Design And Implementation Of Tool For Detecting Anti-Patterns In Relational Database

Gaurav Kumar, Rahul Kumar Yadav, Sanjay Bhutungru

Abstract: Anti-patterns are poor solution to design and im-plementation problems. Developers may introduce anti-patterns in their software systems because of time pressure, lack of understanding, communication and or-skills. Anti-patterns create problems in software maintenance and development. Database anti-patterns lead to complex and time consuming query process-ing and loss of integrity constraints. Detecting anti-patterns could reduce costs, efforts and resources. Researchers have proposed approaches to detect anti-patterns in software development. But not much research has been done about database anti-patterns. This report presents two approaches to detect schema design anti-patterns in relational database. Our first approach is based on pattern matchingwe look into potential candidates based on schema patterns. Second approach is a machine learning based approach we generate features of possible anti-patterns and build SVMbased classifier to detect them. Here we look into these four anti-patterns a) Multi-valued attribute b) Nave tree based c) Entity Attribute Value and d)Polymorphic Association . We measure precision and recall of each approach and compare the results. SVM-based approach provides more precision and recall with more training dataset.

Index Terms: Anti-pattern, RDBMS, Design Pattern, SVM, SVMLearn.

I. INTRODUCTION

Developers continuously evolve software systems to imple-ment and adapt to new customers needs, as well as to fix bugs. Due to the time-to-market, lack of understanding, and the developers experience, developers cannot always follow standard designing and coding techniques, i.e., design pat-terns. Design patterns are good solutions to recurring design problems, conceived to increase reuse, code quality, code readability, resilience to changes and above all, maintainability. Consequently, anti-patterns creep up in software systems. Antipatterns are poor solutions to recurring design and imple-mentation problems. They are generally the result of misuse of the design patterns. Like many software anti-patterns relational database also has many anti-patterns like schema design anti-patterns, phys-ical design anti-patterns. These anti-patterns are a result of not complete understanding of problem, lack of integrity constraints in the design, improper way to store data, lack of dependencies in schema. Query processing on any relational database depends on schema design of the database. If schema design is good query processing will be very fast and simple. Otherwise we need to write a very complex query which take more time to process data. Also data integrity is a key feature of relational database. A bad schema design badly affects data integrity. A good Schema design considers all possible integrity constraints like domain integrity, referential integrity, entity integrity to preserve data integrity. Thus, it is important to know if your schema design is optimal or not. Here we are looking into four types of anti-patterns related to schema design of relational databases. These are multi-valued attribute anti-pattern, nave tree anti-pattern, entity attribute value anti-pattern and polymorphic association anti-pattern.

- *Gaurav Kumar; NIT Warangal, Email: gaurav.sachin.007@gmail.com*
- *Rahul Kumar Yadav; NIT Warangal, Email: rahulk@gmail.com*
- *Sanjay Bhutungru; NIT Warangal, Email: sanjaybhu@gmail.com*

We develop two approaches to detect these anti-patterns from relational database. Our approach is based on machine learning. We use support vector machine (SVM) as a classifier. SVM have been applied in various areas, e.g., bioinformatics and object recognition. SVM is a recent alternative solution to the classification problem and relies on the existence of a linear classifier in an appropriate space by using a set of training data to train the parameters of the model. It is based on the use of functions called kernel, which allows an optimal separation of data in different categories. When apply to anti-patterns detection, we believe that SVM can yield better precision and recall values when compared to that of previous approaches. SVM takes p features generated from each schema, maps it to p-dimensional vector space and build a hyper-plane as classifier using training database. We extract features from each schema of relational database related to each anti-pattern we are looking for and build a binary classifier for each anti-pattern. Binary classifier tells if the anti-pattern is present or not. We verify the accuracy of classifier with test database. In this way we can detect the schema design anti-patterns in our relational database design. We calculate precision and recall of both methods for comparative study and we find SVM based approach provides more precision and recall. This SVM based approach is more robust, accurate And can be applied incrementally.

II. RELATED WORK

Researchers have performed empirical studies to show that anti-patterns in software development like Spaghetti Code and Blob create hurdles during program comprehension, software evolution and maintenance activities. The Spaghetti code anti-pattern is characteristic of procedural thinking in object-oriented programming. But these approaches do not focus on database domain to detect anti-patterns. Ruben Wieman developed an Anti-Pattern Scanner to detect anti-pattern. He developed a famix parser to detect anti-pattern in java source code and added this parser on eclipse IDE .John Long from IBM worked on ”software reuse anti-pattern”. Vittorio Cortellessa? Anne Martensy, Ralf Reuss-nery, CatiaTrubiani? Worked on Identification of ”Guilty” Performance Antipatterns.ManjulaPeirisJames H. Hill devel-oped classifier to detect software performance antipattern.They developed a Non-intrusive Performance Anti-pattern Detector (NiPAD)which looks into cpu utilization to detect software performance. NiPAD defines the software performance anti-pattern detection problem as a binary classification problem of system performance metrics. Maria Teresa Llano and RobPoo-ley used UML specification to identify and predict object oriented anti-pattern. Ivan Polek, Samuel Snopko and Ivan Kapustk developed a rule based approach to automatically detect anti-patterns in software. Abdou Maiga developed a support vector machine to detect software anti-pattern. He used the training dataset to train the support machine vector machine to build the classifier and the used it to detect anti-patterns. Abdou Maiga extended his previous work to add user feedback in his design and plotted recall and precision graph to verify the accuracy of the classifier. But up to our knowledge no mechanism is developed to detect anti-pattern in schema design of database. We develop support vector machine to detect anti-pattern in schema design of database

III. ARCHITECTURE AND ALGORITHM

Now we discuss our detection algorithms .we use two approaches for anti-pattern detection one is based on pattern in schema for given anti pattern and the other is building classifier based on support vector mechanism

A. Support Vector Machines Based Approach

1) Work Flow: Our approach consists of building a classi-fier based on SVM as discussed above. We take metrics base on each of those anti-patterns .The flow chart as shown below elaborates actual procedure.

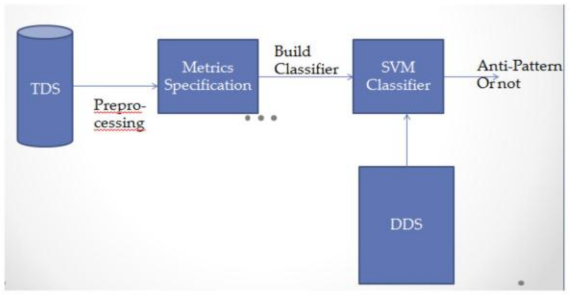


Fig. 1: *SVM Based Work Flow*

TDS is training data set. The stepwise procedure is as below

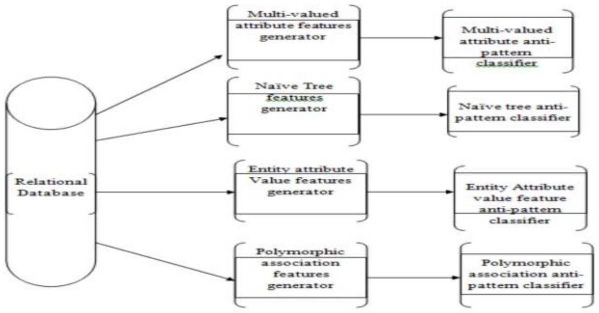
a) Step: It takes as input the training data set TDS . For each class from TDS, We calculate Object Oriented metrics that will be used as the attribute xi for each class in TDS to compute metrics for all the studied system.

b) Step: We train the SVM Classifier using the data set TDS and the set of the metrics computed in step 1. We define the training dataset as:

T DS = (xi; yi)jxi R^p; y 1; 1; 8i (1; 2; ::::::::; n) where yi is either 1 or -1, indicating respectively if a class xi is a multi-value attribute anti-pattern or not . Each xi is a p-dimensional real vector with p the number of metrics. The objective of the training step is to find the maximum margin hyper plane that divides the classes into the two different groups, multi-value attribute anti-pattern or NOT.

c) Step: Construction of the dataset DDS and detection of the occurrence of an anti-pattern: We build the dataset of the system on which we want to detect an anti-pattern as follows: For each class of the system, we compute the same set of metrics as in Step 1, We use the SVM classifier trained in Step 2 to detect the new occurrence of the anti-pattern in the dataset DDS.

2) Architecture of Support Vector Machine Based Ap-proach:


Relational Database
Multi-valued attribute features generator
Naïve Tree features generator
Entity attribute Value features generator
Polymorphic association features generator
Multi-valued attribute anti-pattern classifier
Naïve tree anti-pattern classifier
Entity Attribute value feature anti-pattern classifier
Polymorphic association anti-pattern classifier

3) Metrics Specification for Various Anti-Patterns: For various attributes we define metrices based on which classifier is trained from training dataset.
   a) Multi-values attributes:
      1) No of attribute with varchar type
      2) Delimiter separated list
      3) Length of delimiter separated list
      4) Varchar types - text,varchar
      5) Varchar(max), nvarchar

   b) Nave tree:
      1) Attribute refers to same table
      2) Depth of tree
      3) Total no of foreign key
      4) Types of hierarchical storage
   c) Entity attribute value features:
      1) Total no of attribute
      2) Total no of foreign key
      3) Total attribute with varchar type
      4) Attribute with name as substring
      5) Integrity constraint
   d) Polymorphic association:
      1) Total attribute with varchar type
      2) Attribute with value as table name
      3) Total no of foreign key

B. Algorithm to detect multi-valued attribute anti-patterns

a) Feature generation: Input: database Algorithm: gen-erate features multi valued attribute
   1) Calculate total no of attribute with varchar type
   2) Check for delimiter list
   3) Calculate size of delimiter list
   4) Store calculated features in a file

Output: mva feartures train.txt file containing features gener-ated by the database

b) Classifier training and testing: Input: mva features train.txt, mva features test.txt Algorithm:svm detect
   1) Train the classifier with mva features train.txt
   2) Test the classifier with mva features test.txt

Output: Precision, recall and accuracy of the classifier.

1) Algorithm to detect naive tree anti-patterns:
   a) Feature generation: Input: database Algorithm: gen-erate features naive tree
      1) Check for self-recurance

2) Calculate depth of tree
3) Calculate total no of foreign key
4) Store calculated features in a file

Output: naive_features train.txt file containing features generated by the database

b) Classifier training and testing: Input: naive features train.txt, naive features test.txt Algorithm: svm detect
1) Train the classifier with naive features train.txt
2) Test the classifier with naive features test.txt

Output: Precision, recall and accuracy of the classifier.

2) Algorithm to detect entity attribute value anti-patterns:

a) Feature generation: Input: relational database Algorithm: generate features entity attribute value
1) Calculate total no of attribute
2) Calculate total no of attribute with varchar type
3) Calculate total no of attribute with foreign key
4) Calculate total attribute with name as substring
5) Calculate total attribute with val as substring
6) Store calculated features in a file

Output: eav feartures train.txt file containing features generated by the database

b) Classifier training and testing: Input:eav features train.txt, eav features test.txt Algorithm: svm detect
1) Train the classifier with eav features train.txt
2) Test the classifier with eav features test.txt

Output: Precision, recall and accuracy of the classifier.

3) Algorithm to detect polymorphic association anti-patterns:

a) Feature generation: Input: relational database Algorithm: generate features poly asso
1) Calculate total no of attribute with varchar type
2) Total no of foreign key
3) Total attribute with value as table name
4) Store calculated features in a file

Output: pa feartures train.txt file containing features generated by the database

b) Classifier training and testing: Input: pa features train.txt, pa features test.txt Algorithm: svm detect
1) Train the classifier with pa features train.txt
2) Test the classifier with pa features test.txt

Output: Precision, recall and accuracy of the classifier.

C. Tools and technologies used

We worked out on phpMyAdmin database for storing train-ing database and php scripts to carry out various computations .For classifier we used built in open source library SVM-light

IV. RESULTS AND OBSERVATIONS

Result for multi-valued attributes

Training data size	Precision(%)	Recall(%)
18	20	25
36	33.3	30.56
54	38.46	41.67
72	41.18	43.75
90	43.75	91.18

TABLE I: *Multi valued attributes*

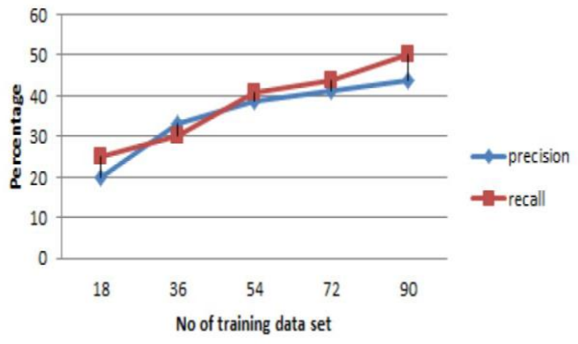


Fig. 2: *Multi valued attributes*

Result for entity attribute value

Training data size	Precision(%)	Recall(%)
20	0.0	0.0
168	25.58	26.19
240	46.67	45.16
408	65.38	66.67
624	75.8	70.28
888	82.36	83.3

TABLE II: *Entity Attribute Value*

Result for polymorphic association Result data obtained for

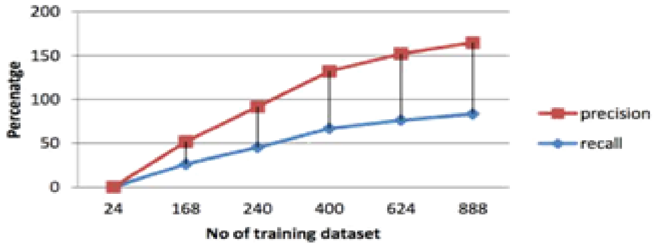


Fig. 3: *Entity Attribute value*

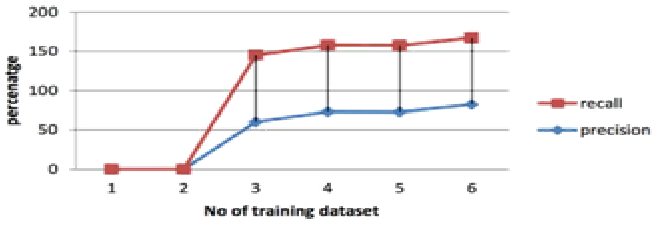


Fig. 4: *Polymorhic association*

Training data size	Precision(%)	Recall(%)
29	0	0
116	0	0
348	60.0	62.50
507	72	75.00
623	77.7	79.55

TABLE III: *Polymorhic association*

Training data size	Precision(%)	Recall(%)
4	0	0
20	50	50
100	77.23	80.95
240	91.2	91.2
368	96.87	96.8

TABLE IV: *Naive Tree*

naive tree anti pattern is as shown below From graph we can clearly see precision and recall value increases with increase in training dataset(TDS) i.e. accuracy of classifier increases.

V. CONCLUSION

Anti-patterns are poor solution to design and implemen-tation problems. Developers may introduce anti-patterns in their software systems because of time pressure, lack of un-derstanding, communication and or-skills. Anti-patterns create problems in software maintenance and development. Database anti-patterns lead to complex and time consuming query processing and loss of integrity constraints. Detecting anti-patterns could reduce costs, efforts and resources. Researchers have proposed approaches to detect anti-patterns in software development. But not much research has been done about database anti-patterns. This report presents two approaches to detect schema design anti-patterns in relational database. Our first approach is based on pattern matchingwe look into po-tential candidates based on schema patterns. Second approach is a machine learning based approach we generate features of possible anti-patterns and build SVM-based classifier to detect them. Here we look into these four anti-patterns a) Multi-valued attribute b) Nave tree based c) Entity Attribute Value and d)Polymorphic Association . We measure precision and recall of each approach and compare the results. SVM-based approach provides more precision and recall with more training dataset.

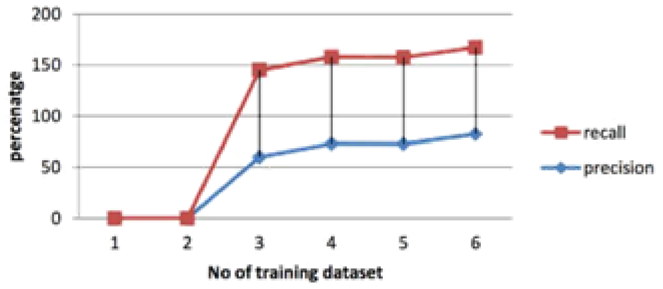


Fig. 5: *Naive Tree*

REFERENCES

[1] AbdouMaiga, Nasir Ali, Neelesh Bhattacharya, Aminata Sabane1,Yann-Gael Gueheneuc, and EsmaAimeur, " SMURF: A SVM-based IncrementalAnti-pattern Detection Approach", 19th Working Conference on Reverse Engineering, 2012.

[2] AbdouMaiga, Nasir Ali,Neelesh Bhattacharya, Aminata Sabane1,Yann "SVM for Antipattern Detection",.

[3] ManjulaPeiris, James H. Hill "Towards Detecting Software Performance Anti- patterns using Classification Techniques" .

[4] Robert Burbidge, Bernard Buxton "An Introduction to Support Vector Machines for Data Mining".

[5] Bo-yun Zhang, Jian-ping Yin, Jin-boHao, Ding-xing Zhang and Shu-lin Wang "Using Support Vector Machine to Detect Unknown Computer Viruses" .

[6] Ivan Polek, Samuel Snopko and Ivan Kapustk "Automatic identification of the antipatternsusing the rule-based approach" SISY 2012, 2012 IEEE 10th Jubilee International Symposium on Intelligent Systems and Informatics,September 20-22, 2012, Subotica, Serbia

[7] Maria Teresa Llano and Rob Pooley "UML Specification and Correction of Object-Oriented Anti-patterns" 2009 Fourth International Conference of Software Engineering Advances .

[8] Vittorio Cortellessa?, Anne Martensy, Ralf Reussnery, CatiaTrubiani - Towards Identification of "Guilty" Performance Antipatterns .

[9] John Long IBM "Software Reuse Anti-patterns "ACM SIGSOFT

[10] Ruben Wieman "Anti-Pattern Scanner: An Approach to Detect Anti-Patterns andDesign"S