Test Case Suite Reduction of High Dimensional Data by Automatic Subspace Clustering

Bhawna Jyoti, Aman Kumar Sharma

Abstract— Mostly, testing techniques are designed for data which are having low dimensional space and less intention is paid to the testing of high dimensional data. In this paper, data undergoes a process of dimensionality reduction by principal component analysis (PCA) which leads to the automate subspace clustering of data. The combination of distributed based approach and coverage based approach is used to test the test cases sampled from each cluster formed. The contribution of this paper is related to the dimensionality reduction as well as test case suite reduction by discovering patterns in software testing in a rigorous manner.

Keywords: Dimensionality reduction using PCA, clustering, the test suite minimization.

I. INTRODUCTION

In today's large-scale software systems, there is great attention paid on the test (suite) maintenance. As a software system spread its dimensions, its test suites need to be updated as well as maintained to verify new or modified functionality of the software. *Software testing* is a critical activity that requires large amounts of test cases to test any new or modified functionality within the program. Such technique attempts to find a minimal subset of test cases which satisfy all the testing requirements as the original set does and is commonly known as *test suite reduction* or *test suite minimization* [7]. A test case is a well documented procedure designed to test the functionality of a feature in the system. It is a collection of test cases that are intended to be used to test a software program to show that it has some specified set of behaviors [7].

In large databases, an object (data record) typically has dozens of attributes and the domain for each attribute can be large. So, it is not meaningful to look for clusters in such a high dimensional space as the average density of points anywhere in the data space is likely to be quite low. Principal Component Analysis which is a standard technique used for data reduction in statistical pattern recognition. The principal component analysis or Karhunen-Lo`eve (KL) transformation is used to project *n*-dimensional points to *k*-dimensional points which is optimal way to give a new set of orthogonal axes, containing a linear combination of the original ones. For each small entry in the matrix, the corresponding vectors may be eliminated and a lower dimensionality space is obtained [2][11][12].

In this paper, we present a way of data reduction technique using Principal Component Analysis, automate subspace clustering of data which is presented in the literature [1][2] and sampled data is tested as well as method of test case suite reduction is applied.

The structure of the paper is as follows:
Section II presents related concepts of proposed work including formal definition of the test suite minimization problem ,need of software clustering techniques and a method Principal Component Analysis to reduce high dimensional data into low dimensional profile space. Section III presents proposed algorithmic steps for testing software containing high dimensional data. Section IV describes explanation of the algorithmic steps and section V describes conclusion.

II. BACKGROUND AND RELATED CONCEPTS

The size and complexity of industrial strength software systems are constantly increasing. This means that the task of managing a large software project is becoming even more challenging, especially in light of high turnover of experienced personnel. Software clustering approaches can help with the task of understanding large, complex software systems by automatically decomposing them into smaller, easier-to-manage subsystems[1]. Data mining applications place special requirements on clustering algorithms including: the ability to find clusters embedded in subspaces of high dimensional data, scalability, end-user comprehensibility of the results[2][3].

The principal component analysis or Karhunen-Lo`eve (KL) transformation is the optimal way to project *n*-dimensional points to *k*-dimensional points such that the error of the projections (the sum of the squared distances) is minimal[2]. Testing activity is performed to provide confidence that changes do not harm the existing behavior of the software .Test suites tend to grow in size as software evolves, often making it too costly to execute entire test suites. The test suite reduction techniques significantly reduce the size of the test suites[3].The requirement matrix is mapped to form mathematical equations and genetic algorithm is used to derive a representative set to eliminate redundant test cases[4]. The related concepts are discussed as follows:

A. The test suite minimization problem:

The first formal definition of test suite reduction problem introduced in 1993 by Harrold et al. [9] as follows: Given a test suite T , $\{t1, t2,..., tm\}$, from m test cases and $\{r1,r2,..., rn\}$ is set of test requirements that must be satisfied in order to provide desirable coverage of the program entities and each subsets $\{T1, T2,..., Tn\}$ from T are related to one of ris such that each test case tj belonging to Ti satisfies ri, find minimal test suite T' from T which satisfies all ris covered by original suite T.

B. Clustering

Manuscript Received on March 2014.

Bhawna Jyoti, Computer Science Department MM University, Solan, India.

Aman Kumar Sharma, Computer Science Department, Himachal Pradesh University ,Shimla, India.

The process of dividing a dataset into mutually exclusive group such that the members of each group are as "close" as possible to one another, and different groups are as "far" as possible from one another, where distance is measured with respect to all available variables. Clustering is an unsupervised learning process: no predefined classes. Cluster analysis is finding similarities between data according to the characteristics found in the data and grouping similar data object into clusters. A good clustering method will produce high quality clusters with high intra-class similarity and low interclass similarity [3].

Khalilian and Parsa [10] proposed Bi-criteria test suite reduction with cluster analysis of execution profiles. They combined the two general techniques called distribution-based and coverage-based techniques to construct full coverage reduced test suites with minimum overlap in the execution profiles. Coverage based techniques uses def-use pair criterion for the selection of test cases because such test cases cover execution paths which may contain faults. Distribution based techniques clusters the test cases on the basis of their execution profiles and can be described by two methods: cluster filtering and failure pursuit.

C. Principal Component Analysis

It involves feature selection in which data space is transformed into feature space which has exactly the same dimension as the original data space .The transformation is designed in such a way that the data set may be represented by reduced number of effective features, the data set undergoes a dimensionality reduction [14].

Basic Data Representation

Let X denote m-dimensional random vector which contain data set x denoting the realization of the random vector x. Let $\lambda_1\ \lambda_2\ \lambda_3\ \ldots\ \lambda_l$ denotes the largest *l* Eigen values of the correlation matrix R. With m possible solution for the unit vector q, there are m possible projection of the data vector x to be considered

$$a_j = q_j^t x = x^t q_j \quad \text{and} \quad j=1,2,3\ldots m \qquad (1)$$

Where aj are projection of x onto the principal directions represented by unit vector v_j .The a_j are called principal component of the data vector x.

To reconstruct the original data vector x exactly from projection a_j , the set of projection $\{ a_j | j = 1,2,3 \ldots . m \}$ is combined into a single vector,

$$a = [a_1\ a_2 \ldots \ldots a_m]^t$$
$$= [x^t q_1, x^t q_2 \ldots \ldots x^t q_m]^t$$
$$= Q^t x \qquad (2)$$

This equation can be re-constructed as

$$X=Qa$$
$$\sum_{j=1}^{m} a_j q_j \qquad (3)$$

By approximating the data vector x after l terms ,we have

$$\hat{x} = \sum_{j=1}^{l} a_j q_j$$

$$= [q_1\ q_2 \ldots q_l] \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix}, l \leq m \qquad (4)$$

Given the original data vector x, we use equation (1) to compute the set of principal components retained in eq(4).

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix} = \begin{bmatrix} q_1^t \\ q_2^t \\ \vdots \\ q_l^t \end{bmatrix} .x,\ l \leq m \qquad (5)$$

The linear projection of this equation from R^m to R^l (the mapping from the data space to feature space) represents encoder for the approximate representation of data vector x. Encoding of input vector into the set of vectors containing principal components is given below:

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \Longrightarrow \begin{bmatrix} q_1^t \\ q_2^t \\ \vdots \\ q_l^t \end{bmatrix} \Longrightarrow \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix} \qquad (6)$$

The mapping from the feature space back to data space from R^l to R^m represent the decoder for the approximate reconstruction of original data vector x. Decoding of set of vectors containing principal components into reconstructed data vectors is given below:

$$\begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix} \Longrightarrow [q_1\ q_2 \ldots q_l] \Longrightarrow \begin{bmatrix} \widehat{x_1} \\ \widehat{x_2} \\ \vdots \\ \widehat{x_l} \end{bmatrix} \qquad (7)$$

The approximation error vector e equal the difference between the original data vector x and the approximating data vector $\hat{x}$

$$e = x - \hat{x} \qquad (8)$$

Substituting eqs (3) and (4) in (8) yields

$$e = \sum_{j=l+1}^{m} a_j q_j \qquad (9)$$

The error vector e is the orthogonal to the approximating data vector $\hat{x}$.In other words , the inner product of $\hat{x}$ and e is zero. This property is shown by equation (4)and (9).

$$e^t \hat{x} = \sum_{i=l+1}^{m} a_i q_i^t \sum_{j=1}^{l} a_j q_j$$
$$= \sum_{i=l+1}^{m} \sum_{j=1}^{l} a_i a_j q_i^t q_j$$
$$e^t \hat{x} = 0 \qquad (10)$$

The eq(10) is known as principle of orthogonality as

it fulfills the following condition:

$$q_i^t q_j = \begin{cases} 1, & j = i \\ 0, & j \neq i \end{cases}$$

The total variance of m components of data vector x is

$$\sum_{j=1}^{m} \sigma_j^2 = \sum_{j=1}^{m} \lambda_j$$

Where σ_j^2 is the the variance of j^{th} principal component a_j.

The total variance of the *l* elements of the approximating vector $\hat{x}$ is

$$\sum_{j=1}^{l} \sigma_j^2 = \sum_{j=1}^{l} \lambda_j$$

The total variance of (*l*-*m*) elements in the approximating vector x- $\hat{x}$ is

$$\sum_{j=l+1}^{m} \sigma_j^2 = \sum_{j=l+1}^{m} \lambda_j$$

The Eigen values λ_{l+1}…….λ_m are the smallest (m-*l*) Eigen values of the correlation matrix R, they corresponds to terms discarded from the expansion of eq.( )used to construct the approximating vector which leads to effective dimensionality reduction.

III. PROPOSED WORK

Most clustering methods are designed for clustering low-dimensional data and encounter high challenges when the dimensionality of data grows really high say over 10 dimensions or even thousands of dimensions for some task. This is because with the increase of dimensions, data in the irrelevant dimensions may produce much noisy data and it would be tough to discover the real clusters. When dimensionality increases data become sparse because data points are located at different dimensional subspaces. To overcome this difficulty, a feature transformation method is used which transforms the data onto a smaller space preserving the original relative distance between the objects.

Step:1. High dimensional data is reduced to low dimensional data using a feature extraction technique called PCA(principal component analyses).

Step:2. After automatic subspace clustering of high dimensional data ,distribution based techniques are used to select test cases based on the dissimilarity metric in the multi-dimensional profile space. The purpose of automatic cluster analysis is to partition the population such that objects must have similar attributes in the same cluster.

After clustering, test cases are sampled from each cluster (one test case per cluster).

Step:3. Selected test cases can exercise the whole program under test where redundancy is removed and test suite reduction is done.

A. Terminology used in the proposed algorithm:

1**.** Define:Requirement set : set of coverage requirements for minimization: *r1,r2,…,r*$_n$.

2. Input**:**test cases set:*t1,t2,…,tm*: all test cases present in the test pool

coverage[*m,n*]:matrix representation describing coverage of each test case.

Set value TRUE for covered and FALSE for uncovered cases.

clusters: a[*1..k*] of cluster instances, each containing similar test cases according to clustering property.

3 Output:*Reduced set*: a reduced suite of test cases from the test pool.

4. Declare: *nextTest*: one of test cases.

currentind: index of currently processing cluster. *counter*: a[1..n] of Boolean number, initial value will be **FALSE.**

LIST: list of *ti's*

Cardinality(): returns value to show the cardinality of a set.

Sortarray(): sorts the input array.

B. Algorithmic Steps

algorithm TSReduction
begin
currentind := 0; // initialization
Sortarray(*clusters*,*ascending*); // sorts the input array of clusters ascending or descending.
while there exist *ri* s.t. *counter*[*i*] == **FALSE do**
if *currentind* == *k* **then** *currentind* := 0;//to start from the first cluster.
LIST := all *tj* ε *clusters*[*currentind*];
if *Cardinality*(LIST) == 1 **then** *test* := *t* ε *LIST*;
if there exists *rj* ε requirements where *coverage*[*test,rj*] == **TRUE** and counter[*j*] == **FALSE then** *nextTest* := *test*;
else
currentind = *currentind* + 1;
continue;
endif
else *nextTest* := SelectTest(*LIST*);
endif
if *nextTest* ≠ 0 **then**
RS := *RS* U{*nextTest*};
foreach *rj* ε requirements where *coverage*[*nextTest,rj*] == **TRUE do**
counter[*i*] := **TRUE**;
endfor
endif
currentind := *currentind* + 1;
endwhile
return *RS*; **end** TSReduction

function SelectTest(testcaseset)
declare: /*This function selects the next test case to be in RS*/
n: the number of test cases in the *testCaseSet*
numUnmarked: array[1..*n*] of the number of unmarked requirements that each test case in testCaseSet covers
numCovered: array[1..*n*] of the number of requirements that each test case in testCaseSet covers
testCase: selected test case from test case set, initially 0
testList1, *testList2*: list of *ti's*
begin
foreach *ti* in *testCaseSet* **do** compute *numUnmarked*[*i*], the number of unmarked requirements *rj* from requirements that *ti* covers;
testList1 := all *ti* from *testCaseSet* for which *numUnmarked*[*i*] is the maximum;
if *Cardinality* (*testList1*) ≠ 0 **then**
if *Cardinality* (*testList1*) == 1 **then** *testCase* := the test case in *testList1*;
else
foreach *ti* in *testList1* **do** compute *numCovered*[*i*], the number of requirements *rj* that *ti* covers;
testList2 := all *ti* from *testList1* for which *numCovered*[*i*] is the maximum;
if *Cardinality* (*testList2*) ≠ 0 **then**
if *Cardinality* (*testList2* == 1) **then** *testCase* := the test case in *testList2*;
else *testCase* := any test case in *testList2*;
endif;
endif;
endif;
endif;
return *testcase* , end *testcase*

IV. EXPLANATION

We have proposed a new approach for test suite minimization of software's containing high dimensional data as input domain to the testing modules. The aim of this research methodology is to reduce test maintenance cost and ensuring the integrity of test suites by detecting redundant test cases. Although, all the previous test suite reduction techniques could significantly reduce the size of test suites, but there is no attention paid to resolve two issues simultaneously:- one major issue is how to deal with profile space containing large dimensional data using PCA(Principal Component Analysis) which automate the data into subspace clusters and second major issue deals with collaboration of coverage based techniques & distributed based techniques.

The traditional software testing is used to achieve maximum code coverage[10] which exercises the whole program under test. But as mentioned [10],code coverage alone is not sufficient for selecting test cases and fault detection effectiveness is of great importance.

In Distribution based techniques, there is a use of dissimilarity metrics in the multidimensional profile space[8][10] [12].These techniques are capable of determining similar test cases by means of clustering analysis within a test suite to exercise sampled test cases from clusters to reduce redundancy. But these techniques do not necessarily provide full coverage of the executions.

So, we combined these techniques to set a platform to form full coverage of reduced test suits with minimum overlap in the execution profile of the including test cases. In our proposed algorithm, an optimal way is opted for subspace clustering for dimensionality reduction and then to reduce test maintenance cost ,an integrated approach of distributed and coverage based technique is followed to eliminate redundant test cases. The following chart shows the comparison of coverage based techniques ,distributed based techniques and integrated approach of distributed schemes & coverage based techniques after clustering.

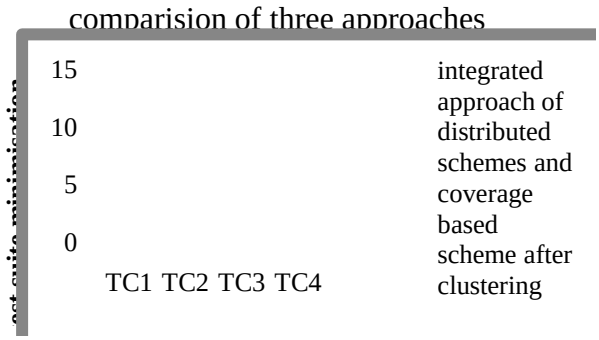

Fig:1

V. CONCLUSION

This paper presented an optimal way of reducing high dimensional data into low profile space by the application of PCA which leads to overcoming the difficulties of applying suitable testing techniques to high dimensional data. In this paper there is an outline of the most important research challenge of test suite case reduction to avoid redundancy .

REFERENCES

[1] Mark Shtern and Vassilios Tzerpos," Clustering Methodologies for Software Engineering "in Hindawi Publishing Corporation Advances in Software Engineering ,Volume 2012, Article ID 792024, 18 pages, doi:10.1155/2012/792024.

[2] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopulos, Prabhakar Raghavan," Automatic Subspace Clustering Of High Dimensional Data, "in Data Mining and Knowledge Discovery, 11, 5–33, 2005, Springer Science ,Inc. Manufactured in The Netherlands.

[3] Lilly Raamesh, Lilly Raamesh," An Efficient Reduction Method for Test Cases,"in International Journal of Engineering Science and Technology",Vol. 2(11), 2010, 6611-6616.

[4] Mamta Santosh,Rajvir Singh,"Test Case Minimization By Generating Requirement Based Mathematical Equations,"in International Journal of Engineering Research & Technology (IJERT), Vol. 2 Issue 6, June – 2013

[5] Rajvir Singh and Mamta Santosh," Test Case Minimization Techniques : A Review," in International Journal of Engineering Research & Technology (IJERT), Vol. 2 Issue 12, December – 2013.

[6] Saeed Parsa and Alireza Khalilian," A Bi-objective Model Inspired Greedy Algorithm for Test Suite Minimization," FGIT '09 Proceedings of the 1st International Conference on Future Generation Information Technology,2009.

[7] Saeed Parsa and Alireza Khalilian," On the Optimization Approach towards Test Suite Minimization," in International Journal of Software Engineering and Its Applications,Vol. 4, No. 1, January 2010.

[8] Shin Yoo & Mark Harman," TR-09-09: Regression Testing Minimisation, Selection and Prioritisation - A Survey," in Technical report TR-09-09, Department of Computer Science, King's College London, 2009.

[9] Mary Jean Harrold ,Rajiv Gupta And Mary Lou Soffa," A Methodology for Controlling the Size of aTest Suite," 1993 ACM 1049 -331X/93 /'O7OO-O27O $01.50 Julv 1993, Pages 270–285.

[10] Alireza Khalilian and Saeed Parsa," Bi-criteria Test Suite Reduction by Cluster Analysis of Execution Profiles," in International Federation for Information Processing, 2012, CEE-SET 2009, LNCS 7054, pp. 243–256, 2012.

[11] Tajunisha1 and Saravanan "An efficient method to improve the clustering performance for high dimensional data by Principal Component Analysis and modified K-means,"in International Journal of Database Management Systems (IJDMS), Vol.3, No.1, February 2011.

[12] Imola K. Fodor "A survey of dimension reduction techniques" UCRL-ID-148494, May 9, 2002.

[13] Siripong Roongruangsuwan, jirapun Daengdej," Test Case Prioritization Techniques" in Journal of Theoretical and Applied Information Technology,© 2005 - 2010 JATIT & LLS.

[14] Book: An Introduction to Neural Networks, *Simon Haykin*, Prentice Hall, 1999.