes such as trace envelope, its derivatives, frequency and phase.

Wavelet Attributes

Wavelet attributes include the instantaneous attributes that are calculated at the peak of the trace envelope and have a direct relationship to the Fourier transform of the wavelet. For instance, instantaneous frequency at the peak of the envelope is equal to the mean frequency of the wavelet amplitude spectrum. Instantaneous phase corresponds to the intercept phase of the wavelet. This attribute is also called the “response attribute”.

Physical Attributes

Physical attributes relate to physical qualities and quantities. The magnitude of the trace envelope is proportional to the acoustic impedance contrast. Frequencies directly relate to bed thickness, wave scattering and absorption. Instantaneous and

average velocities directly relate to rock properties. These attributes are classically used for lithological classification as well as in reservoir characterization.

Geometrical Attributes

Geometrical attributes define the spatial and temporal relationship of all other attributes. For instance, lateral continuity using semblance is a good indicator of bedding similarity and discontinuity. Because they define event characteristics and their spatial relationships, geometrical attributes are also used for the recognition of depositional patterns, and the lithology.

In summary, variation of amplitude of post and pre-stack seismic data are valuable for hydrocarbon investigation, especially in relation to gas reservoirs.

Some Basic Attribute Characteristics

Let us consider few examples of attributes and their characteristics. The envelope represents the instantaneous energy of the signal which is proportional in its magnitude to the reflection coefficient. The Trace Envelope is a physical attribute and it can be used as an effective discriminator for bright spots, sequence boundaries, acoustic impedance contrast, gas accumulation, and others. The First Derivative of the Envelope with respect to time represents the variation of the energy of the reflected events. This attribute is also a physical attribute and it can be used to detect possible fracturing and absorption properties. The Second Derivative of the Envelope gives a measure of the sharpness of the peak of the envelope. It can identify all reflecting interfaces within the seismic bandwidth.

The instantaneous phase attribute is also a physical attribute and can be effectively used as a discriminator for geometrical shape classifications. Instantaneous

phase is a good indicator of lateral continuity, can be used to compute the phase velocity, and can produce detailed visualization of stratigraphic elements.

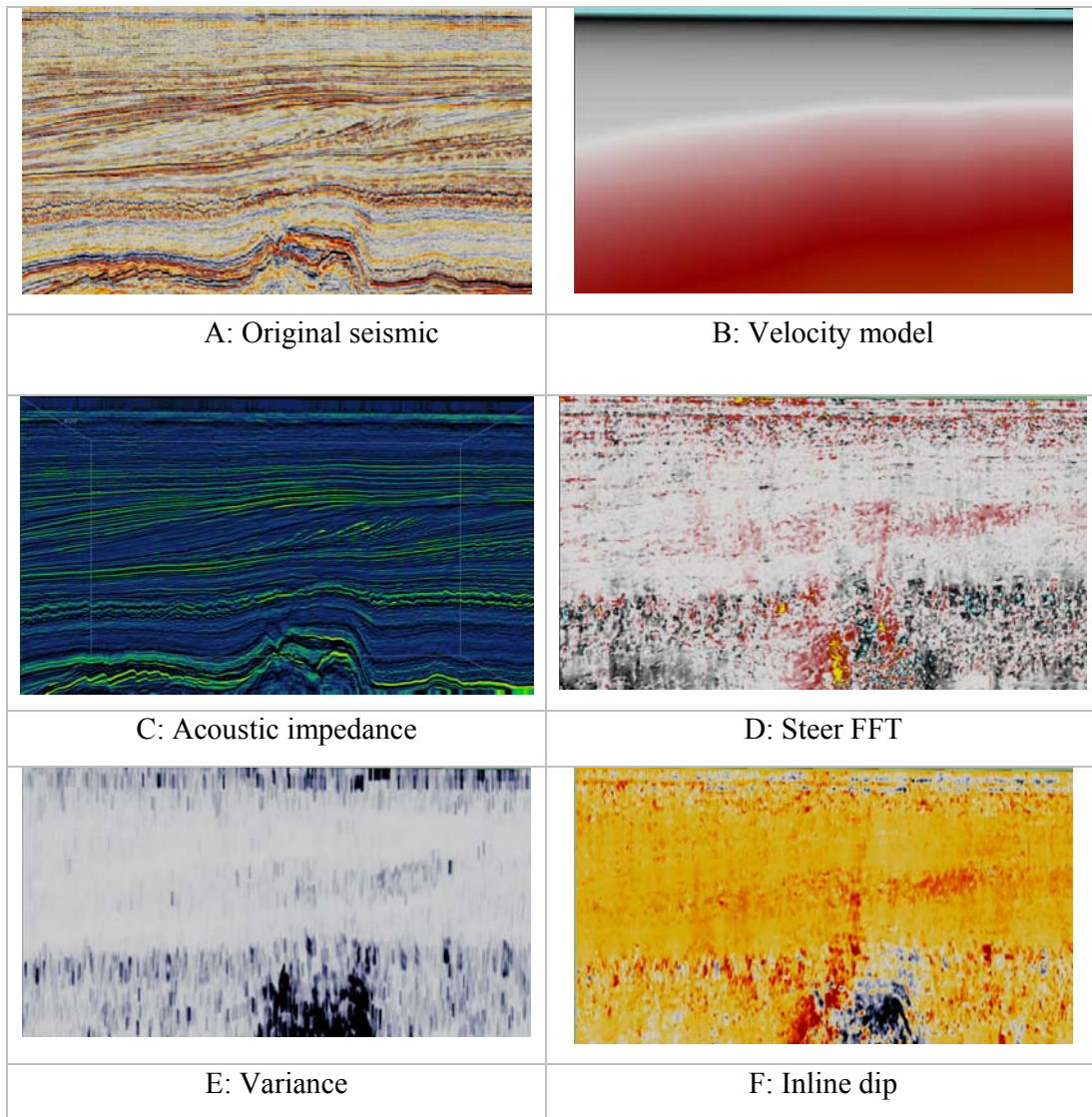


Figure 13 Different attributes computed using seismic sections [141]

Instantaneous frequency is the time derivative of phase, which relates to the centroid of the power spectrum of the seismic wavelet, responds to both wave propagation effects and depositional characteristics. It is a physical attribute and can be used as a Hydrocarbon indicator by low frequency anomaly. It can also be used to identify fracture zones at the lower frequency zones. Instantaneous frequency can also be used to indicate bed thickness. Figure 13 illustrates examples of seismic attributes represented as 2D images.

2.2.3.2 Seismic Texture Analysis

Texture is another important characteristic that can help in identifying certain patterns of interest within an image and can be applicable on a wide variety of applications that deal with images. Haralick et al. developed 14 features for texture [39]. However, defining or selecting these features is dependent on the application itself. Textural features are computed based on the tonal variation within a band or a window. Varying shades of grey in an image can be used to calculate texture features. These textures can be fine, coarse or smooth, irregular or lineated.

Grey-Level Co-occurrence matrices (GLCM) is the basis of Haralick feature attributes which considers the relation between two pixels at a time, called the reference and the neighbor pixel. Any image, which can be defined as a two dimensional array or a matrix, can be quantized into a finite number of grey levels N_g . The grey-level co-occurrence matrix can be defined as the probability P_{ij} where two neighboring pixels separated by a certain distance occur in an image, one with grey tone i and one with grey tone j . The GLCM is also dependent on the angular relationship between the two

cells. Neighboring cells can have an angular relationship θ equal to 0° , 45° , 90° and 135° as shown in figure 14.

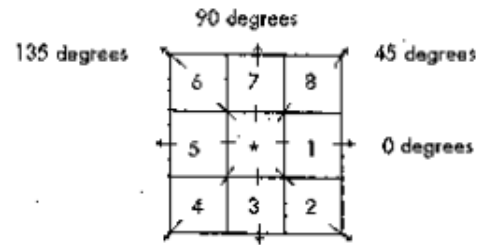


Figure 14 Angular relationship between two neighboring cells in an image [39]

After setting the distance and the angular relationship, the GLCM can be computed as follows. First, the original image of dimension $n \times m$ is quantized into a number of grey levels N_g , usually 8, 16, or 32. The GLCM will have a dimension $N_g \times N_g$. Based on the distance and the orientation, the number of co-occurrences of different grey levels in the image is computed. The symmetric of the computed matrix is added to it and then averaged to obtain a normalized symmetric square matrix. $V_{i,j}$ being the sum of the GLCM and its symmetric, the probabilities $P_{i,j}$ are computed by normalization using equation (1) :

$$P_{i,j} = \frac{V_{i,j}}{\sum_{i,j=1}^{N_g} V_{i,j}} \quad (1)$$

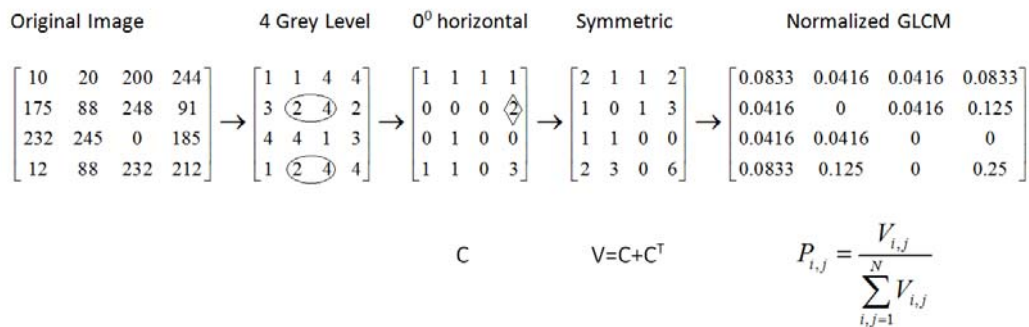


Figure 15 Calculating the GLCM matrix of a 4x4 sample image

As an example, consider figure 15. The 4x4 image is quantized into 4 grey tones (1, 2, 3, and 4). Next the matrix C is constructed which contains the relative frequencies of the different grey tone combinations within the image with a 0 degree angular relationship separated by a distance 1 which means the first neighbor to the right. As an example, the value “2” is adjacent to the value “4” two times thus C (2, 4) is set to 2. The GLCM matrix is constructed by adding the symmetric of the matrix C to itself and then normalized.

Haralick introduced fourteen statistical texture features which were generated from the GLCM in different directions and averaged. These features can be divided into three major groups: contrast group, orderliness and statistical some of which are defined and discussed.

Contrast Group

This group comprises the features that use weights related to the distance from the GLCM diagonal. All these features include the (i-j) term multiplied by the GLCM probability values.

A. **Contrast:** Contrast is a local grey level variation in the grey level co-occurrence matrix. It can be thought of as a linear dependency of grey levels of neighboring pixels. According to the equation of contrast, the weights increase exponentially. High contrast values are likely for heavy textures and low for smooth and soft textures.

$$Contrast = \sum_{i,j=1}^{N_g} P_{i,j} (i-j)^2$$

B. **Dissimilarity:** Dissimilarity is a measure that defines the variation of grey level pairs in an image. It is the closest to Contrast with a difference in the weight. Unlike contrast, the weights of the dissimilarity function increase linearly.

$$Dissimilarity = \sum_{i,j=1}^{N_g} P_{i,j} |i-j|$$

C. **Homogeneity:** Homogeneity measures the uniformity of the non-zero entries in the GLCM. Homogeneity weights values by the inverse of the Contrast weight, with weights decreasing exponentially away from the diagonal.

$$Homogeneity = \sum_{i,j=1}^{N_g} \frac{P_{i,j}}{1+(i-j)^2}$$

Orderliness

Orderliness indicates how regular the pixel values within a specific window are.

A. **Angular second moment (ASM) and energy:** ASM and Energy use each P_{ij} as a weight for itself. High values of ASM or Energy occur when the window is very orderly.

$$ASM = \sum_{i,j=1}^{N_g} P_{i,j}^2$$

$$Energy = \sqrt{ASM}$$

B. **Maximum Probability:** occur if one combination of pixels dominates the pixel pairs in the window.

$$Maximum\ probability = Max(P_{i,j})$$

C. **Entropy:** Energy is the opposite of entropy. Energy can be used to do useful work. In that sense it represents orderliness. Entropy in any system represents disorder, where in the case of texture analysis is a measure of its spatial disorder

$$Entropy = \sum_{i,j=1}^{N_g} P_{i,j} (-\ln P_{i,j})$$

Statistical

The third group of GLCM texture measures consists of statistics derived from the GLCM. Examples include

A. **The mean:** which represents the average GLCM value horizontally or vertically

$$p_x(i) = \sum_{j=1}^{N_g} P_{i,j} \quad \text{and} \quad \mu_x = \sum_{i=1}^{N_g} i p_x(i)$$

$$p_y(j) = \sum_{i=1}^{N_g} P_{i,j} \quad \text{and} \quad \mu_y = \sum_{j=1}^{N_g} j p_y(j)$$

μ_x and μ_y are the means

B. The variance: Variance is a measure of the dispersion of the values around the mean. It is similar to entropy.

$$Variance = \sum_{i=1}^{N_g} p_x(i)(i - \mu_x)^2$$

C. The correlation: The Correlation texture measures the linear dependency of grey levels on those of neighboring pixels

$$Correlation = \frac{\sum_{i,j=1}^{N_g} ijP_{i,j} - \mu_x\mu_y}{\sigma_x\sigma_y} \text{ where } \sigma_x \text{ and } \sigma_y \text{ are the standard deviations}$$

GLCM and Haralick were the basis of face recognition analysis [40]. An Oil spill detection technique using GLCM and texture analysis is described in [41] where pixels or remote images are classified into two classes; oil or non-oil. Other applications include the use of GLCM in edge enhanced images [42] and as texture analysis in seismic data [43] and in fabric surface [44]. Volume texture extraction for 3D visualizations and interpretation has also been studied [45]. Texture attribute analysis was also employed in reservoir prediction and characterization [46] and for detecting colon cancer cells [47].

2.2.3.3 The Edges of the Image

While computing the Haralick attributes, the edges of the image are not included in the computation since values assigned to a specific pixel are defined by the window centered at that pixel. The edge pixels of an image usually represent a very small fraction of total image pixels which make this problem minor [48]. As shown in figure 16, windowing of the edges of the image is not possible when square matrix of

odd dimension (“5” in the figure) is employed. Only the windows centered at the pixels in the yellow region will be processed.

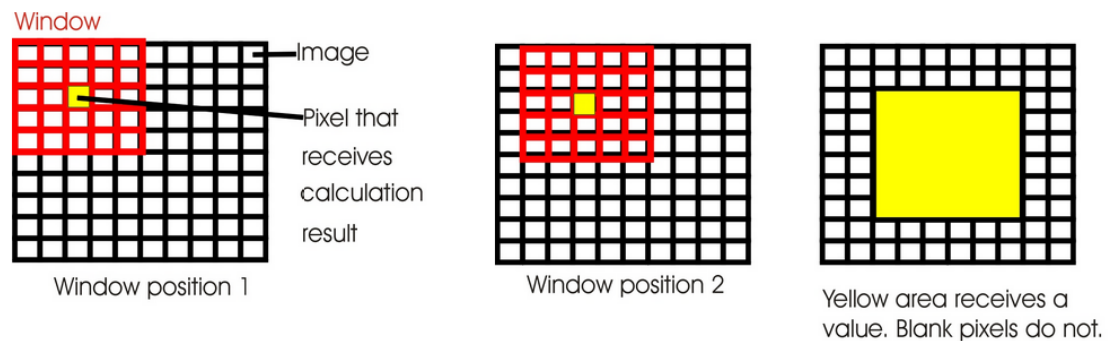


Figure 16Windowing of images [48]

2.3 Seismic Data Analysis

During analysis of seismic data, the selection of the proper set of attributes for a specific application is essential. Many attributes duplicate each other such as the amplitude attributes. Cross-plotting of seismic attributes can be employed to visually relate attributes and eliminate similar ones [49][50]. Principal component analysis (PCA) can also be used to determine the most effective seismic attributes [51]. Other attribute reduction methods employ rough set theory and support vector machines (SVM) [52]. Figure 17 depicts the general multi-attribute analysis of seismic data. Examples of applications include reservoir characterization [53], predicting reservoir properties from well logs using neural networks [54], and seismic analysis workflow for reservoir characterization in the vicinity of salt [55].

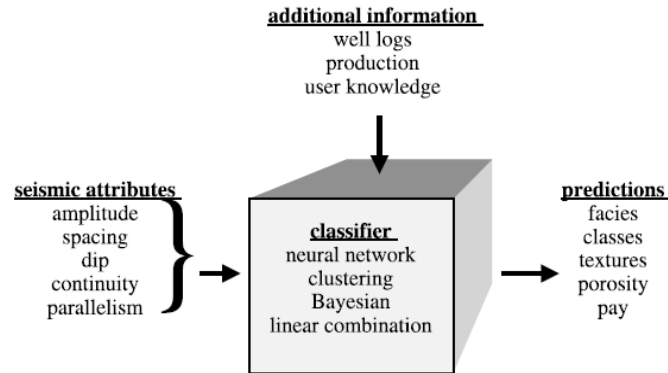


Figure 17 Basic flow of seismic multi-attribute analysis [22]

Hashemi et al. presented a classification application to detect gas chimneys using seismic attributes [56] and RDA method based on the set of popular attributes for the gas chimney problem introduced by Tingdahl et al. [57]. Other applications discuss the relationships between the different attributes [49]. Kliff analyzed 8 key hydrocarbon indicators retrieved from post-stack data [58].

2.3.1 Support Vector Machines

Support vector machines (SVMs) were first introduced by Vapnik which aims at finding the separating hyperplane that best separates the samples into two classes [59][60]. SVMs have many variations and different incremental and distributed forms which are described in what follows.

The original SVM is a constrained quadratic optimization problem, which produces a global and a unique solution and aims at maximizing the margin and minimizing the error of the misclassified samples. Given a training set $\{(x_i, y_i)\}_{i=1}^N$ where $x_i \in \mathbb{R}^n$, and assuming the pattern is linearly separable, there are two classes: $y_i = 1$ and

$y_i = -1$. The aim is to find the optimized hyperplane that separates the two classes (Figure 18).

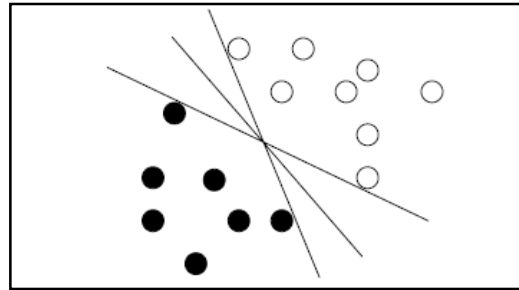


Figure 18 Learning the separating hyperplane of SVMs

Define by ω the hyperplane weight vector and b by its bias, the margin separating the two support vector hyper-planes is $\frac{2}{\|\omega\|}$ which needs to be maximized

and is equivalent to minimizing $\frac{\|\omega\|^2}{2} \Leftrightarrow \frac{\omega^T \omega}{2}$ under the minimized constraints

$y_i(\omega^T x + b) \geq 1$. The problem can be defined as

$$\begin{cases} \text{Minimize} & \frac{\omega^T \omega}{2} \\ \text{Subject to} & y_i(\omega^T x + b) \geq 1 \end{cases} \quad (2)$$

Let $\alpha_i \in \mathbb{R}$ be the Lagrange multipliers of the samples. The Lagrange equation of the optimization problem can be written as

$$L(w, b, \alpha) = \frac{\omega^T \omega}{2} - \sum_{i=1}^m \alpha_i \{y_i(\omega^T x + b) - 1\} \quad (3)$$

By finding the derivatives of $L(w, b, \alpha)$ with respect to ω and setting it equal to zero we get:

$$\omega = \sum_{i=1}^m \alpha_i y_i x_i \quad (4)$$

By finding the derivatives of $L(w,b,\alpha)$ with respect to b and setting it equal to zero we get:

$$\sum_{i=1}^m \alpha_i y_i = 0 \quad (5)$$

In general, in any optimization problem, the dual problem is easier to solve than the original problem. The dual problem can now be defined as:

$$\begin{cases} \text{Minimize} & L(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j x_i x_j \\ \text{Subject to} & \alpha_i \geq 0 \\ & \sum_{i=1}^m \alpha_i y_i = 0 \end{cases} \quad (6)$$

The samples with non-zero α_i are called the support vectors which represent the samples that decide the direction and the position of the separating hyperplane. In case the data is linearly inseparable, a relaxation technique is used to minimize the number of misclassified points (Figure 19).

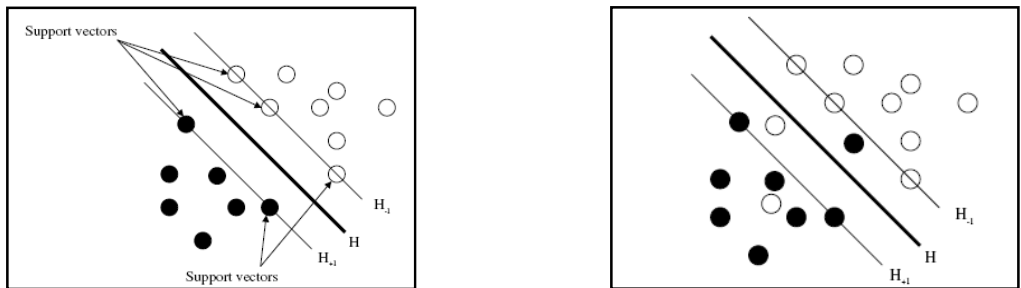


Figure 19 Linearly separable and non-separable cases [64]

In this case a soft margin slack variables ξ_i and the penalty parameter C (which controls the trade-off between the complexity and the number of misclassified points and needs

to be tuned) are introduced and the optimization problem becomes:

$$\left\{ \begin{array}{l} \text{Minimize} \quad \frac{\omega^T \omega}{2} + C \sum_{i=1}^N \xi_i \\ \text{Subject to} \quad y_i(\omega^T x + b) \geq 1 - \xi_i \\ \quad \quad \quad \xi_i \geq 0 \\ \quad \quad \quad \text{where } i = 1, \dots, N \end{array} \right. \quad (7)$$

The dual is almost similar to the previous formulation except for one of the constraints.

$$\left\{ \begin{array}{l} \text{Minimize} \quad L(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j x_i x_j \\ \text{Subject to} \quad \sum_{i=1}^m \alpha_i y_i = 0 \\ \quad \quad \quad 0 \leq \alpha_i \leq C \end{array} \right. \quad (8)$$

After determining the Lagrange multipliers, we compute the vector ω , then b , and then the equation of the hyper-plane. SVM is different from neural network in that it cannot have local minima hence producing a unique hyper-plane.

In case of non-linear boundaries as in figure 20, SVMs can employ several kinds of kernel functions such as linear, polynomial, radial basis functions (RBF) as long as they satisfy Mercer's condition. With the use of kernels, the data becomes linearly separable and SVM can be applied in feature space.

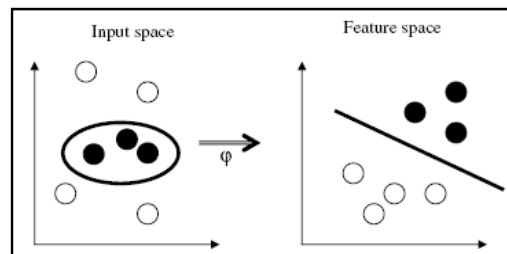


Figure 20 Non-linear boundaries are transformed into a linear one using kernel methods [64]

Support vector machines have been used in a plethora of applications such as text categorization [61], object recognition [62], computer graphics [63] and geotechnical engineering [64]. SVMs have been modified and restructured in different forms such as the sequential minimal optimization (SMO) [65] and least squares support vector machines (LS-SVM) [66].

2.3.2 Least Squares Support Vector Machines

The SVM formulation proposed by Vapnik can be solved using quadratic programming (QP) that is usually computationally expensive. Suykens and Vanderwalle proposed an alternative to SVM, the LS-SVM formulation by changing the error term in the objective function to a least square error function similar to ridge regression, and the inequality constraint was changed into an equality, which results in a system of linear equations instead of a QP problem [59]. X being the training samples, D being the diagonal matrix containing the labels of the training samples and e being a vector of ones, the optimization problem of the LS-SVM can be expressed in matrix form by

$$\begin{cases} \text{Minimize} & \frac{1}{2} \omega^T \omega + \frac{1}{2} C \xi^T \xi \\ \text{Subject to} & D(X^T \omega + be) = e - \xi \end{cases} \quad (9)$$

Due to the equality constraint, the above problem can be further reduced by substituting the error term into the objective function. The optimization function is reduced to an unconstrained one:

$$\begin{cases} \text{Minimize} & \frac{1}{2} \omega^T \omega + \frac{1}{2} (e - D(X^T \omega + be))^T (e - D(X^T \omega + be)) \end{cases} \quad (10)$$

Finding the partial derivatives with respect to ω and b and rearranging the terms, the problem becomes a system of linear equations.

$$[\omega_1 \ \omega_2 \dots \omega_n \ b]^T = \left(\frac{I_0}{C} + E^T E \right)^{-1} E^T D e \quad (11)$$

where $E = [X \ -e]$ and I_0 is a $(m+1)$ by $(m+1)$ diagonal matrix whose last entry is a zero and the others have diagonal entries equal to 1. Predicting the class of new samples is done by using

$$y_j = \text{sign}(wX_j - b) \quad (12)$$

2.3.3 Distributed SVMs

Do and Poulet proposed an incremental and distributed version of LS-SVM that solves the problem on smaller portions of the dataset at a time instead of loading the whole data into memory [67]. In their design, they consider the parts that can be incrementally solved. Suppose the data was split into k smaller blocks of data X_i which contains a certain amount of samples to be trained and suppose the diagonal matrix D is also divided into smaller diagonal matrices D_i as shown in figure 21.

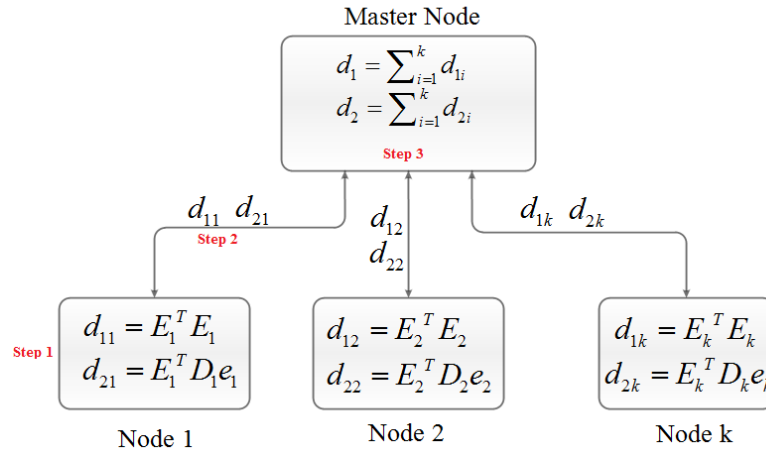


Figure 21 Distributed LS-SVM

We can write

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_k \end{bmatrix} \quad D = \begin{bmatrix} D_1 & 0 & 0 & \dots & 0 \\ 0 & D_2 & 0 & & 0 \\ \vdots & \vdots & & & 0 \\ 0 & 0 & 0 & & D_k \end{bmatrix} \quad \text{and } e = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_k \end{bmatrix} \quad (13)$$

Using linear algebra, the two terms $d_1 = E^T E$ and $d_2 = E^T D e$ of equation (11) can be computed in the following manner

$$d_1 = E^T D e = E_1^T D_1 e_1 + E_2^T D_2 e_2 + \dots + E_k^T D_k e_k = \sum_{i=1}^k E_i^T D_i e_i \quad (14)$$

$$d_2 = E^T E = E_1^T E_1 + E_2^T E_2 + \dots + E_k^T E_k = \sum_{i=1}^k E_i^T E_i \quad (15)$$

Therefore, the calculation of these two terms can be done block by block whose aggregated results can be used to find the final hyper-plane parameters using equation (11). The equation becomes

$$[\omega_1 \ \omega_2 \dots \omega_n \ b]^T = \left(\frac{I_0}{C} + \sum_{i=1}^k d_{1i} \right)^{-1} \sum_{i=1}^k d_{2i} \quad (16)$$

Do and Poulet's extended these concepts to a distributed version to run on distributed systems as shown in figure 21. Supposing there are K nodes in the distributed system, each node will have a portion of the data matrix X_i along with its portion of the diagonal matrix D_i . On each node i , $d_{1i} = E_i^T E_i$ and $d_{2i} = E_i^T D_i e_i$ are computed and the results are sent to the master node which aggregates them using equations (14) and (15). The master node finds the hyper-plane parameters ω and b using equation (16).

Distributed block minimization [68], the cascaded SVMs[69] , parallel SVM (PSVM) [70] and others [71] are examples of distributed SVMs. Distributed block minimization is a linear distributed SVM solver proposed by Pechyony et al. [68] which works in an environment in which the data is distributed over k nodes. A mapper function runs an updated version of SVM in the dual space which uses the difference of the parameter $\omega \Delta \omega = \omega_{t+1} - \omega_t$. The solver is run on each subset data at the nodes. The partial results are sent back to the master node where the reducer function gathers all the solutions from the different nodes and outputs a single global solution

$$\omega_{t+1} = \omega_t + \frac{1}{k} \sum_{i=1}^k \Delta \omega_i'$$

This solution is sent back to the nodes where a new solution is

computed. A threshold value controls the number of cycles of this operation. This algorithm was implemented on Map-Reduce/Hadoop on a dataset of 79M and took only 11 minutes to achieve an equivalent accuracy to the normal Liblinear SVM that took around 3 hours.

The parallel cascaded SVM [69] is another technique which works in a distributed fashion. It basically decomposes the problem into many sub-problems on which SVMs are applied and then it combines the resulting SVs two by two (or 3 by 3) as shown in figure 22. The SVM is applied again on these resulting sets to produce new SVs until 1 set remains and the final set of SVs is produced and the final hyper-plane parameters are computed.

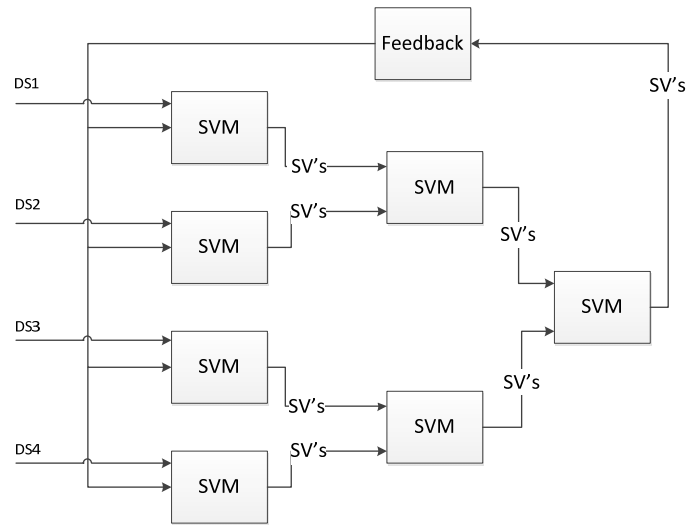


Figure 22The cascaded SVM

The last output can be sent back as feedback to update some of the initialized variables and the data will be scanned again in the cascaded form until a globally converging result is achieved. Filtering is performed by taking all elements that are interior to the region and discard the ones outside the region. In other words, a non-support vector in a subset has a higher chance of being a non-support vector of the whole problem, and that's why it can be eliminated. An attempt to speed-up the cascade SVM further was implemented using field programmable gate arrays (FPGA) as described in [72].

DominikBrugger has compared and analyzed in [73] different support vector machines where he investigated at identifying the computational most demanding portions of the algorithm and the parts of the quadratic program that can be parallelized. He concludes that all types of SVMs can be parallelized and linear and even super-linear speedups can be achieved. Different types of applications have been applied with

distributed SVMs such as in image annotation [74] pattern matching algorithms [75] and cloud based classification [76].

2.4 Distributed Platforms

In this section, we present a brief overview of the different distributed methods employed during our experiments: Hadoop Map/Reduce, Matlab, and RASSD.

The size of the digital universe has increased around ten times from 2006 to 2011 to reach 1.8 zettabytes (1 zettabyte-1 million terabytes) [77]. This is not surprising since a large part of the world uses social media, email services, and mobile applications. Hadoop was created by Doug Cutting who had already created Apache Lucene for text searching. The Google file system soon flourished in 2003 which had the capability to handle very large files around the web [78]. In 2004, Google introduced Map/Reduce to the world [79-82].

Map/Reduce is a programming model for processing and handling large datasets (Figure 23). The processes are divided into two phases: a mapper and a reducer. Each phase employs key-value pairs as input and output. The programmer specified the key value pairs used, the map function, and the reduce function. The map function processes data at the different nodes of the system to emit key-value pairs. The reducer merges all the intermediate results with the same key.

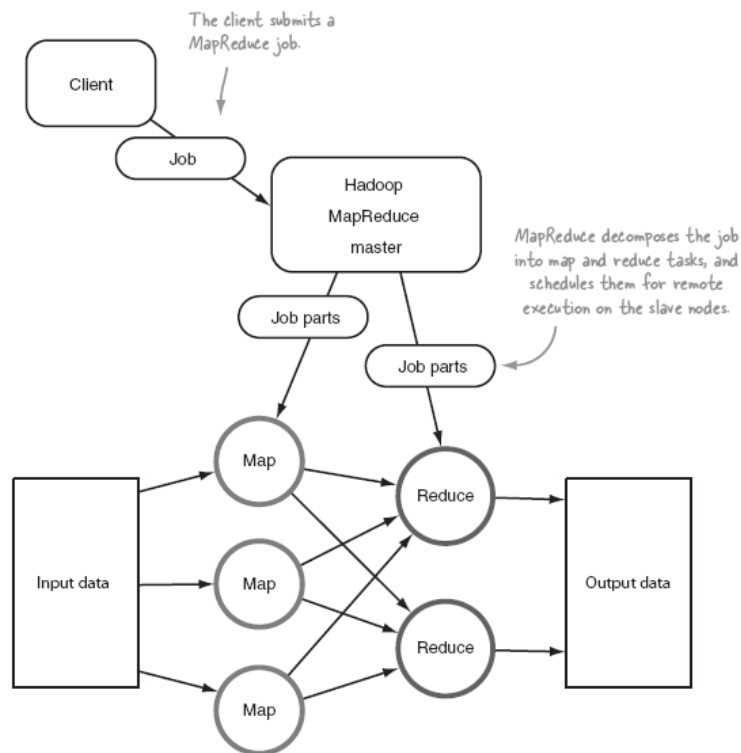


Figure 23The Hadoop Map/Reduce architecture [83]

Programs written in this style are automatically parallelized and executed on a large cluster of machines. Hadoop takes care of the partitioning details of the input data, scheduling the execution of the different tasks via job-trackers and task-trackers, handling machine failures, and managing the communication between the different machines. Hadoop uses the Hadoop Distributed File-system (HDFS) to manage and handle the data.

Map/Reduce design was employed for many applications that involve large amounts of data. Spam filtering [84], seismic data analysis and monitoring [14][85], image and video processing [86], video watermarking [87][88], cloud security[99], web

video categorization[90], and SVM classifiers [91][92] to mention a few, have been already applied on the Hadoop framework.

Matlab [94] has introduced a distributed and parallel platform on which the Hadoop Map/Reduce method can be applied. The parallel implementation allows multithreading, while the distributed form can be performed on a cluster [95]. Reconfigurable Accelerated Solid State Drives (RASSD) is a distributed platform for data intensive applications which uses solid state drives combined with field programmable gate arrays (FPGA) placed at the different nodes of the distributed system [96]. FPGA are used to accelerate processes by hardware design [97]. FPGAs have also been applied for seismic attribute extraction as illustrated in [98-102]. The RASSD platform (Figure 24) assumes that the data is already dispersed in different geographical locations and schema servers are used to resolve the location of the data. A client sends a query request to the middleware server which processes the query and identifies the location of the concerned data and the processing required. Each server being connected to multiple RASSD nodes downloads the appropriate function codes called “drivelets” and the corresponding acceleration configuration bit stream onto the RASSD’s active processing unit. The capabilities integrated into this system can be built using open source frameworks such as Hadoop Map/Reduce. From the application perspective, the programmer needs to identify the data, its location, the processing required, the needed acceleration as well as the needed parameters. The RASSD OS is a real-time multitasking operating system that uses 32 bit MicroBlaze soft processor core available for Xilinx FPGA’s [103]. The K-means was distributed and analyzed on Hadoop [104] and even accelerated using FPGAs [105].

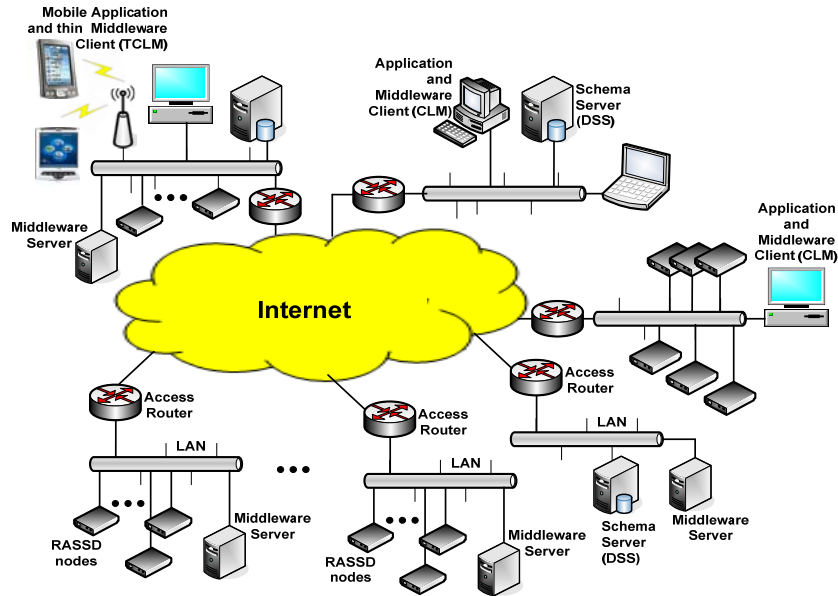


Figure 24 Overview of the general system architecture of the RASSD [96]

Many of the algorithms employed during seismic data analysis are usually applied per data unit such as per trace, or per image window, or per frequency range. Thus, these algorithms can be easily parallelized where processing of units can be performed separately. The results from the different parallel processes can be combined into a final aggregated result.

2.5 Literature Review on Imbalanced Classification and Other Related Work

Image processing techniques are used in seismic data analysis as well. Handling noise data enhances the signal to noise ratio by filtering or resisting the noise [106]. Instances that are suspected to be noisy are discarded based on some evaluation

criteria, or replaced by a more appropriate value [107-110]. Feature extraction techniques have been developed to render the attribute set into a concise and non-redundant attribute set. Techniques include principal component analysis (PCA), independent component analysis (ICA) [111], correlation, and the regularized discriminant analysis (RDA) [112]. Support vector machines, neural networks, and decision trees are among the popular classifiers that are commonly used in pattern recognition problems [113].

Classification of large and imbalanced datasets is one of the leading challenges in data analytics since traditional machine learning algorithms are usually biased towards the majority class. In many real life applications the data is imbalanced i.e. the important class has much less instances than the other class. The numerous efforts that have aimed to learn from imbalanced datasets can be divided as either data, algorithmic, or kernel based nature. At a data level, resampling methods such as: over-sampling of the minority class to balance the class distribution by replication and under-sampling of the majority samples which balances the data by eliminating samples randomly from the majority class [114][115][116], were proposed to resample the data prior to training. Because under-sampling methods proved to be inefficient because of the loss of important information [117], the Synthetic Minority Over-Sampling Technique (SMOTE) [118] added “synthetic” data to the minority class by k nearest neighbor. Other extensions of the SMOTE algorithm have been developed such as SMOTE-RSB which uses the rough set theory [119], the Safe Level-SMOTE [120] where safe level coefficients are computed by considering the majority class in the neighborhood, and the Borderline-SMOTE [121] which oversamples the minority class at the borderline by

considering the nearest minority neighbors. An extension of Borderline-SMOTE, borderline-SMOTE-2, oversamples the minority class by considering the nearest neighbor from the majority class in addition to the k nearest minority neighbors. Also, SPIDER [122] combines local over-sampling of the minority class with filtering difficult examples from the majority class.

Extensions to Support Vector Machines are among techniques that have been proposed at an algorithmic level to handle imbalanced datasets. Veropoulos et al. assigned different error costs for each class [123] while Tang and al. merged a cost sensitive learning approach that extracted a smaller number of the support vectors, with different error costs [124]. Another popular technique is the one-class SVM which estimates the probability density function and gives a positive value for the elements in the minority class and a negative value for everything else [125][126] as if the cost function of the majority class samples is taken to be zero. Scholkopf et al. proposed matching the data into a new feature space using a kernel function and separating the new samples from the origin with a maximum margin [127]. Support Vector Data Description aims at finding a sphere that encompasses the minority class and separates them from the outliers. Since kernel parameters influence the size of the region, correct tuning is essential for satisfactory accuracy [128]. ν SVM modified the decision boundary in such a way to remove the minority class's bias towards the majority class [129]. Other techniques have also been proposed to solve the problem of imbalanced datasets that include the combination of two or more different algorithmic approaches [130-134]

Kernel modification methods among which are Class Boundary Alignment (CBA) and Kernel Boundary Alignment (KBA), transform the kernel function to enlarge the region around the minority class in an attempt to overcome the imbalance problem [135][136][137]. Wu and Chang point out that since positive samples lie further from the ideal boundary; SVMs produce skewed hyper-planes with imbalanced datasets.

CHAPTER III

COMPUTATIONAL DESIGNS, FLOWS, ALGORITHMS AND ANALYSIS

3.1 Introduction

As evident by the discussion made in the previous section, seismic data analysis is naturally complex. After their collection, seismic data undergoes a series of composite algorithms (pre-processing, attribute computation, and data analysis) to identify interesting sites such as oil and gas reservoirs, drilling sites, faults and salt domes. Each of these site identification problems employs a diverse yet a specific set of attributes. Selected by an expert in the field, the attributes could be further analyzed to select the most significant ones by using feature analysis methods (such as PCA, correlation). Finally, the final attribute set is used as an input to a classifier to identify interesting locations in the exploration sites.

Seismic data are by nature so huge and processing large amounts of data results in a more expensive computational cost. Distributed systems have proved to reduce the computational expense of problems consisting of complex procedures and that handle volumetric data. In addition, using the FPGAs to accelerate the computationally most expensive portions of the different algorithms may further result in larger computational gains.

On the other hand, seismic attribute data may be imbalanced since oil bearing samples will be relatively much smaller relative to the non-oil bearing ones. We propose

in the last section, the Barricaded Boundary Minority Over-sampling to overcome the problem of the bias of LS-SVM.

3.2 Seismic Data Processing and Analysis Workflows

In this section, we begin with the typical sequential processing flows employed during seismic data analysis. Later, the flows are migrated into a distributed design using the Map/Reduce paradigm. Using this method, we finally illustrate an example flow that employs seismic images to compute textural attributes and uses the incremental LS-SVM to predict hydrocarbon-bearing sites employing Map/Reduce.

3.2.1 Sequential Flow of Seismic Data Interpretation

As illustrated in figure 25, the overall flow of the seismic data analysis and interpretation can be divided into two major phases: the data processing phase and the classification phase.

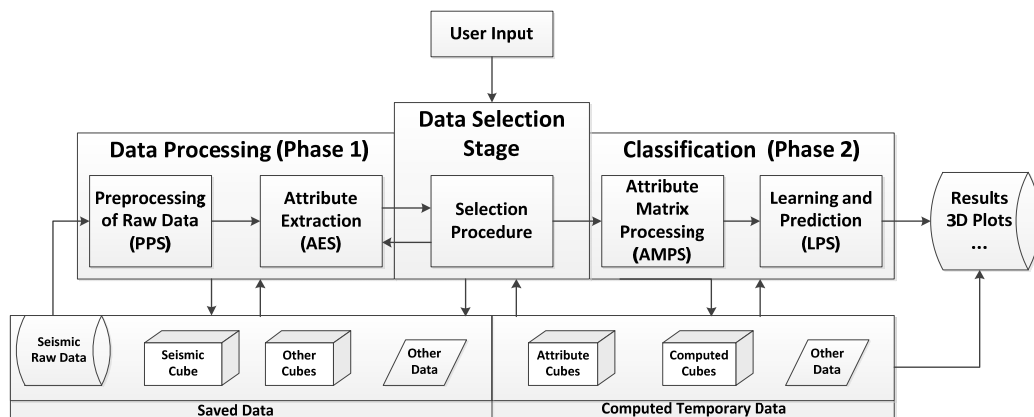


Figure 25 Overview of the general system flow of seismic data processing

The first phase can be divided into two major stages:

- the Pre-Processing Stage (PPS) : where the collected raw seismic data is reconditioned before it is used to produce migrated 3D models of the exploration site (Figure 5) .
- the Attribute Extraction Stage (AES) where the output data cubes produced during PPS will be used to compute the seismic attributes (Figure 26).

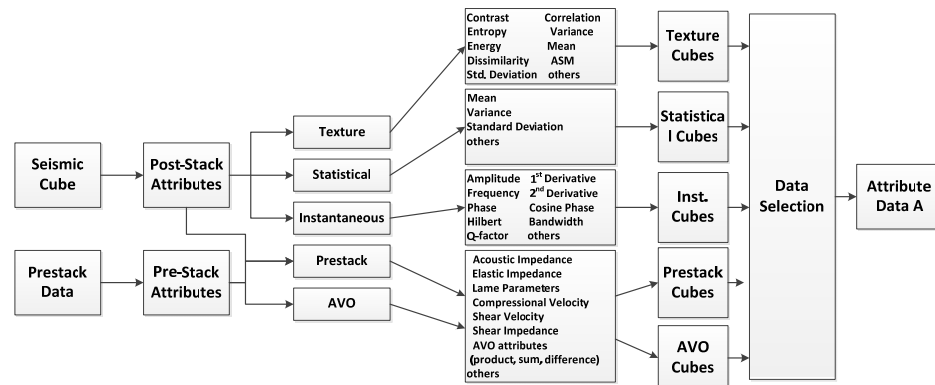


Figure 26 Attribute Extraction Stage

Many of these cubes, such as the seismic cube, are permanently stored in the memory since they are used frequently during AES. Attribute data are computed from pre-stack and post-stack data depending on their type as shown in figure 26. Different types of attributes are computed in different ways. Textural, statistical, geometrical, and instantaneous data can be extracted from the post-stack data. AVO attributes and Inversion analysis employs pre-stack and post-stack data and produces a different set of attributes.

The attributes are saved in the form of attribute cubes. Typically an expert selects a set of suitable attributes for each application during the Data Selection Stage (DSS) during which the chosen attribute set is either fetched or computed and arranged into the attribute matrix A.

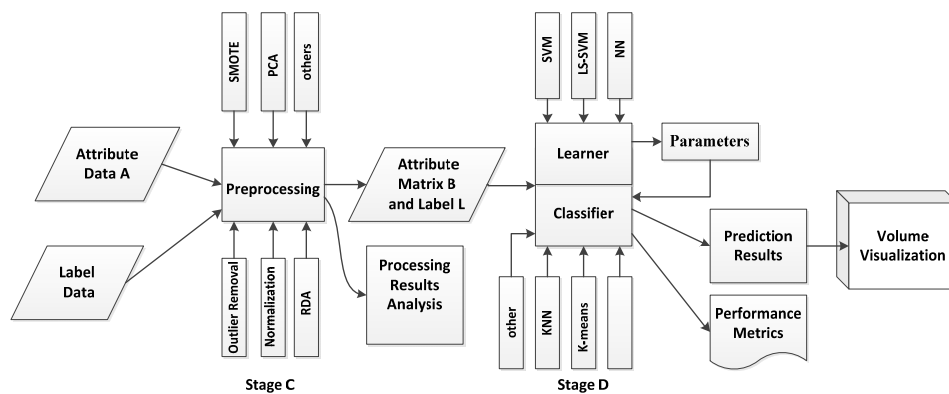


Figure 27Classification stage

In any supervised learning, labeled data are needed. Both the labels and their respective attribute matrix are employed as inputs to the Classification Phase which is in turn divided into two major stages:

- The attribute matrix processing (AMP): where the data is reconditioned and enhanced before being fed to the classifier.
- Learning and Prediction Stage (LPS): where different classification and clustering algorithms are employed.

Normalization, imbalanced data analysis, PCA, ICA, RDA, noise removal, and other methods can be employed to process the data. SVM, LS-SVM, NN, and K-means

are some examples of popular classifiers. At this stage, classifiers are learnt according to the label and attribute matrix and then predictions are made. The results are then analyzed through the performance metrics available for each classifier and repeated if needed. The prediction results are used to analyze the sites through volumetric analysis and visualization to identify and assess the different geological structures in the site.

Figure 27 depicts the classification stage.

3.2.2 Distributed Seismic Data Processing Workflows

In general, every problem that can be written as a summation can be parallelized. Seismic analysis involves many processes which are in this form and hence many of the operations can be distributed-ly computed. We employ Hadoop's Map/Reduce paradigm to illustrate the distributed form of the different procedures. We first define for every task a splitter input data format and splitter output data format depending on the process which we intend to parallelize (Figure 28). The data splitter function will be responsible for indexing and splitting the data into smaller portions into the different nodes of the distributed system. Every process that can be parallelized is defined by a mapper and a reducer each of which inputs and outputs specific <key,value> pairs. A special combiner method is needed to recombine the distributed processed data. Specific data input and output formats are defined for the combiner as well. The combined data is then saved into the data cubes or memory for later use.

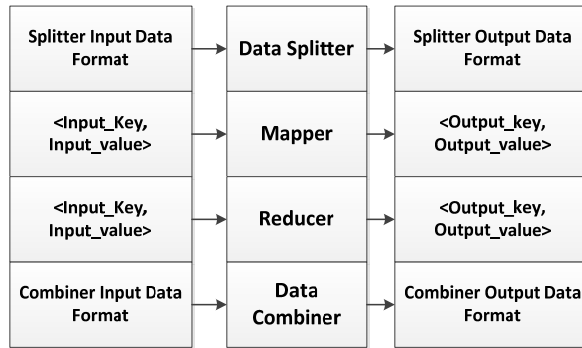


Figure 28 Map/Reduce input and output

Figure 29 depicts a Map/Reduce design of the pre-processing and attributes extraction phases. The original seismic data is split over the different nodes first using the Data Splitter 1 function. Each of the pre-processing algorithms (PPSi) that have the capability of being designed in the Map/Reduce framework are coded as Map/Reduce pairs and processed distributed-ly. For each mapper and reducer, specific <key,value> pairs are defined. Special data splitters and combiners manage the data over the distributed system.

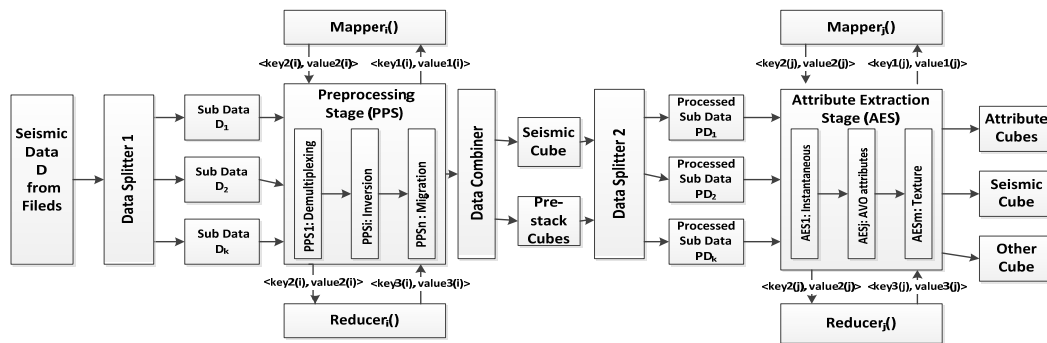


Figure 29 Map Reduce based architecture of the preprocessing stage (PPS) and the attribute extraction stage (AES)

As an example, a distributed implementation of Kirchhoff Time Migration using Hadoop's Map/Reduce was studied in [10]. Since individual traces are processed separately and then summed based on their travel times, PKTM can be directly parallelized [48]. In this example, the input $\langle \text{key}, \text{value} \rangle$ pair are the trace number and the sampling values of the trace respectively. Mappers independently process the different $\langle \text{key}, \text{value} \rangle$ pairs to migrate the data using sequential forward PKTM. The reduce function simply gathers the migrated values to produce the final image. Another example of parallel computation of the correlation process using seismic data is illustrated in [125]. The correlation runtime was reduced to produce improvements by a factor of 19. Similarly, during the attribute extraction phase (AEP), parallelization is performed again using the Map/Reduce paradigm and newly defined set of mappers, reducers, and $\langle \text{key}, \text{value} \rangle$ pairs for each attribute. Typically, the user must select the attributes needed for a specific application.

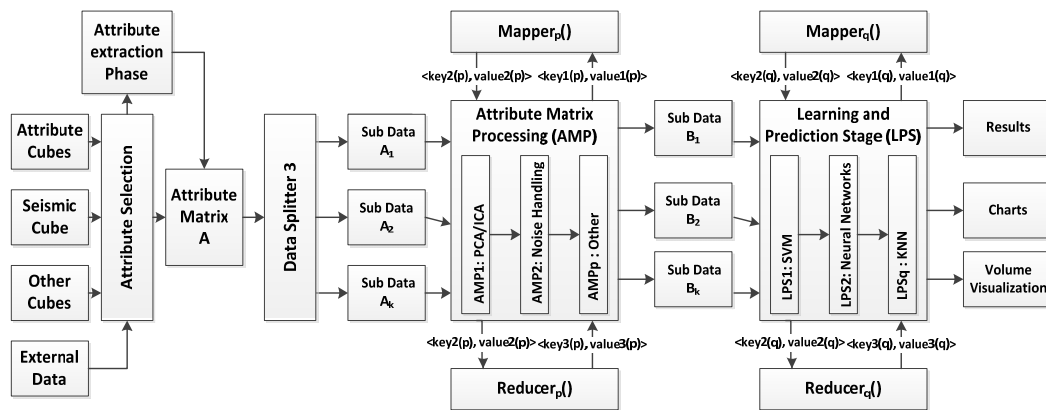


Figure 30 Map/Reduce based architecture of seismic attribute data processing (AMP) and the learning and prediction stages (LPS).

The selected attributes are computed if they are not already stored in the attribute cubes. As shown in figure 30, the attribute matrix processing (AMP) follows where the data is processed in a distributed fashion. Some of the processes in this phase may alter the data into a new attribute set, and other processing types may result in new data or attributes that are aggregated to the original seismic attribute set to produce the matrix B. Finally, the attribute matrix B is used along with some external data such as label and the classifier parameters to classify for a specific type of seismic event. Classifiers are also learned and used in a distributed fashion.

3.3 Seismic Texture Analysis

As an example to seismic data processing and analysis, we present in this section a sample application of seismic texture analysis. The study includes data preparation, data distribution and windowing, four Haralick attribute extraction, and classification using distributed LS-SVM. This application itself employs the migrated seismic data cube produced by data preprocessing stage PPS. The seismic cube is supposed to be divided into a number of 2D slices or images numbered based on their location in the volume. Images were extracted from the volume and numbered based on their location from $1 \dots N_{\text{images}}$ to keep track of their respective coordinates as shown in figure 31.

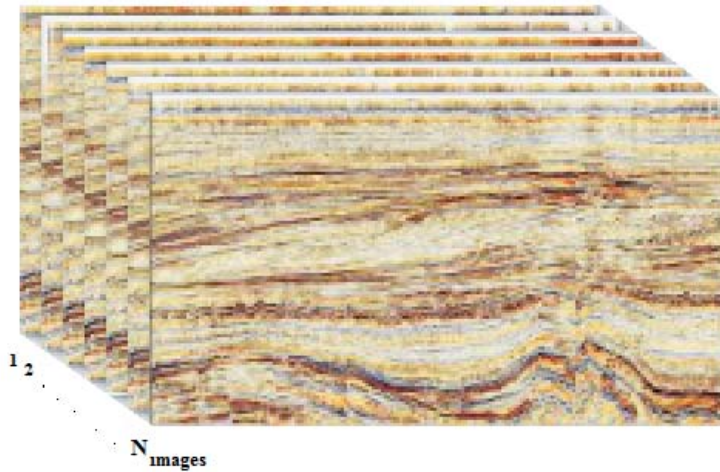


Figure 31 Seismic slices from the 2D volume taken as images

Every Image is supposed to be read and processed separately serially in the serial design and in a distributed manner in the distributed design. Each image is then subdivided into a number of 32x32 windows which is in turn subdivided into exactly 49 (8x8) windows by taking half-way offsets. The window size 8x8 can be easily changed into 16x16 and then 32x32 by changing a variable. However, most of the design was based on an 8x8 window size.

Figure 32 represents the overall flow of Seismic texture extraction. Each of the windows is processed separately to find their respective GLCMs. This matrix holds the probability of the co-occurrence of the grey levels N_g . From the GLCM, all the Haralick attributes can be computed.

As pointed out by Chopra and Alexeev in [43], low-frequency and high-amplitude anomalies, that are indicative of hydrocarbon accumulation, exhibit high energy, low contrast, and low entropy, when compared to non-hydrocarbon sediments.

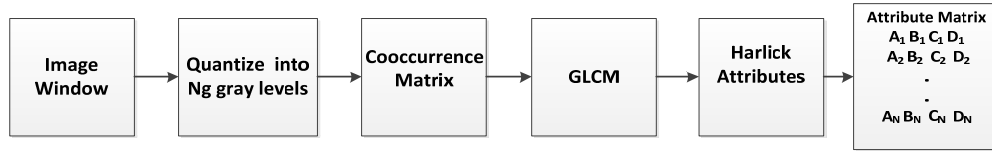


Figure 32Texture attribute computation flow

The following Haralick attributes were considered to identify oil bearing sites.

- 1) Contrast: which uses weights relative to the distance from the GLCM diagonal
- 2) Energy: which has high values when an image is very orderly
- 3) Entropy: which is also indicative of the amount of information an image represents
- 4) Variance: which is the popular statistical measure similar to entropy

Every sub-window thus produces four Haralick attributes. All are gathered into a single matrix called the attribute matrix which is properly indexed to keep track of the positions of the data. An oil bearing block of the 3D seismic data is used during the labeling process. The block is assumed to be labeled by an expert that identifies locations as oil or non-oil bearing.

An oil bearing block of the 3D seismic data is used during the labeling strategy (Figure 33). The locations of the oil bearing sites in each of the 2D images in the 3D block are labeled by following the following strategy.



Figure 33Labeling strategy

A label of 1 corresponds to oil bearing sites and a label of 0 for non-oil bearing sites. The labels can be also imagined as 2D label images containing zeros and ones. These label images are again divided the same way the seismic images were divided resulting into sub-windows of size 8x8. These windows can now be considered as two tone gray level images. The normalized GLCM of these label images produce a 2x2 matrix since we have only two possible values. We find the sum of probabilities of the non-zero-zero occurrences, P_{NZZ} , and consider the label of a window to be 1 if it's greater than a certain threshold "th" (which takes a default value of 0.5 in this study).

$$\begin{array}{l}
 \text{Label Matrix} \\
 \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \rightarrow \begin{array}{c} 0 \\ 1 \end{array} \begin{bmatrix} 0 & 0.21 \\ 0.21 & 0.58 \end{bmatrix} \rightarrow P_{NZZ} = 1.00 \rightarrow \text{Label} = 1 \\
 \\
 \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \rightarrow \begin{bmatrix} 0.58 & 0.16 \\ 0.16 & 0.08 \end{bmatrix} \rightarrow P_{NZZ} = 0.416 \rightarrow \text{Label} = 0 \\
 \\
 \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 0 & 0.50 \\ 0.50 & 0 \end{bmatrix} \rightarrow P_{NZZ} = 1.00 \rightarrow \text{Label} = 1
 \end{array}$$

Figure 34 Labeling examples

Consider figure 34, where three 4x4 example label matrices are shown with their corresponding label. In the first window, the probability of zero-zero occurrence is zero, which indicates that there are no non-oil bearing sub windows beside each other. This obviously results into a positive label which indicates a site that contains oil. In the

second window has high probability of non-zero-zero occurrence and it will have a zero label. The last window shows alternating zeros and ones, however, its P_{NZZ} is 1 and it is labeled positive.

When the attribute matrix and the label data are ready, they are passed to the learning and the prediction phase as shown in figure 35.

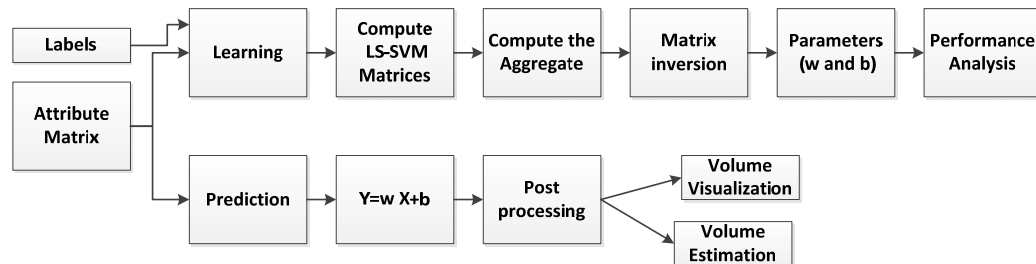


Figure 35 Learning and prediction flow using texture attributes

3.4 Seismic Texture Analysis Pseudocode and Complexity Study

In this section we describe the different methods used in this application detailing the input, the output, and the procedures. Each of these performs a specific task and will be used in different ways during both the learning and the prediction phases.

The pseudo code for the sub programs called operations are:

Operation1: Read the training and label image

Input: Index of the training image

Output: Image and label matrices

Operation2: Computation of the Haralick attributes

Input: Training image

Output: Haralick attribute matrix

Steps:

- Read 32x32 matrix window from image

- Cut it into 49 (8x8) windows
- Compute GLCM of each 8x8 window resulting in an 8x8 GLCM matrix with 8 grey levels
- Compute the Haralick attributes producing 4 values
- Aggregate into the Haralick Attribute matrix

Operation3: Compute the label vector

Input: Label matrix

Output: Label Vector

Steps:

- Read 32x32 matrix window from label image
- Cut it into 49 (8x8) windows
- Compute GLCM of each 8x8 window resulting in a 2x2 GLCM matrix with 2 grey levels
- Compute the non-zero-zero probability
- Compute the label based on a threshold
- Aggregate into a label vector

Operation4: Compute the matrices

Input: Haralick matrix from operation 2 and the label vector

Output: Incrementally added matrices

Steps:

- Form the diagonal label matrix
- Compute the intermediate sum of matrices as described in equations 14 and 15

Operation 5: Compute the LS-SVM parameters

Input: the computed matrices from operation 4

Output: ω, b

Steps:

- Aggregate incoming matrices from operation 4
- Matrix inversion
- Compute w and b using equation 16

Operation 6: Predict the label

Input: The Haralick Attributes and ω, b

Output: the label of the image

Steps:

- Predict using equation 12
- Aggregate results into their correct position

During the learning phase, operations 1 through 5 are used. On the other hand, only operation 1, operation 2 and operation 6 are employed during the prediction phase. The complexity analysis of these operations is shown in table 2.

Table 2.Computational Complexity of the Different Operations

Operation 1	$\Theta(MxN)$
Operation 2	$\Theta(\frac{M}{m'} * \frac{N}{n'} * [M * N + (3N_g^2 + p^2 + \Theta(\log \text{ function}))p^4])$
Operation 3	$\Theta(\frac{M}{m'} * \frac{N}{n'} * [M * N + p^4])$
Operation 4	$\Theta(5 * m' * n')$
Operation 5	$\Theta(m' * n')$
Operation 6	$\Theta(N_s)$
MxN : image size m'xn' : subimage size	p : window size N _s : number of samples

The most complex operation is operation 2 where the 2D GLCM and the Haralick attributes are computed in both training and prediction phases. GLCM's complexity is $\Theta((2 * k^2 + 20) * p^2)$ while that of the Haralick attributes is $\Theta(N_g^2(\Theta(\ln) + p^2))$. In the case of the 3D GLCM, the fetching of the pixels is performed in a 3D volume or a moving box instead of a 2D window. The GLCM complexity in the 3D case is $\Theta((2 * k^2 + 20) * p^3)$ where p is the window cube's dimension. The Haralick attributes will still have the same complexity as in the 2D case.

3.5 Map/Reduce Based Design of Seismic Texture Analysis

In the Map/Reduce framework, the above processes are designed as follows. During the attribute extraction phase, we define a mapper that handles the seismic images and their respective labels. The mapper uses the image number as the key and the image as the value. For the seismic image, the output <key,value> pair will be the image number and the corresponding computed Haralick attributes. The labels are tied to their respective images. The reducers simply join the produced values into a single image matrix and label vector.

The resulting data is then passed to the classification stage where classification is performed using the Map/Reduce design. The attribute matrix and label vector are distributed evenly on the different nodes with proper indexing. The <key, value> pair in this case are the index and a portion of the attribute matrix and the label vector. LS-SVM has to compute the two matrices $d_{1i} = E_i^T E_i$ and $d_{2i} = E_i^T D_i e_i$ as discussed in the previous chapter. After all the windows are processed, the matrices are aggregated into a single matrix. Then, the LS-SVM parameters ω and b are computed using equation (12) which involves an inversion of a 5x5 matrix.

The Learning Phase (Figure 36)

Preparation

- Data distribution to nodes, images are split on the different nodes available
- Set initial parameters such as the penalty term C , Number of gray levels, window size.

Main Class

- Configure the job
- Set the mapper and reducer classes

Mapper Function

- While (there are still images to read)
- Read parameters from the main class

- Load seismic cube section and label image using operation 1
- Compute Haralick attributes using operation 2
- Compute the Labels using operation 3
- Calculate partial matrices of the classifier using operation 4
- Local summation of the partial matrices of the classifier

Reducer (Operation 5)

- While (there are still nodes that didn't finalize sending the data)
- Read the local summation matrices from the nodes and aggregate them into a single matrix
- Compute the parameters ω and b

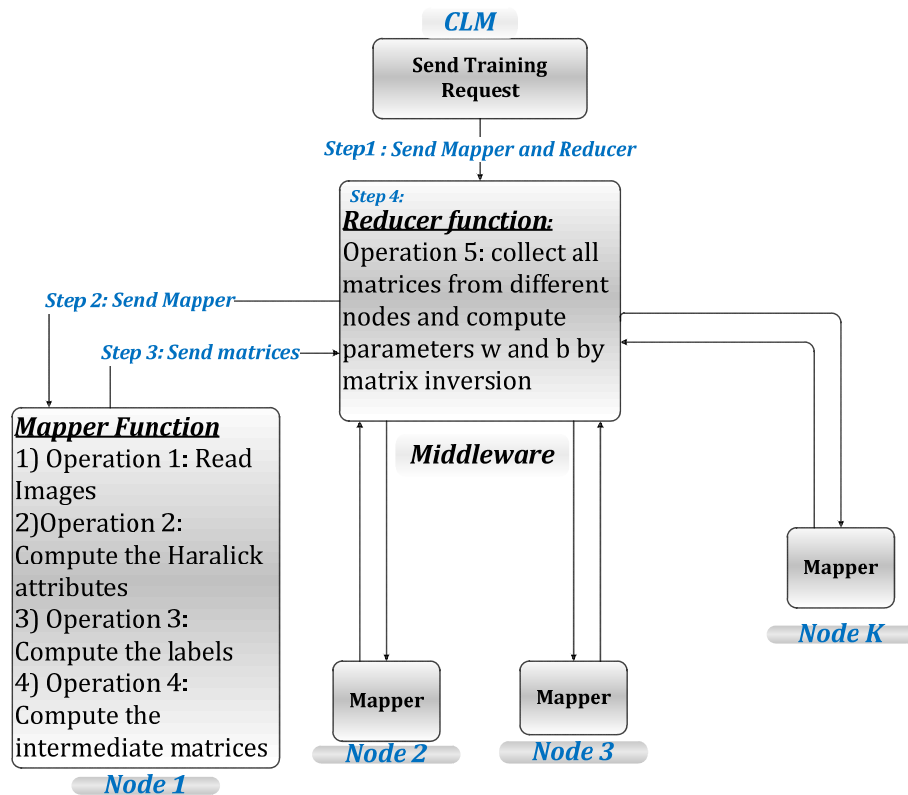


Figure 36 Distributed Map/Reduce design of the learning phase

During the prediction phase, the attributes of the new images are computed the same way as in the learning phase. The aggregated attribute matrix is used in equation (12) to produce a label. The output labels are then reassembled to give an overall 3D

volume which shows the locations where there is a high probability of hydrocarbon presence. The <key,value> pairs here are the index and their respective Haralick attribute values or matrix if taken as a block.

Prediction Phase (Figure 37)

Preparation

- Data distribution to nodes, images are split on the different nodes available
- Set initial parameters such as the penalty term C, number of gray levels, window size,...

Main Class

- Configure the job
- Set the mapper and reducer classes

Mapper Function

- While (there are still images to read)
- Read learnt parameters from the main class
- Load Seismic section using operation 1
- Compute Haralick attributes using operation 2
- Local aggregation of the partial Haralick matrices

Reducer (Operation 5)

- While (there are still nodes that didn't finalize sending the data)
- Read the matrices from the nodes and aggregate them into a single matrix
- Compute the labels using equation

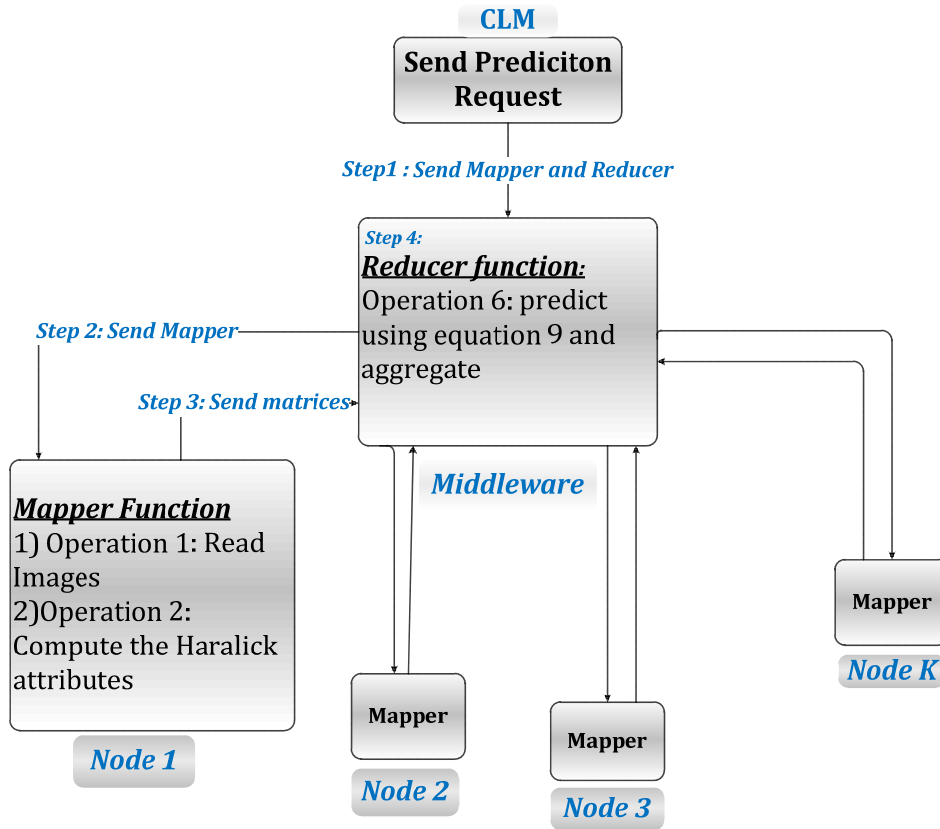


Figure 37 Distributed Map/Reduce design of the prediction phase

A combiner function is defined to control the exact locations of the predicted results in the 3D volumes. The combiner in our application simply reconstructs the label images and then arranges the images in the order of their index or numbering to produce a 3D volume as shown in figure 38.

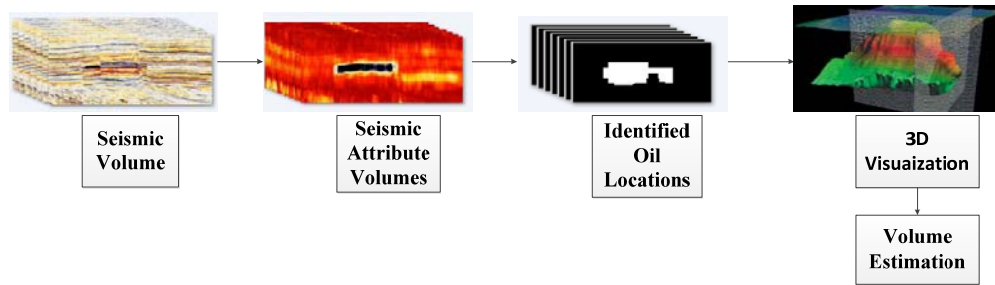


Figure 38 Volume estimation and visualization

The unit length which is relative to the site and instrumentation is used and multiplied with the estimated volume of the predicted oil that has resulted from meshing the different predicted positively labeled images:

$$V = V_{unit} \times V_{estimated} \quad (17)$$

3.6 Barricaded Boundary Minority Oversampling Method

LS-SVM often produces biased hyper-planes when trained with imbalanced datasets. In this section, we describe the Barricaded Boundary Minority Oversampling (BBMO) method that oversamples the minority samples in the direction of the closest majority samples to solve the problem. The nomenclature used with BBMO, the formulation, BBMO and LS-SVM formulation merged together, and its complexity analysis is described in this section.

3.6.1 Nomenclature

Consider the following nomenclature for this section.

X :training samples	α : Lagrange multipliers of LS-SVM
X_1 :minority class samples	α_{\max} : maximum Lagrange multiplier of minority samples
X_2 :majority class samples	α_{\min} : minimum Lagrange multiplier of majority samples
m :number of minority samples	th_1 :threshold value for the linearly separable case
n :total number of samples	th_2 :threshold value for the linearly non-separable case
d : dimension of input space	Ψ_{B1} :matrix holding the replicated minority samples
B : the boundary samples	Ψ_{B2} :matrix holding the replicated majority samples
X_{B1} :boundary minority samples	$K(.,.)$:kernel matrix
X_{B2} :boundary majority samples	γ :the weight for the computation of the weighted means
n_{B1} :number of boundary minority samples	IR :imbalance ratio of the dataset ($m/n-m$)
n_{B2} :number of boundary majority samples	Z : weighted means , the "barricade"
n_z :number of synthetic weighted means	

3.6.2 BBMO formulation

LS-SVM as originally proposed by Suykens and Vanderwalle introduced two major changes to SVM[66]. First, the error term in SVM was changed to a least square error. Second, the inequality constraint was changed into equality. These changes render the hyper-plane's orientation more data oriented instead of being oriented by the support vectors. Therefore, LS-SVM suffers from sparseness [138] since all the dataset is considered to behave as support vectors.

Taking LS-SVM drawbacks into consideration, BBMO creates synthetic minority samples at the hyper-plane boundary in the direction of the majority boundary samples to push the hyper-plane away from the minority to remove the bias.

Two variations of BBMO are represented: BBMO-1 which handles the linearly separable case by adding synthetic data in the direction of all majority boundary samples

and BBMO-2 of the linearly non-separable case that uses the kernel matrix values, which is the general case, adds synthetic data around each boundary minority sample in the direction of all close majority boundary samples if there are any.

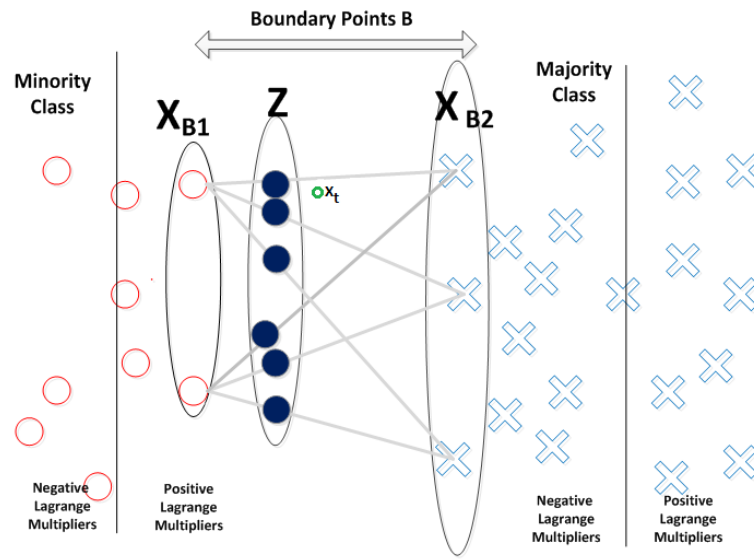


Figure 39 Illustrative extraction of the boundary samples of the linearly separable case and the formation of the barricade Z by calculating the weighted means of all the boundary samples

First, let us consider the linearly separable case. LS-SVM computes the Lagrange multipliers of all the samples and produces positive Lagrange multipliers for all the minority samples within the band and negative values for the majority samples (labels are taken to be 1 for the minority and -1 for the majority) as shown in figure 39. To find the samples closest to the boundary, we first find the absolute maximum Lagrange multipliers of both classes then select the samples based on a threshold th_l . Next, we compute the inter-class weighted means of the selected boundary samples by considering all the combinations forming a “barricade” Z in front of the minority boundary samples in the direction of the majority boundary samples. The set Z

represents the weighted means which employ a weight γ that varies between 0.5 and 1 (since the synthetic samples are closer to the minority boundary samples). We assume the weight varies linearly with the imbalance ratio (IR) of the dataset as follows:

$$\gamma = 0.5 + \frac{1}{2IR}$$

If the two classes are balanced, IR is close to 1 which sets γ to 1 that

results in simply duplicating the minority boundary samples. As the imbalance ratio increases, γ takes smaller values (tending to 0.5) resulting in synthetic samples that occupy further locations around the minority boundary samples widening the distribution of the minority samples at the boundary however in the direction of the majority boundary samples. This will result in pushing the hyper plane away towards the majority removing the bias as shown in figure 40. The following algorithm represents the BBMO-1 algorithm for the linearly separable case.

BBMO-1 algorithm workflow: the linearly separable case

- 1) $\forall x \in X$, compute α using LS-SVM
- 2) $\forall x_i \in X_1$, $\alpha_{\max} = \max(\alpha_i)$ where $i=1,2,\dots,m$
 $\forall x_j \in X_2$, $\alpha_{\min} = \min(\alpha_j)$ where $j=1,2,\dots,n-m$
- 3) Let $0.8 \leq th_1 \leq 1$
 $\forall x_i \in X_1$, if $\{\alpha_i > th_1 \cdot \alpha_{\max}\}$, add x_i to X_{B1}
 $\forall x_j \in X_2$, if $\{\alpha_j < th_1 \cdot \alpha_{\min}\}$, add x_j to X_{B2}
- 4) $\forall x_k \in X_{B1}$, where $k=1,2,\dots,n_{B1}$
 $\forall x_l \in X_{B2}$, where $l=1,2,\dots,n_{B2}$
 Compute $z_p = \gamma x_k + (1-\gamma)x_l$ where $p=1,2,\dots,n_z$
 $n_z = n_{B1} \cdot n_{B2}$, $z_p \in Z$, $\gamma = 0.5 + \frac{1}{2IR}$, $IR = \frac{m}{n-m}$
- 5) Add Z to X_1 and train using LS-SVM

This approach may be successful in linearly separable cases, however when the data becomes linearly non-separable and kernels are used to solve the classification problem, finding the boundary samples cannot be achieved using the Lagrange multipliers of the LS-SVM. In addition, the orientation of the added synthetic data may be very random and the number of oversampled instances will be very large creating in this case a bias towards the minority. Instead we adopt another method which can be generalized to the non-linear case. In our study, we employ the RBF kernel. Using the kernel matrix values instead of the Lagrange multiplier values to find the boundary samples, for each minority class sample, the maximum kernel value with each of the majority sample is computed. Next, the maximum of these values is extracted and used to select the boundary samples of the minority class with the corresponding closest majority boundary samples according to a threshold th_2 . For each boundary minority sample and its corresponding “near” majority boundary samples, the weighted means are computed and added to the minority class. The following algorithm describes the steps in detail.

BBMO-2 algorithm: the non-linear case

- 1) $\forall x_i \in X_1, \forall x_j \in X_2$, find $K(x_i, x_j)$ where $i=1,2,\dots,m$ and $j=1,2,\dots,n-m$
- 2) $\forall x_i \in X_1$ compute $\max(i)=\max\{K(x_i, x_j)\}$
- 3) calculate $\text{Max}=\max\{\max(i)\}$
- 4) let $0.8 \leq th_2 \leq 1$
 $\forall x_i \in X_1, \forall x_j \in X_2$, if $\{K(x_i, x_j) > th_2 \cdot \text{Max}\}$
 add x_i to X_{B1} and x_j to $X_{B2}(q)$ where $q=1,2,\dots,n_{B1}$
- 4) for each $x_k \in X_{B1}, \forall x_r \in X_{B2}(q), r=1,2,\dots,\text{Card}(X_{B2}(q))$
 compute $z_p = \gamma x_k + (1-\gamma)x_r$ where $p=1,2,\dots,\text{Card}(X_{B2}(q))$
 knowing that $\gamma=0.5+\frac{1}{2IR}$, $IR=\frac{m}{n-m}$
 add z_p to Z
- 5) repeat step 4 for all x_k where $k=1,2,\dots,n_{B1}$
- 6) add Z to X_1 and train using LS-SVM

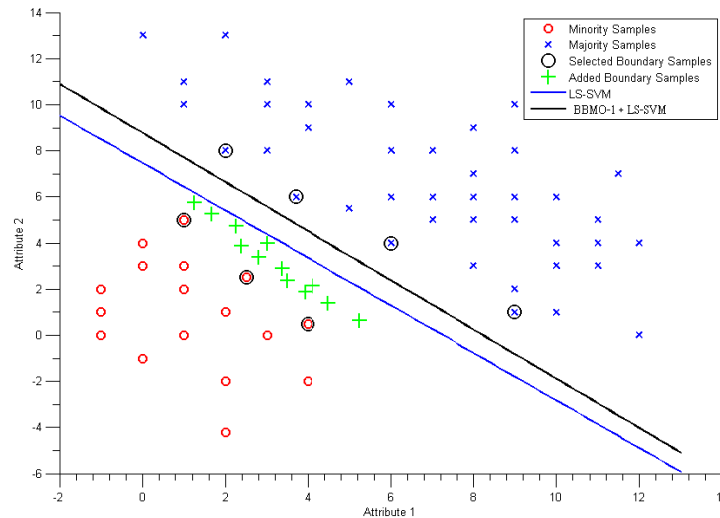


Figure 40BBMO-1 pushes the hyper-plane away from the minority by adding the Barricade

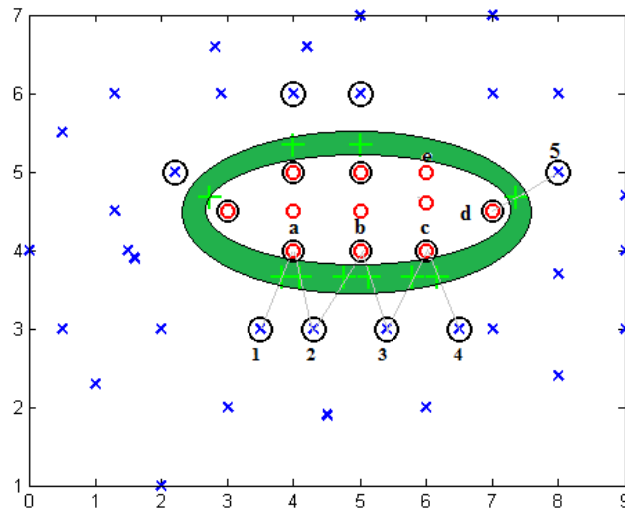


Figure 41 BBMO-2 oversamples in the direction of the closest majority

Consider figure 40, which describes the behavior of the BBMO-1 algorithm for the linearly separable case on a synthetic 2D dataset. The boundary minority and majority samples are selected based on the Lagrange multiplier values and the synthetic weighted means are generated. When LS-SVM is applied on the new dataset, the resulting line is pushed away towards the majority class removing the bias. In addition, the boundary is better defined by closing the gaps between the minority class samples. Figure 41 shows another example where BBMO-2 was applied using the RBF kernel on a synthetic 2D dataset. The minority sample “a “ finds majority samples 1 and 2 to be closest within the threshold value and generated two synthetic samples in the direction of these majority samples. Sample d finds only sample 5 and generates a weighted mean in that direction. Other samples, such as e are not even oversampled as there are no close majority samples at the boundary. The distribution of the minority samples will be increased in the direction of the majority as indicated by the green zone. The created

“barricade” around the minority boundary samples will also allow a wider region for the minority samples to populate in. Other oversampling techniques have also aimed at providing more general data distribution for the minority such as SMOTE however assumptions vary. To understand the difference, consider figures 42 and 43.

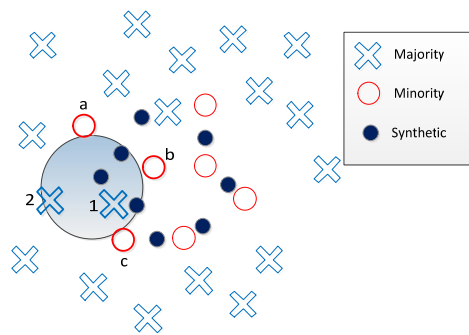


Figure 42 The way SMOTE modifies the minority class distribution when synthetic data are added

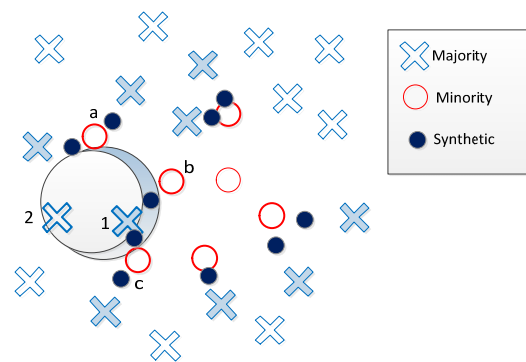


Figure 43 The way BBMO-2 modifies the minority class distribution when synthetic data are added

SMOTE oversamples the minority samples by introducing synthetic samples along the line joining any or all k minority class nearest neighbors depending on the amount of over-sampling needed. As shown in figure 42, samples “a”, “b”, and “c” have each other as nearest neighbors, synthetic samples are generated along their lines. The three synthetic samples in the circle now occupy most of this region although the region contains the majority sample “1”.

BBMO-2 on the other hand adds synthetic data around the boundary minority samples in the direction of all near (not nearest) majority samples. Sample “a” has two

close majority samples close to it, thus two synthetic samples are added in their direction as shown in figure 43. Similarly, synthetic samples are added for samples “b” and “c”. As we can observe, the region in the circular region is not totally occupied by the minority but only a portion of it.

The way oversampling techniques perturb the distribution of the minority sample plays an important role in classification results especially the samples at the boundary. While SMOTE reserves larger areas for the minority samples in the direction of the nearest neighbors, BBMO reserves only the regions surrounding the minority samples at the boundary in the direction of the close majority samples. Thus, BBMO leaves unknown regions unoccupied until new samples arrive and help construct the ideal boundary.

3.6.3 BBMO and LS-SVM

In this section, BBMO is incorporated into the LS-SVM formulation. Consider the following nomenclature:

- ω : hyperplane weight vector
- ξ : error term for the original training data
- ξ' : error term for the synthetic training data
- C : trade-off constant
- b : bias term
- y_i : labels of the training samples $\{-1,1\}$
- Y :vector containg the labels y_i
- y_p : labels of the synthetic samples $\{1\}$
- Y' :vector containing the labels y_p
- α : Lagrange multipliers of the training samples
- β : Lagrange multipliers of the synthetic samples
- e :vector of ones
- Ψ_{B1} : matrix containing the minority class boundary samples replicated the needed number of times
- Ψ_{B2} : matrix containing the majority boundary samples ordered according to the respective minority boundary samples
- I_n : Identity matrix of size n

Merging the BBMO technique into the LS-SVM formulation, the problem can now be defined by

$$\left\{ \begin{array}{l} \text{Minimize } \frac{1}{2} \omega^T \omega + \frac{1}{2} C \xi^T \xi + \frac{1}{2} C \xi'^T \xi' \\ \text{Subject to } y_i (\omega^T x_i + b) = 1 - \xi_i \text{ where } i = 1, 2, \dots, n \\ y_p (\omega^T z_p + b) = 1 - \xi'_p \text{ where } p = 1, 2, \dots, n_z \end{array} \right. \quad (18)$$

where e_p is a vector of ones since the new synthetic weighted means belong to the minority class.

The objective function of the LS-SVM is written as:

$$L(\omega, b, \xi, \alpha) = \frac{1}{2} \omega^T \omega + \frac{C}{2} \sum_{i=1}^n \xi_i^2 + \frac{C}{2} \sum_{i=1}^{n_z} \xi_i'^2 - \sum_{i=1}^n \alpha_i \{[\omega^T x_i + b] + \xi_i - y_i\} - \sum_{p=1}^{n_z} \beta_p \{[\omega^T z_p + b] + \xi'_p - y_p\} \quad (19)$$

Applying the KKT conditions:

$$\frac{\partial L}{\partial \omega} = 0 \Rightarrow \omega = \sum_{i=1}^n \alpha_i x_i + \sum_{p=1}^{n_z} \beta_p z_p \quad (20)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^n \alpha_i + \sum_{p=1}^{n_z} \beta_p = 0 \quad (21)$$

$$\frac{\partial L}{\partial \xi} = 0 \Rightarrow \alpha_i = C \xi_i \quad (22)$$

$$\frac{\partial L}{\partial \xi'_p} = 0 \Rightarrow \beta_p = C \xi'_p \quad (23)$$

$$\frac{\partial L}{\partial \alpha} = 0 \Rightarrow y_i = \omega^T x_i + b + \xi_i \quad (24)$$

$$\frac{\partial L}{\partial \beta} = 0 \Rightarrow y_p = \omega^T z_p + b + \xi'_p \quad (25)$$

Replacing again equations (16), (18) and (19) in both equations (20) and (21), and rearranging the terms we get

$$\begin{bmatrix} b \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} 0 & e_n^T & e_{n_z}^T \\ e_n & XX^T + \frac{I_n}{C} & XZ^T \\ e_{n_z} & ZX^T & ZZ^T + \frac{I_{n_z}}{C} \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ Y \\ Y' \end{bmatrix} \quad (26)$$

Where Y' is a vector of ones since the newly added weighted means belong to the minority class.

Let us define Ψ_{B_1} and Ψ_{B_2} by the two matrices that contain the samples of the boundary points paired in such a way that considers all the possible combinations needed as shown in figure 44.

$$\begin{array}{l}
X_{B1=} \begin{bmatrix} x1 \\ x2 \end{bmatrix} \\
X_{B2=} \begin{bmatrix} x3 \\ x4 \\ x5 \end{bmatrix}
\end{array}
\left. \vphantom{\begin{array}{l} X_{B1=} \\ X_{B2=} \end{array}} \right\} \rightarrow \Psi_{B1} = \begin{bmatrix} x1 \\ x1 \\ x2 \\ x2 \\ x2 \end{bmatrix}, \Psi_{B2} = \begin{bmatrix} x3 \\ x4 \\ x5 \\ x3 \\ x4 \\ x5 \end{bmatrix} \rightarrow Z = \gamma \Psi_{B1} + (1-\gamma) \Psi_{B2}$$

Figure 44Preparing the BBMO matrices to compute the weighted means implicitly using LS-SVM

Since the set Z is dependent on the original set X from which the boundary subset is selected, the computation of XZ^T , ZX^T , and ZZ^T can be done implicitly using the Ψ_{B1} and Ψ_{B2} matrices as shown in equation (23).

$$\begin{aligned}
ZX^T &= [\gamma \Psi_{B1} + (1-\gamma) \Psi_{B2}] X^T = \gamma \Psi_{B1} X^T + (1-\gamma) \Psi_{B2} X^T \\
XZ^T &= (ZX^T)^T \\
ZZ^T &= [\gamma \Psi_{B1} + (1-\gamma) \Psi_{B2}] [\gamma \Psi_{B1} + (1-\gamma) \Psi_{B2}]^T \\
&= \gamma^2 \Psi_{B1} \Psi_{B1}^T + \gamma(1-\gamma) \Psi_{B1} \Psi_{B2}^T + \gamma(1-\gamma) \Psi_{B2} \Psi_{B1}^T + (1-\gamma)^2 \Psi_{B2} \Psi_{B2}^T
\end{aligned}$$

The product between the matrices in equations (22) and (23) can be replaced by any kernel function $K(.,.)$ that satisfies Mercer's condition yielding:

$$\begin{bmatrix} b \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} 0 & e_n^T & e_{n_z}^T \\ e_n & K(x, x) + \frac{I_n}{C} & K(x, z) \\ e_{n_z} & K(z, x) & K(z, z) + \frac{I_{n_z}}{C} \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ Y \\ Y' \end{bmatrix} \quad (28)$$

$$\begin{aligned}
K(x, z) &= \gamma K(\Psi_{B1}, X) + (1-\gamma) K(\Psi_{B2}, X) \\
K(x, z) &= K(z, x)^T \\
K(z, z) &= \gamma^2 K(\Psi_{B1}, \Psi_{B1}) + \gamma(1-\gamma) K(\Psi_{B1}, \Psi_{B2}) + \gamma(1-\gamma) K(\Psi_{B2}, \Psi_{B1}) + (1-\gamma)^2 K(\Psi_{B2}, \Psi_{B2})
\end{aligned}$$

3.6.4 Complexity Analysis

BBMO computes new synthetic samples from the boundary and adds them to the dataset. First, the boundary samples are extracted. Considering the linearly separable case, BBMO-1 uses the Lagrange multipliers of LS-SVM to perform the extraction whose computation is mainly a matrix inversion with complexity $\Theta(n^3)$. Next, the boundary samples are selected using the threshold values includes mainly comparison of the Lagrange multipliers. This operation has a complexity of $\Theta(2n)$. The preparation of Ψ_{B_1} and Ψ_{B_2} and the computation of the weighted means has a complexity $\Theta(4n_z)$. Next, the weighted means are included in the LS-SVM formulation resulting in a larger matrix inversion with complexity $\Theta((n+n_z)^3)$. The overall complexity of the BBMO-1-LS-SVM algorithm is: $\Theta[n^3+(n+n_z)^3+2n+4n_z] < \Theta(3n^3)$ knowing that $n_z \ll n$ and n is large.

BBMO-2 uses on the other hand the kernel matrix values to find the boundary samples whose complexity is $\Theta(n^2d)$. Next, the boundary samples are to be selected using the threshold values includes which uses the kernel matrix values. This operation has a complexity of $\Theta(m(n-m))$. Similar to BBMO-1, the computation of the weighted means has a complexity $\Theta(4n_z)$. Next, the weighted means are included in the LS-SVM formulation resulting in a larger matrix inversion with complexity $\Theta((n+n_z)^3)$. The overall complexity of the BBMO-2-LS-SVM algorithm is:

$$\Theta[n^2d+m(n-m)+(n+n_z)^3+4n_z] < \Theta(3n^3) \text{ as } n_z \ll n \text{ and } n \text{ is large.}$$

CHAPTER IV

EXPERIMENTAL RESULTS

4.1 Introduction

In order to understand the behavior of the different elements studied in this paper, we performed a series of experiments and then merged them into the designed distributed system. After defining the metrics used to evaluate the performance and the data selected in the experiments, BBMO-1 and BBMO-2 were analyzed to understand its impact on the data and the performance with LS-SVM. Next, seismic textures were analyzed where different types of lithological structures were taken and their respective attributes were computed. The seismic volume was studied and accuracy of LS-SVM with/without BBMO and the estimation of the volume were carried out. Finally, the distributed version of seismic texture analysis is studied by first profiling the application to identify the parts which are computationally most expensive, then the results of the distributed computation on Matlab were reported, and later the application was projected into the RASSD distributed system where acceleration of the most computationally expensive operation was explored.

All the experiments in this section were performed on an Intel core i7 2GHz processor with 6GB RAM using MATLAB 7.10.0 except for the acceleration part which were performed and analyzed on the specifications described in the corresponding section.

4.2 Experiments with BBMO

In this section, we briefly present the different experiments performed to assess the performance of the BBMO. First, the performance metrics and the data details are defined. Then, SVMs are used with the BBMO-1 and BBMO-2 and their performance is evaluated and compared.

4.2.1 Performance Metrics

The accuracy of a classifier's performance, which gives the proportion of the total number of predictions that were correctly classified, is a misleading measure especially when used with imbalanced datasets. A classifier might produce high overall accuracy by misclassifying most of the important minority class due to the bias. Instead, many other measures are used to evaluate the performance of classifiers with imbalanced datasets since with extreme imbalance situations a classifier may show high accuracy by misclassifying most of the minority class samples. For a two class classification, true positive (TP), true negative (TN), false positive (FP), and false negative (FN) are the different possible prediction results as described in the confusion matrix of figure 45.

		Predicted Class	
		YES	NO
Actual Class	YES	TP	FN
	NO	FP	TN

$$\text{accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

$$\text{recall} = \frac{TN}{FP+TN}$$

$$\text{precision} = \frac{TP}{TP+FN}$$

$$\text{G-mean} = \sqrt{\text{recall} \times \text{precision}}$$

Figure 45 The confusion matrix and the evaluation matrix

The TP-rate, also known as recall, is a measure that gives the percentage of the correctly classified positive samples. The precision defines a measure of exactness or the predicted positive cases that were correct, while the geometric mean, G-mean, is a measure that combines between the precision and the recall.

4.2.2 Imbalanced Data Description

In our experiments we will use the above described metrics to evaluate the performance of BBMO. A selected popular set of imbalanced datasets are described in Table 3. The selection was made to include different data sizes, types, imbalance ratio (IR) and different number of attributes. The datasets were retrieved from KEEL data repository [139] and UCI [140].

Table 3. Description of Dataset

Dataset Type	Name	IR	Samples	Attributes
Real World Datasets Balanced	Yeast	8.11	1484	8
	Ecoli	8.19	336	7
	Segmentation	6.01	2308	19
	Ionosphere	2.02	351	34
	Spambase	1.54	4601	57
	Wisconsin Diagnostic	3.21	198	32
Synthetic Imbalanced Datasets	Clover	5	600	2
	Subclass	5	600	2
	Paw	5	600	2

4.2.3 Experimental Results using BBMO

Let us examine the results of the BBMO minority oversampling technique described in chapter 3. First, BBMO-1 that uses the Lagrange multiplier values to extract the boundary samples was run against three popular datasets and their respective accuracy, recall, and the G-mean are tabulated in table 4.

Table 4. Performance of SVMs compared with BBMO-1: the linearly separable case

Data	SVM			LS-SVM			BBMO1+LS-SVM		
	Accuracy	Recall	G-mean	Accuracy	Recall	G-mean	Accuracy	Recall	G-mean
Spambase	0.91257	0.8758	0.8876	0.8858	0.7842	0.8462	0.9127	0.8896	0.8893
Wisconsin	0.9561	0.9339	0.9408	0.9543	0.8860	0.9362	0.9701	0.9483	0.9594
Ionosphere	0.8690	0.7301	0.8065	0.8578	0.6523	0.7769	0.8830	0.8005	0.8357

One can observe that the BBMO-1 with the LS-SVM performs better than LS-SVM with all the metric measures used. The performance is slightly better when compared with the SVM. Since the accuracy, recall, and G-mean all increase together, this means the boundary minority samples that were not classified correctly before have been correctly classified without any trade-off with the majority boundary samples. Thus, the bias is removed safely in the linear case for the datasets used. The performance of BBMO-1 was degraded when it was run against other datasets which were linearly inseparable. Because the boundary samples rely on the Lagrange multipliers, in the linearly non-separable case, the method fails as the means of the boundary samples do not necessarily lie on the boundary. On the other hand, especially

with the linearly non-separable case, BBMO-2 performs well on most of the datasets when it fails with the sub-class, the e-coli, and the ionosphere datasets by only recording

Table 5. Performance of SVMs compared with BBMO-1: Non-Linearly Separable Case

Data	SVM			LS-SVM			BBMO2+LS-SVM		
	Accuracy	Recall	G-mean	Accuracy	Recall	G-mean	Accuracy	Recall	G-mean
Paw	97.16	0.9300	0.9165	96.97	0.9000	0.9011	97.67	0.9800	0.9354
Subclass	94.5	0.86	0.8402	94.67	0.7600	0.8281	93.50	0.8100	0.8077
Clover	96.56	0.8750	0.9713	96.94	0.9000	0.8853	97.65	0.9375	0.9127
Yeast	95.35	0.8809	0.8384	93.74	0.7109	0.7862	96.13	0.9054	0.8818
E-coli	95.21	0.8072	0.8449	92.26	0.6145	0.7183	94.65	0.8672	0.8366
Segment	99.49	0.9747	0.9822	99.42	0.9697	0.9796	99.49	0.9747	0.9822
Ionosphere	95.1831	0.9062	0.9317	93.73	0.9363	0.9152	94.59	0.9600	0.9290
Spam	91.56	0.8852	0.8879	91.34	0.8598	0.8871	91.82	0.8830	0.8948
Wisconsin	95.42	0.9150	0.9376	94.90	0.9056	0.9303	95.43	0.9292	0.9383

higher recall values which may be an indication of a trade-off between the minority and the majority boundary class samples.

This is true since in any classification problem, new samples to be classified may or may not resemble the data employed during the prediction process. The new samples that are closer to the boundary have higher probability of being misclassified. In order to understand this phenomenon, we define the function that extracts for each of the test samples the closest inter-class boundary samples and their respective distances from each other using the kernel distance. The experiments show that BBMO-2 reduces the distance slightly for the minority samples as it adds synthetic minority samples at the boundary in the direction of the majority samples. This will increase the probability of the boundary incoming samples to be classified correctly. In figure 39, the test sample x_t which is halfway at the boundary and is approximately equidistant from both classes

is now closer to the minority class and will most probably be classified as a minority sample. This explains the slight improvement in both overall accuracy and the recall when compared with the SVM as few of the samples are being saved if they are close enough to the minority boundary samples. This also explains the trade-off between the recall and the overall accuracy in some of the cases in table 5. Figure 46 represents the average error of some of the performance metrics when LS-SVM is compared with the original SVM with and without BBMO. The results clearly show that with BBMO-2 LS-SVM results approaches the results of SVM and records on average slightly better results.

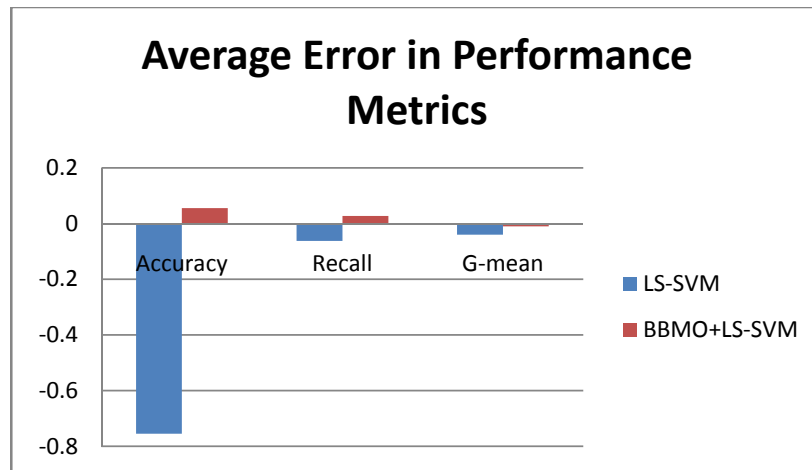


Figure 46 Average error in performance metrics when LS-SVM is compared with SVM with and without BBMO-2

We also compare the BBMO with SMOTE with some of the datasets. BBMO2 with LS-SVM again outperforms the SMOTE with the SVM in most of the cases where it fails with the e-coli and produces similar results with the skin segmentation dataset as shown in table 6.

Table 6. Comparison of BBMO-2 against SMOTE

	SMOTE+SVM		BBMO2+LS-SVM	
	Accuracy	G-mean	Accuracy	G-mean
Paw	97.50	0.9278	97.67	0.9354
Subclass	90.00	0.6851	93.50	0.8077
Clover	95.06	0.815	97.65	0.9127
Yeast	95.35	0.8384	96.13	0.8818
Ecoli	96.94	0.8449	94.65	0.8366
Segment	99.49	0.9822	99.49	0.9822
Spam	90.75	0.8808	91.82	0.8948

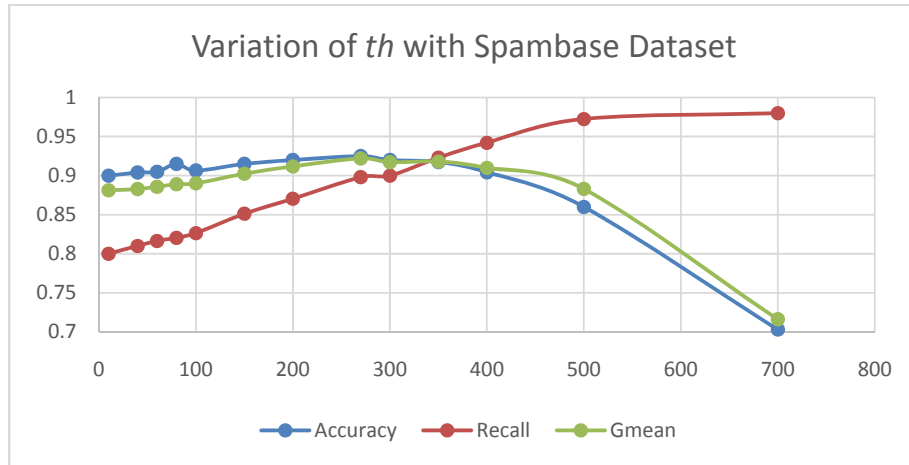


Figure 47 Metric value variation when number of synthetic data is varied

The threshold th_2 defines the number of samples chosen at the boundary and respectively the number of oversampled data forming the barrier. Figure 47 shows how the accuracy, recall and the G-mean vary with the variation of the number of samples chosen. Since the RBF kernel's sigma parameter choice affects the values in the kernel matrix, th_2 needs to be tuned. In general, th_2 was chosen $0.8 < th_2 < 1$ in all our experiments where sigma was of the order of tens. This ensured the selection of the closest samples. For the spambase dataset, th_2 was set to 0.998 which resulted in oversampling of around 300 samples. As shown in the figure, when the number of

synthetic samples varied between 300 and 400 samples a slight trade-off between the different metrics was produced. The figure also shows that when the number of oversampling of the minority using BBMO-2 increases, accuracy and G-mean decreased while the recall increased which means that the majority samples were being misclassified in expense of the minority. The same variation were obtained with the other datasets with different th_2 values as increasing its value resulted in the selection of only very few samples from the extreme boundary which didn't produce any enhancement in the performance. Thus, th_2 needs to be tuned correctly to select a good representative amount of boundary samples and subsequently a representative number of oversamples minority boundary samples.

4.3 Experiments on Seismic Textures

A plethora of seismic textures have been identified and related to lithological structures or other accumulations such as gas and oil. In this section we study briefly some of these textures and their respective Haralick Attributes. The images were extracted from the OpenDtect [141] software. Figure 48 shows a seismic section on which 8 different textures are marked with squares and labeled in capital letters referred to as windows. Window A represents a parallel rock bed which represents sedimentary rocks. Window F represents a bright spot which is an indication of oil and gas while E represents a gas escape channel. D represents an inclined parallel rock formations and H represents a chaotic fault.

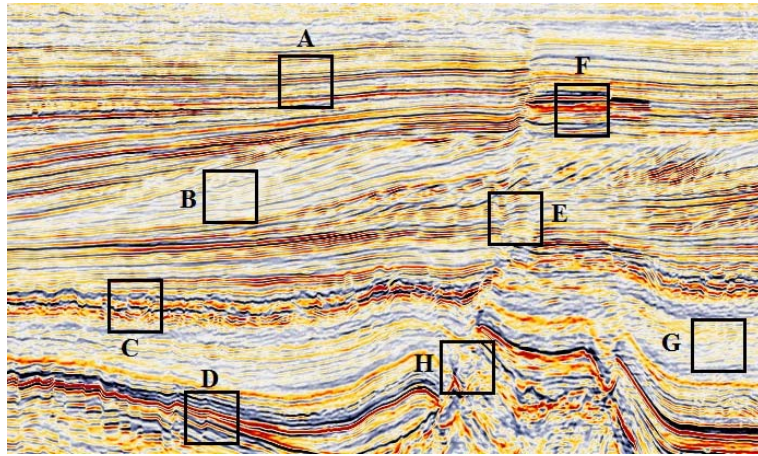


Figure 48Types of textures in a seismic slice

Figure 49 represents the attribute values recorded at the different sections considered in figure 48. The represented results are relative to the highest values recorded for each attribute. In particular, window F which represents a bright spot i.e. hydrocarbon accumulation, resulted in highest energy, lowest contrast, highest variance, and lowest entropy. These results validate Goa's results. Note that, our computations were done using the angular relationship of 0 degrees and a distance of 1.

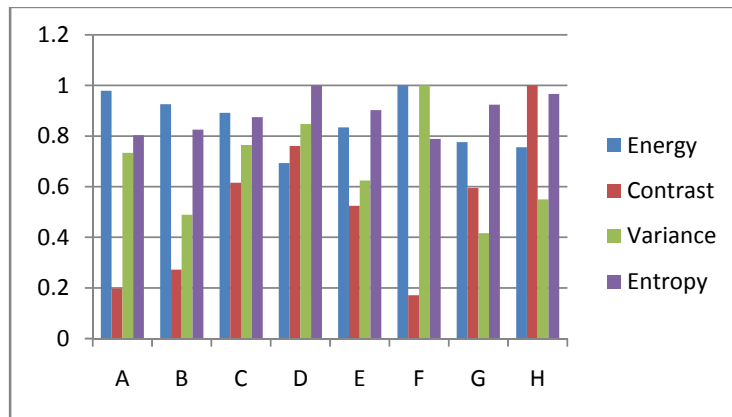


Figure 49Seismic texture attribute values at the considered windows in figure 48

The energy attribute of the seismic section in figure 48 is shown in figure 50. The bright spot is directly visible as the horizontal dark region as shown by the OpenDtect software.

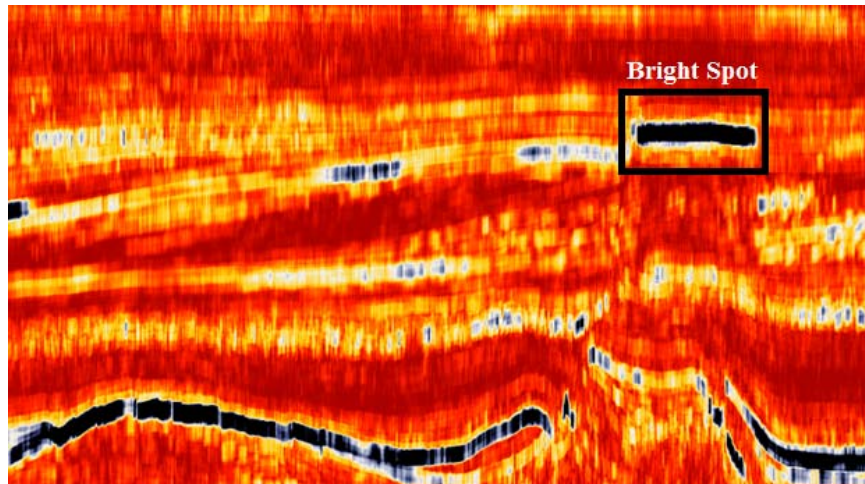


Figure 50The energy attribute computed in 3D fashion using OpenDtect Software

4.3.1 Performance of the Classifiers on the Seismic Data

In this section, we demonstrate the performance of LS-SVM and BBMO when trained and evaluated with a sample 3D seismic volume. The sample seismic volume extracted from the F3 demo data which contains an oil reservoir. This particular sub volume of data was used for training the classifiers serially. The volume contains 10 slices of 2D seismic images where a 10-fold cross validation was performed. Nine of the slices were used for training while the 10th slice was used for validation. The next figure shows the overall accuracy along with other imbalanced data performance.

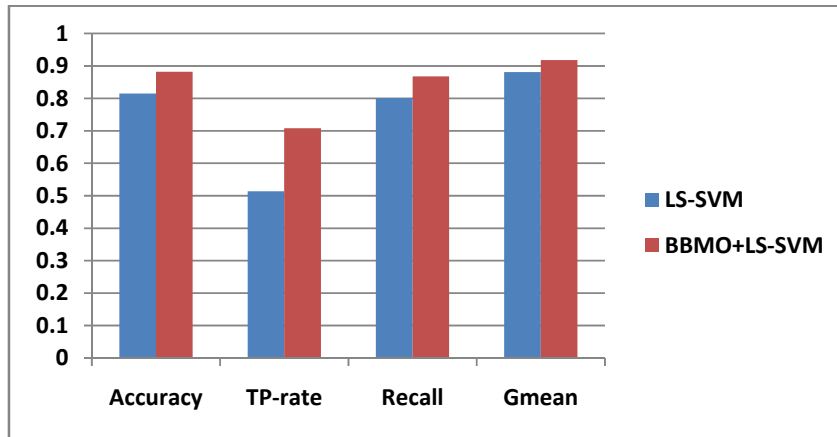


Figure 51 Performance of LS-SVM with BBMO

Since the data at hand is naturally imbalanced; the oil bearing locations are much smaller than the non-oil bearing locations, the classifier was fed BBMO-2 to improve the accuracy. As expected, all the performance metrics showed an enhancement and recorded higher results with the BBMO when compared with LS-SVM as shown in figure 51.

4.3.2 Volume Estimation and Visualization

The unit area of the F3demo data was estimated to be around 4m² per pixel in the retrieved images from the OpenDtect Software. Thus the estimated volume is around 64m³/1 pixel³. This unit volume is multiplied by the estimated pixel volume of the predicted oil site. The estimated volume was found to be around 1200 pixel³ in one section. If 10 sections are considered and the distance between the two slices is supposed to be d_z , then the estimated pixel volume is around 12000 d_z . In our stud, we assume d_z to be 1pixel resulting in 12000 pixel³. The overall volume is

$12000 \times 64 \text{m}^3 = 0.768 \text{ million m}^3$. The figure shows 12 seismic slices where oil and gas were predicted. In figure 52, the red surface indicates oil bearing sites in the volume.

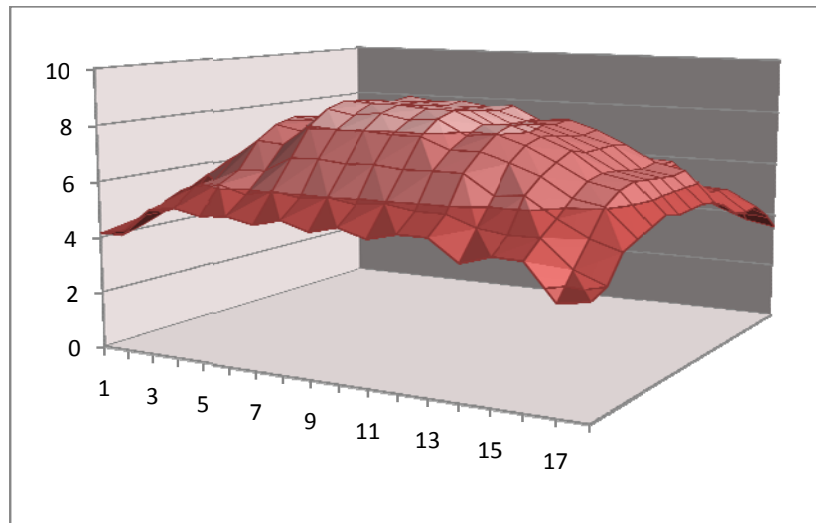


Figure 52 3D seismic prediction and volume reconstruction

4.4 Profiling of the Seismic Data Analysis Algorithm

Since the system is a distributed one, before distributing the tasks on the nodes of our distributed system, it is necessary to perform profiling of the algorithms to identify the operations or tasks that are computationally the most expensive. In this study the communication is not taken into consideration. The profiling of the seismic code design discussed earlier in this section was performed on an Intel core i7, 2GHz processor with 6GB RAM.

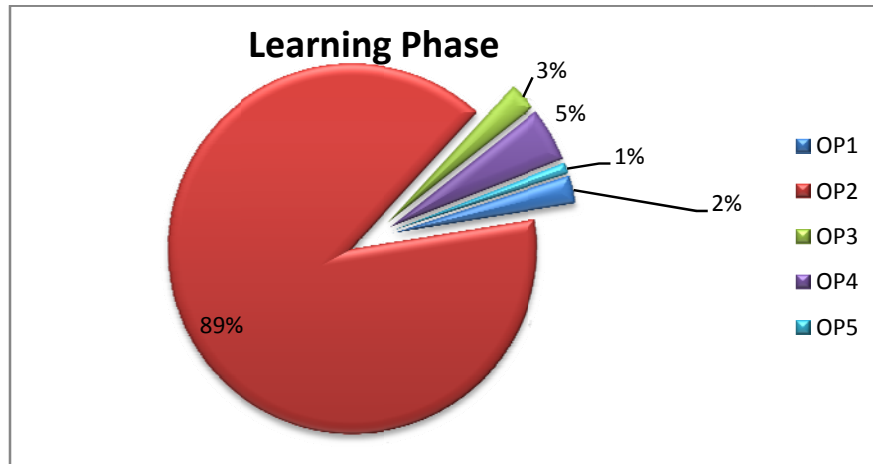


Figure 53 Profiling of the learning phase

Figure 53 shows the percentage of time spent in the five operations defined during the learning phase. We notice that operation 2 takes around 89% of the total time while the other operations together take 11%. Thus it was crucial to study operation. Figure 54 shows the profiling of operation2 which shows that the Haralick and the GLCM consume around 93% of the total time when the number of grey levels is 8. We also notice that GLCM takes more computational time as the number of grey levels is increased. The computation of the Haralick attributes will be less expensive in this case. Here, the results are based on an 8x8 window with 8 grey levels.

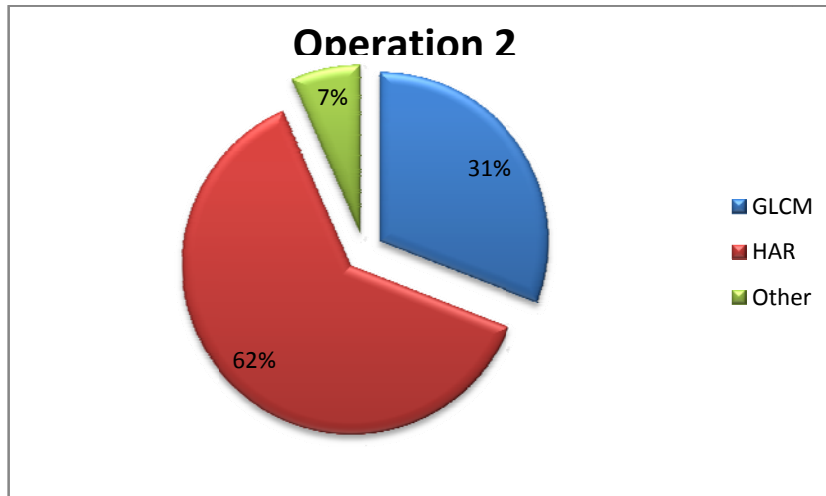


Figure 54 Profiling of the computationally most expensive operation of the learning phase

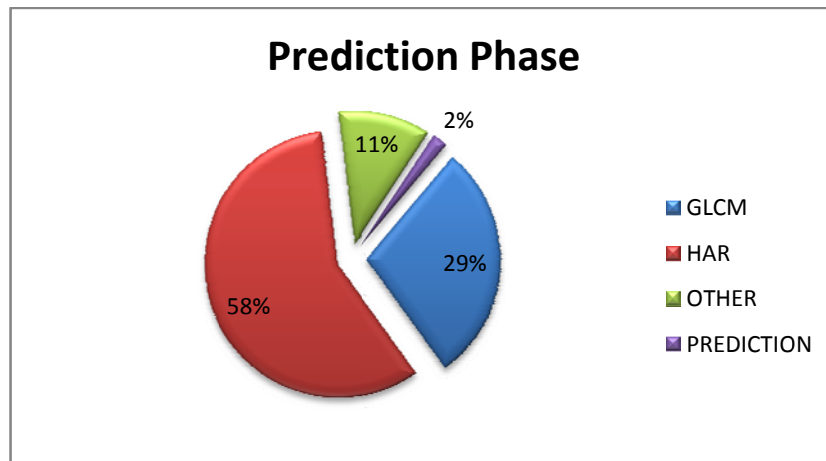


Figure 55 Profiling of the prediction phase

During the seismic analysis, the prediction phase will be the most commonly used phase as learning involves only a sub section of the volume while the prediction will be applied on large volumes under exploration. Thus it is crucial to identify the computationally most expensive operations in the prediction phase. Figure 55 shows

again the GLCM and the Haralick are the most expensive tasks during the prediction phase and distributing and accelerating them would produce faster computational times.

In the case of the distributed training of the classifier, the LS-SVM produces similar results. The distributed version of the BBMO was not studied in this paper since the learning phase is usually performed on a sub volume of the data whose dimension is much smaller than the data used for prediction since computationally the most expensive operations were found to be the extraction of the GLCM and the Haralick per window.

4.5 Experimental Results of the Seismic Application on Distributed Matlab

In this section, the effect of multithreading on the prediction phase using distributed Matlab on eight cores is highlighted; nevertheless similar results were recorded for the learning phase. Different data sizes were used and their respective computational times were computed. The volume sizes for typical image volumes under study ranges from 308 KB to 2.25 GB. Figure 56 shows that as the amount of data increases, the computational times for the prediction phase increase exponentially when run serially.

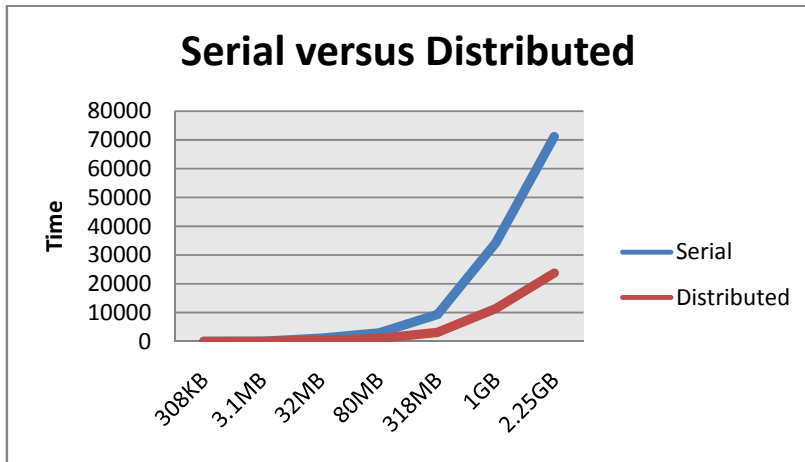


Figure 56 Computational cost of the prediction phase as data size increase: serial versus multithreaded

The distributed computations also increase exponentially as data is increased with speedups of up to three times as shown in figure 57. The computational costs of the distribution and the aggregation of the results diminish the speedup from the expected linear speedup to up to three times only.

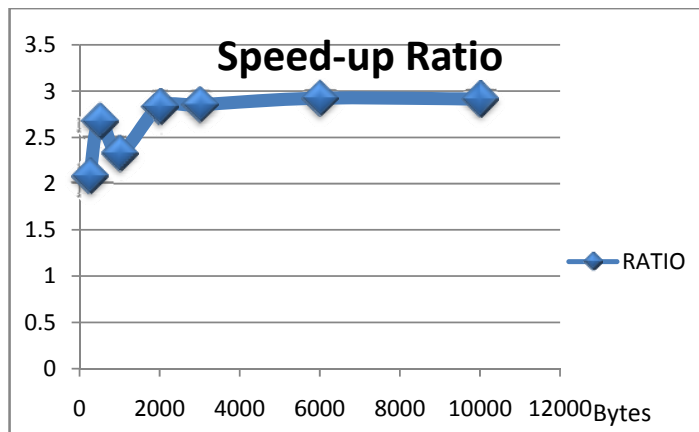


Figure 57 Speed-up due to multithreading

4.6 Seismic Analysis on RASSD with Acceleration

Experiments on the RASSD system of the designed application were performed by accelerating the GLCM and the Haralick attribute extraction [142]. The results show that, when no communication cost is included, speedups of the order 100x can be achieved for the learning phase.

While multithreading with eight cores achieved up to 3x speedup, the distributed hardware accelerated system using RASSD nodes can achieve speedups of more than 18x. Additionally, the effect of replacing our Ethernet connection with a 1Gbps one shows the distributed RASSD time with a 1Gbps Ethernet connection to the middleware client. The distributed system can achieve a speedup of more than 200x. Figure 58 summarizes the results for the prediction phase which takes 60,000 seconds (16.67 hours) to finalize when computed serially now takes around 3000 seconds (0.83 hours) only.

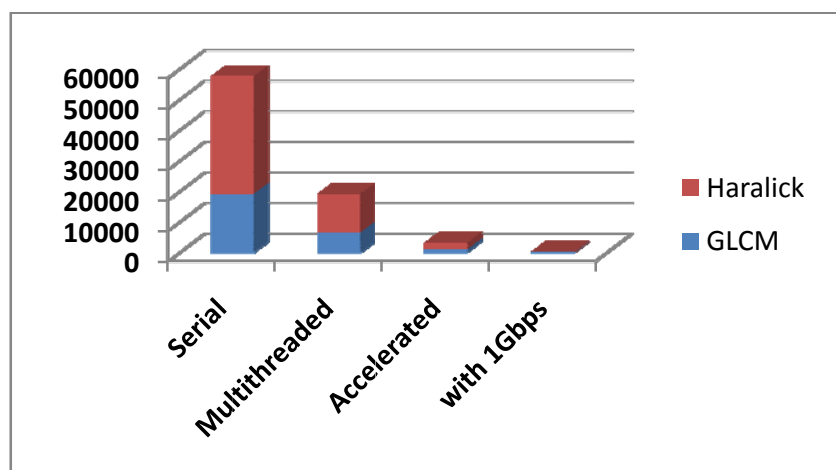


Figure 58 Haralick and GLCM operations' speedups

CHAPTER V

CONCLUSION AND FUTURE WORK

Seismic data analysis is a large scale and complex method which includes a plethora of techniques whose computational cost increases with the amount of data. Seismic volumes generally contain hidden information which can be indicative of different lithological structures and other information. The extraction of this information requires a complex analysis of the seismic data to retrieve the needed attributes. These attributes are classified into different classes and each may be used in the identification of a certain structure in the seismic data. Texture attributes were described and used in this paper to identify oil bearing sites. Naturally, oil bearing sites are relatively much smaller than the non-oil bearing sites which make the data used for training the classifier imbalanced. LS-SVM is known to produce biased separating hyper-plane when trained against imbalanced data. The Barricaded Boundary Minority Oversampling method was introduced to remove the bias of the LS-SVM and produce better results when compared with LS-SVM and relatively better or comparative results when compared to the SVM depending on the data. The development phase of BBMO highlights the idea in two versions. When applied on the seismic data, LS-SVM produces better results with LS-SVM. Finally, the application is reconstructed to fit a map/reduce based distributed environment. The application was designed to automatically read, extract windows, extract the GLCM and compute the Haralick attributes the matrices in the LS-SVM formulation, learning and the prediction phases all in a distributed fashion. The application was profiled to identify the computationally most expensive parts and

include these sections in the mapper function that is performed at each node in a distributed system. The reducer simply aggregated the results. The computations were further accelerated by Dr. MagedaSharafeddin on RASSD nodes and speedups of more than 200 times were recorded for the prediction phase.

The work presented in this paper can be further developed in the future in different ways. First of all, the classifier used in this paper is the LS-SVM which was further studied using the BBMO method described to solve the problem of the bias. The different parameters in BBMO can be studied to tune the results. Other classifiers may be employed and compared with these results to understand the impact of the classifier on seismic texture analysis.

As for the seismic data analysis, the aim in this paper is to train the data using oil bearing sites and later use the classifier to identify these sites in a 3D fashion. Other Haralick attributes coupled with other seismic attributes may be employed to generalize the method to identify any type of features in the seismic volume such as faults and gas chimneys. 3D GLCM with different orientation and distances may be also explored on this system. Also, the method can be generalized to be used for texture analysis of other types of 2D or 3D data such as in the medical field and satellite images. 4D data analysis can also be studied.

The computations in the RASSD node were done using a window size of 8x8 which can be later improved to include 16x16 and 32x32 window sizes. The whole design can be also implemented on the popular Hadoop framework since it was designed using the Map Reduce framework.

Finally, the overall distributed seismic data analysis flows described in this thesis can be developed. Different ways to distribute the load onto the distributed system may be studied and compared.

REFERENCES

- [1] Explaining Exploration and Production Timelines (Offshore). Energy API , 2011
Retrieved from
http://www.api.org/newsroom/upload/51073205_explaining_exploration_and_production_timelines_offshore1.pdf
- [2] Retrieved from <http://www.storagesearch.com/nstorart.html>
- [3] “Oil On My Shoes”, Electric Logs. Retrieved from
<http://www.geomore.com/electric-logs/>
- [4] N. Ezekwe, “Petroleum Reservoir Engineering Practice”, Prentice Hall, September 14, 2010.
- [5] Trap Types, Heavy Oil Science Centre, Retrieved
from <http://www.lloydminsterheavyoil.com/traptypes.htm>
- [6] Retrieved from www.sercel.com
- [7] “An Overview of marine seismic operations”, OGP, International Association of Oil and Gas Producers, 2011.
- [8] Maritimeportal.net , retrieved from: <http://maritimeportal.net/what-is-a-offshore-seismic-survey-vessel/>
- [9] Retrieved from <http://www.rag-austria.at/en/business-area/search/seismic.html>
- [10] ION, Retrieved from <http://www.iongeo.com/>
- [11] M. W. Norris, A.K. Faichney, SEG Y Data Exchange Format, Release 1.0, Society of Exploration Geophysicists, May 2002.
- [12] Ö. Yilmaz, “Seismic data analysis”, Society of Exploration Geophysicists, 2001.
- [13] M. Hall, Visual cross-plotting, 2011 retrieved from
<http://www.agilegeoscience.com/journal/tag/attributes?currentPage=3>
- [14] N. B. Rizvandi, A. J. Bolori, N. Kamyabpour, and A. Zomaya, “MapReduce Implementation of PrestackKirchoff Time Migration (PKTM) on Seismic Data,” IEEE, 2011, pp.86-91.
- [15] B.F. Rummerfield, “Reflection quality a fourth dimension”, Geophysics, vol. 19, 1954, pp. 684-694.
- [16] M.B. Dobrin, “Introduction to geophysical prospecting” , 3rd Ed.: McGraw-Hill, Inc., 1976.
- [17] R.E. Sheriff, L.P. Geldart, “Exploration seismology volume 1: History, theory, and data acquisition”. Cambridge University Press, 1989.
- [18] M.T. Taner, R.E. Sheriff, “Application of amplitude, frequency, and other attributes to stratigraphic and hydrocarbon exploration”, in Payton, C.E.,Ed., Seismic stratigraphy - Applications to hydrocarbon exploration: Am. Assn. Petr. Geol. Memoir 26, 1977, pp. 301-327.
- [19] J.D. Robertson, H.H. Nogami, “Complex seismic trace analysis of thin beds,” Geophysics, vol. 49, 1984, pp. 344-352.
- [20] W.J. Ostrander, “Plane Wave reflection coefficients for gas sands at non-normal angles of incidence,” SEG Expanded Abstracts, 1, 1982, pp. 216-218.

- [21] M. Bahorich, S. Farmer, "3-D seismic discontinuity for faults and stratigraphic features," The coherence cube: 65th Ann. Internat. Mtg., Soc.Expl. Geophysics., Expanded Abstracts, 1995, pp. 93-96.
- [22] J.H. Justice, D.J. Hawkins, G. Wong, "Multidimensional attribute analysis and pattern recognitions for seismic interpretation," Pattern Recognition, vol. 18, 1985, pp. 391-407.
- [23] R.B. Oliveros, B.J. Radovich, "Image-processing display techniques applied to seismic instantaneous attributes on the Gorgon gas field, North West Shelf, Australia," 67th Ann. Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, 1997, pp. 2064-2067.
- [24] R.O. Lindseth, "Synthetic sonic logs-a process for stratigraphic interpretation," Geophysics, 44, 1979, pp. 3-26.
- [25] S. Lancaster and D. Whitcombe, "Fast-track coloured inversion," SEG Expanded Abstracts, 19, 2000, pp. 1572-1575.
- [26] S.N. Dasgupta, M.R. Hong, P. La Croix, L.Al-Mana, G. Robinson, "Prediction of Reservoir Properties by Integration of Seismic Stochastic Inversion and Coherency Attributes in Super Giant Ghawar Field," SEG Abstracts, 2000.
- [27] A. Lau, J. Dai, A. Robinson, B. Flack, C.-C. Shih, R. Utech, N. Banik, Colored Inversion: Application In A Tertiary Basin Offshore China, Offshore Technology Conference, 2005.
- [28] M. Burianyk, S. Pickford, "Amplitude vs Offset and Seismic Rock Property Analysis: A Primer," CSEG Recorder November, 2000, pp.4-14.
- [29] X.G. Li, D.H. Han, J. Liu, D. McGuire, "Inversion of Sw and porosity from seismic AVO," SEG/Houston Annual Meeting, 2005.
- [30] K.J. Marfurt, "Robust Estimates of 3D reflector dip and azimuth," Geophysics, vol.71, no. 4, 2006, pp. 29-40.
- [31] Shuey, R.T., "A simplification of the Zoeppritz equations," Geophysics, 50, 1985, 609-614.
- [32] P. Connolly, "Elastic Impedance," The Leading Edge, 18, 1999, pp. 438-452.
- [33] Singh, Y., "Litho-facies detection through simultaneous inversion and principal component attributes," The Leading Edge, 26, 2007, 1568-1575.
- [34] A. Swisi, "Post- and Pre-stack attribute analysis and inversion of Blackfoot 3D seismic dataset", 2009.
- [35] K. Marfurt, "Robust estimates of 3D reflector Dip and Azimuth," Allied Geophysics Laboratories, 2006, pp. 29-40.
- [36] R.E. White, "Properties of instantaneous seismic attributes," The Leading Edge, 10, no. 7, 1991, pp. 26-32.
- [37] M. T. Taner, "Seismic Attributes", Rock Solid Images, Houston, U.S.A., CSEG, 2001.
- [38] A. E. Barnes "Seismic Attributes in your Facies", Landmark Graphics Corp., Englewood, Colorado, U.S.A., CSEG Recorder (History), 2001.
- [39] R. Haralick, K. Shanmugam, and I. Dinstein, "Textural Features for Image Classification", IEEE Trans. on Systems, Man and Cybernetics, SMC-3(6), 1973, pp. 610-621.

- [40] A. Eleyan, H. Demirel, "Co-occurrence based Statistical Approach for Face Recognition," IEEE, 2009, pp. 611-615.
- [41] L. Lopez, M. Moctezuma, and F. Parmiggiani, "Oil Spill detection using GLCM and MRF", IEEE, 2005, pp. 1781-1784.
- [42] P.J. Costianes, J.B. Plock, "Gray-level co-occurrence matrices as features in edge enhanced images," Applied Imagery Pattern Recognition Workshop (AIPR), IEEE 39th, 2010, pp. 1-6.
- [43] S. , V. Alexeev, "Application of texture attribute analysis to 3D seismic data," The Leading Edge, pp. 934-940, 2006,
- [44] X. Wang, N.D. Georganas, "GLCM Texture based Fractal Methods for Evaluating Fabric Surface Roughness," IEEE, 2009, pp. 104-107.
- [45] G. Gao, "Volume Texture extraction for 3D seismic visualization and interpretation," Geophysics, vol 68, no.4, 2003, pp. 1294-1302.
- [46] M.Yenugu, K.J. Marfurt, S. Matson, "Seismic Texture analysis for reservoir prediction and characterization," the Leading Edge, 2010, pp.1116-1121.
- [47] A. Chaddad, C. Tanougast, A. Dandache, A. Al Houseini, A. Bouridane, "Improving of Colon Cancer Cells Detection Based on Haralick's Features on Segmented Histopathological Images," ICCAIE, 2011, pp. 87-90.
- [48] Mryka Hall-Beyer, "GLCM Texture Tutorial", 2008, available at <http://www.fp.ucalgary.ca/mhallbey/tutorial.htm>
- [49] D.Subrahmanyam, P.H.Rao , "Seismic Attributes- A Review," 7th international Conference and exposition on petroleum physics, Hyderabad, 2008.
- [50] A. Barnes, "Too many seismic attributes?" CSEG Recorder, 2006, pp.41-45.
- [51] M. G. Orozco-del-Castillo, C. Ortiz-Alem'an, R. Martin, R. A'vila-Carrera, and A. Rodr'iguez-Castellanos, "Seismic data interpretation using the Hough transform and principal component analysis," Nanjing Geophysical Research Institute, 2011, pp.61-73.
- [52] N. Ru, Y.Jianhua, "An Attribute Reduction Methods Based on Rough Set and SVM and with Application on Oil-Gas Prediction," International Conference on Computer and Information Science, IEEE, 2007.
- [53] R.L. Chambers and J.M. Yarus , "Quantitative Use of Seismic Attributes for Reservoir Characterization," Quantitative Geosciences, Inc., 2002.
- [54] A. Bhatt, "Reservoir properties from well logs using neural networks," Adissertation. 2012.
- [55] C. J. Ferguson, A. Avu, N. Schofield, G. S. Paton, "Seismic analysis workflow for reservoir characterization in the vicinity of salt," Reservoir Geoscience and Engineering, First Break, vol.28, 2010, pp.107-113.
- [56] H. Hashemi, D. M. J. Tax, R. P. W. Duin, A. Javaherian, and P. de Groo, "Gas chimney detection based on improving the performance of combined multilayer perceptron and support vector classifier," Non Linear Processes Geophysics, vol. 15, 2008, pp. 863-871.
- [57] K.M. Tingdahl, A.H. Bril , P.F. de Groot, "Improving seismic chimney detection using directional attributes," J. Petrol.Sci.Eng., vol. 29, 2001, pp. 205-211.
- [58] C. Kliff, "Analyzing 8 key direct hydrocarbon indicators on post stack data," HIS, 2010, pp.1-9.

- [59] V. Vapnik, "The Nature of Statistical Learning Theory", Springer, 1995.
- [60] C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition", *Data Mining and Knowledge Discovery*, 2, 1998, pp. 121-167.
- [61] T. Joachims, "Text categorization with Support Vector Machines: Learning with many relevant features", *Machine Learning: ECML-98*, Lecture Notes in Computer Science Volume 1398, 1998, pp. 137-142.
- [62] V. Blanz, B. Schoelkopf, H.H. Buelthoff, C. Burges, V. Vapnik and T. Vetter, "Comparison of View-Based Object Recognition Algorithms Using Realistic 3D Models," In *ICANN96*, Bochum, Springer, 1996, pp. 251-256.
- [63] F. Steinke, B. Schoelkopf, V. Blanz, "Support Vector Machines for 3D Shape Processing," In *Computer Graphics Forum* 24(3), Eurographics, 2005.
- [64] A. Goh, S.H. Goh, "Support Vector Machines: Their use in geotechnical engineering as illustrated using seismic liquefaction data," *ScienceDirect Elsevier, Computers and Geotechnics* 34, 2007, pp. 410-421.
- [65] J. Platt, "Sequential minimal Optimization: a Fast Algorithm for Training Support Vector Machines", 1998.
- [66] J. Suykens, J. Vanderwalle, "Least Squares Support Vector Machines Classifiers," *Neural Processing Letters*, 1999, pp. 293-300.
- [67] T.-N. Do and F. Poulet, "Classifying one billion data with a new distributed SVM algorithm," in *4th IEEE International Conference on Computer Science, Research, Innovation and Vision for the Future*, Vietnam, 2006.
- [68] D. Pechyony, L. Shen, R. Jones, "Solving Large Scale Linear SVM with Distributed Block Minimization," *Akamai Technologies*, 2012.
- [69] H. P. Graf, E. Cosatto, L. Bottou, I. Durdanovic, V. Vapnik, "Parallel support vector machines: The cascade svm," In *Advances in Neural Information Processing Systems*, 2005.
- [70] E. Chang, K. Zhu, H. Wang, H. Bai, "PSVM: Parallelizing Support Vector Machines on Distributed Computers," *Google Research*, Beijing, China, 2007.
- [71] D. Wang, Y. Zhou, "Distributed Support Vector Machines: An overview," *IEEE*, 2012, pp. 3897-390.
- [72] T.G. Addair, D.A. Dodge, S.D. Rupert, "Large-scale Seismic Signal Analysis with Hadoop," *Computers and Geosciences* 66, 2014, pp. 145-154.
- [73] D. Brugger, "Parallel Support Vector Machines", WSI, 2006.
- [74] N.K. Alham, M. Li, S. Hamoud, Y. Liu, M. Ponraj, "A Distributed SVM for Image Annotation", *IEEE*, 2010.
- [75] N. Rizvandi, J. Taheri J., A. and Zomaya, "On using Pattern Matching Algorithms in map reduce Applications," 2011, *IEEE*, pp. 75-80.
- [76] C. Cortes, V. Vapnik. "Support-Vector Networks," *Machine Learning*, vol. 20, issue 3, 1995, pp. 273-297.
- [77] T. White, "Hadoop the Definitive Guide", O'Reilly, Third Edition, May 2012.
- [78] S. Ghemawat, H. Gobioff, S.T. Leung, "The Google File System," *Google, SOSP'03*, Bolton Landing, New York, USA, 2003.
- [79] R. Lämmel, "Google's mapreduce programming model - revisited," *Science of Computer Programming*, vol. 70, 2008, pp. 1-30

- [80] J. Dean, S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," Google, Inc., OSDI, 2004.
- [81] S. Dorward, R. Griesemer, S. Quinlan, "Interpreting the Data: Parallel Analysis with Sawzall," Google, Inc., 2005
- [82] Q.Z.Huang, L. Ye, M.Y. Yu, F.L. Wu ; R. Liang, " Information Integration Based Cloud Computing ," Network Computing and Information Security (NCIS), 2011 International Conference on, Vol. 1 , 2011, pp. 79 – 83.
- [83] A. Holmes, Hadoop In Practice, Manning Publications Co, 2012, pp. 7.
- [84] G. Caruana, M. Li, M. Qi, "A map reduce based Parallel SVM for Large Scale Spam Filtering," 8th International Conference on Fuzzy Systems and Knowledge Discovery, IEEE, 2011, pp. 2659-2662.
- [85] Y. Goto, R. Yamada, Y. Yamamoto, S. Yokoyama, H. Ishikawa, "SOM-Based Visualization for Classifying Large-Scale Sensing Data of Moonquakes ," P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on , 2013, pp. 630 – 634.
- [86] M. Kim, Y. Cui, H. Lee, H. Lee, C. Jeong, "A Hadoop-based Multimedia Transcoding System for Processing Social Media in the PaaS Platform of SMCCSE," Transactions on Internet and Information Systems, vol. 6, No. 11., 2012. pp. 2827-2848.
- [87] C.H. Lin, C.Y. Lee, S.P. Chien, "Digital Video Watermarking on Cloud Computing Environments," SDIWC. ,2013, pp. 49-53.
- [88] P. Singh and R.S. Chadha, "A Survey Digital Watermarking Techniques, Applications and Attacks," International Journal of Engineering and Innovative Technology (IJEIT), Volume 2, Issue 9, 2013, pp. 165-175.
- [89] N. Singh, S. Matele, S. Singh, "An Efficient Approach for Security of Cloud Using Watermark Technique," IJARCCCE., 2013, pp. 2814-2817.
- [90] L. Duan, D. Xu, I.W.H. Tsang, J. Luo, "Visual Event Recognition in Videos by Learning from Web Data," IEEE, Vol. 34, No 9, 2012, pp. 1667-1680.
- [91] F. OzgurCatak, M. ErdalBalaban. "CloudSVM : Training an SVM Classifier in Cloud Computing Systems," Turkish Journal of Electrical Engineering and Computer Sciences, 2013.
- [92] Sun Z, Fox G, " Study on Parallel SVM Based on MapReduce," International Conference on Parallel and Distributed Processing Techniques and Applications; 16-2012, pp. 495-561.
- [93] J.R. Quinlan, "Introduction of decision trees," Machine Learning, vol. 1, 1986, pp. 81-106.
- [94] Retrieved from www.mathworks.com/products/matlab
- [95] Retrieved from www.mathworks.com/products/distriben
- [96] N. Abbani , A. Ali , D. Al Otoom , M. Jomaa , M. Sharafeddine, H. Artail, H. Akkary, M. Saghier, M. Awad, H. Hajj, "A Distributed Reconfigurable Active SSD Platform for Data Intensive Applications", 13th IEEE International Conference on High Performance Computing and Communications, 2011.
- [97] D.V.Rao, S. Patil, N.A. Babu, v. Muthukumar, "Implementation and Evaluation of Image Processing Algorithms, on Reconfigurable Architecture using C-based

- Hardware Descriptive Languages,” International Journal of Theoretical and Applied Computer Science, vol. 1, no. 1, 2006, pp. 9-34.
- [98] H. Fu and B. Clapp, O. Mencer, and O. Pell, “Accelerating 3D Convolution using Streaming Architectures on FPGAs,” 79th Society of Exploration Geophysicists (SEG), 2009.
- [99] T. Nemeth, J. Stefani, W. Liu, O. Pell, R. Dimond, R. Ergas, “An implementation of the Acoustic wave equation on FPGAs,” 78th Society of Exploration Geophysicists (SEG), 2008.
- [100] T. Nemeth, J. Stefani, O. Pell, R. Ergas, “Design space analysis for the acoustic wave equation implementation on FPGA,” 70th European Association of Geoscientists and Engineers, 2008.
- [101] H.Fu, W. Osborne, B. Clapp, O. Pell, “Accelerating Seismic Computations on FPGAs from the perspective of Number Representations,” 70th European Association of Geoscientists and Engineers, 2008.
- [102] O. Pell, B. Clapp, “Accelerating Subsurface Offset Gathers, for 3D Applications,” 77th Society of Exploration Geophysicists (SEG), 2007. C. Brodley and M. Friedl, "Identifying and eliminating mislabeled training instances," Proc. of 13th National Conf. on Artificial Intelligence, 1996, pp. 799-805.
- [103] X. Wu, and P. Gopalan, Xilinx Next Generation 28 nm FPGA Technology Overview. s.l. : Xilinx White Paper, 2013.
- [104] Retrieved from www.mahout.apache.org/users/clustering/k-means-clustering.html
- [105] H.M. Hussein, K. Benkrid, C. Hong, H. Seker, “Highly Parameterized k_means Clustering on FPGA: Comparative Results using GPPs and GPUs,” IEEE, 2011.
- [106] X. Zhu, X.D. Wu and S.Chen., "Eliminating class noise in large datasets," Proceedings of the 20th ICML International Conference on Machine Learning, Washington D.C., 2003, pp. 920-927.
- [107] O. Barinova and v. Gavrishchaka, "Removal of confusing training samples as a generic mechanism to improve and diversify trading strategies discovered by boosting-based optimization," Proc. of CIEF, 2008.
- [108] H. Yin, H.B. Dong and Y.x.Li., "A Cluster-Based Noise Detection Algorithm," DBTA 2009, pp. 386-389.
- [109] C. M. Teng, "Correcting noisy data," In Proceedings of the International Conference on Machine Learning, 1999, pp. 239-248.
- [110] C. M. Teng, "Polishing blemishes: Issues in data correction," IEEE Intelligent Systems, 2004, pp. 34-39.
- [111] A. Hyvarinen and E. Oja, “Independent Component Analysis: algorithms and applications,” Neural Networks, vol. 13, no. 4-5, 2000, pp. 411-430.
- [112] J. H. Friedman, “Regularized Discriminant Analysis,” SLAK, 1988 .
- [113] R.C. Gonzalez, R. E. Woods, Digital Image Processing, Pearson, 2008.
- [114] Kotsiantis S., Kanellopoulos D., Pintelas P., Handling Imbalanced Datasets: a review. GESTS International Transactions on Computer Science and Engineering, vol. 30, 2006, pp. 1–11.

- [115] H. He ,E. Garcia , “Learning from Imbalanced Data. IEEE Transactions on knowledge and data engineering, Vol 21, 2009, pp. 1263-1284.
- [116] Y. Ou,H. Hung , Y. Oyang, “A Study of Supervised Learning with Multivariate Analysis on Unbalanced Datasets”, 2006, pp. 2201-2205.
- [117] R. Akbani, S. Kwek, N. Japkowicz, “ Applying support vector machines to imbalanced datasets,” In Proceedings of the 15th European Conference on Machine Learning, 2004, pp. 39-50.
- [118] N.V. Chawla, K. W. Bowyer, L.O. Hall, W. P. Kegelmeyer, “SMOTE: Synthetic Minority Over Sampling Technique,” Journal of Artificial Intelligence Research, vol. 16, 2002,pp. 321-357.
- [119] B. Ramentol, Y. Caballero, R. Bello, F. Herrera, “SMOTE-RSB: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced datasets using SMOTE and rough set theory,” Springer, 2011.
- [120] C. Bunkhumpornpat,K. Sinapiromsaran, C. Lursinsap, “Safe-LevelSMOTE: Safe Level Synthetic Over-Sampling TEchnique for Handling the Class Imbalanced Problem,” In Proceedings of PAKDD, Springer LNAI 5476, 2009, pp. 475–482.
- [121] H. Han, W.Y.Wang, B.H. Mao, “Borederline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning,” In Proceedings of ICIC, Springer, 2005, pp. 878-887.
- [122] J. Stefanowski, Sz. Wilk, “Improving rule based classifiers induced by MODLEM by selective pre-processing of imbalanced data,” In Proceedings of the RSKD Workshop at the ECML/PKDD Conference, 2007, pp. 54–65.
- [123] K. Veropoulos, C. Campbell , N. Cristianini, “Controlling the sensitivity of support vector machines,” In Proceedings of the International Joint Conference on Artificial Intelligence, 1999, pp. 55-60.
- [124] Y. Tang, Y., Y.Q. Zhang, Y. Q., N.V. Chawla, S. Krasser, “SVMs modeling for highly imbalanced classification,” IEEE Transactions on Systems, Man, and Cybernetics, Part B, 2009, pp. 281-288.
- [125] D.M.J.Tax,R.P.W. Duin, “Support vector domain description,” In Pattern Recognition Letters, vol. 20, 1999, pp. 1191-1199.
- [126] A. Kowalczyk, B. Raskutti, “One class svm for yeast regulation prediction,” SIGKDD Exploration Newsletters, vol. 4, no. 2, 2002, pp. 99-100.
- [127] B. Scholkopt, J. C. Platt , J. Shawe-Taylor , A.J. Smola, R. C. Williamson,“Estimating the support of a high dimensional distribution,” Neural Computation, vol. 13, 2001, pp. 1443–1471.
- [128] L. Zhuang, H. Dai , “Parameter Optimization of Kernel-based One-class Classifier on Imbalanced Learning,” Journal of Computers, vol. 1, 2006, pp. 32–40.
- [129] T. Imam ,K. Ting , J. Kamruzzaman, “z-svm: An SVM for improved classification of imbalanced data,” In Proceedings of the 19th Australian joint conference on Artificial Intelligence: advances in Artificial Intelligence, Springer-Verlag, 2006, pp. 264-273.
- [130] X. Wang, S.Matwin, N.Japkowicz, X. Liu, “Cost-Sensitive Boosting Algorithms for Imbalanced Multi-instance Datasets,”Canadian Conference on AI, 2013,pp. 174-186.

- [131] P. Li, K. Chan, W. Fang, "Hybrid kernel machine ensemble for imbalanced data sets," In Proceedings of the 18th International Conference on Pattern Recognition, IEEE Computer Society, 2006, pp. 1108-1111.
- [132] N. Ajeeb, A. Nayal A, M. Awad M., "MinSVM for Linearly Separable and Imbalanced Datasets," International Joint Conference on Neural Networks (IJCNN), TX, 2003.
- [133] Wu G., Chang E., "Adaptive feature-space conformal transformation for imbalanced-data learning," In Proceedings of the 20th International Conference on Machine Learning, , 2003, pp. 816-823.
- [134] Wu G., Chang E., "Class-boundary alignment for imbalanced dataset learning," In Proceeding of the International Conference on Machine Learning: Workshop on Learning from Imbalanced Data Sets, 2003, pp. 49-56.
- [135] Wu G., Chang E., "KBA: Kernel boundary alignment considering imbalanced data distribution," IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, 2005, pp. 786-795.
- [136] Li P., Chan K., Fang W., "Hybrid kernel machine ensemble for imbalanced data sets," In Proceedings of the 18th International Conference on Pattern Recognition, IEEE Computer Society, 2006, pp. 1108-1111.
- [137] Wu G., Chang E., "KBA: Kernel boundary alignment considering imbalanced data distribution," IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, 2005, pp. 786-795.
- [138] J. L'opez, K. De Brabanter, J.R. Dorronsoro and J.A.K. Suykens, "Sparse LS-SVMs with L0-norm minimization", ESAN, 2011, pp.189-194.
- [139] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, "KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework," Journal of Multiple-Valued Logic and Soft Computing 17:2-3, 2011, pp. 255-287.
- [140] K. Bache, M. Lichman, UCI Machine Learning Repository [<http://archive.ics.uci.edu/ml>]. Irvine, CA: University of California, School of Information and Computer Science.2013.
- [141] Available at <http://opendtect.org>

Do not include this page