A Survey of Bayesian Network Models for Decision Making System in Software Engineering

Nageswarao M.
Research Scholar,
CSE Department,
Sri Krishnadevaraya University,
Ananthapuram, India.

N. Geethanjali, PhD
Associate Professor ,
Dept. of Computer Science & Technology ,
Sri Krishnadevaraya University,
Ananthapuram, India.

ABSTRACT

Defect prediction and assessment are the essential steps in large organizations and industries where the software complexity is growing exponentially. A large number of software metrics are discovered and used for metric prediction in the literature. Bayesian networks are applied to find the probabilistic relationships among the software metrics in different phases of software life cycle. Defects in a software project lead to minimize the quality which might be the impact on the overall defect correction. Traditional Bayesian networks are system dependable and their models are invariant towards the accurate computation. Bayesian network model is used to predict the defect correction at various levels of the software development. This model reveals the high potential software efforts and metrics required to minimize the overall cost of the organization for decision support.

Keywords

Project metrics,probability estimation,Bayesian network.

1. INTRODUCTION

Over the last few years, Bayesian networks have become widely used in probabilistic models to predict the defect or quality analysis of software development to reduce human resources as well as software metrics. Bayesian networks served as a unique solution to the future generation of the artificial intelligence to predict the software development metrics. Early detection of defects in each phase can reduce the efforts of human and maintain the software resources efficiently. Software reuse Since software validity is judged by the software experts,which is measured by the defects in the software development phases.Prediction of defects in the projects may result decreasing at test time and also reduces the delivery time of the product.

Bayesian networks are directed acyclic visual graphical model in which nodes represent random

variables and edges represents conditional probabilistic dependencies of the random variables. Each node in the Bayesian network is represented as ovals or circles. The relationship of the nodes in the random variables indicates the joint probability distribution among them. Each node computes conditional probability distribution on its predecessors. So, each node maintain's the probability state table which specifies the variable dependency on its parent node.

For example, if there is an edge form A to B it means the node at the tail (C or A) of the edge depends the value of the node at the head B as shown in Fig 1. Software uncertainties should be modeled using predicted probability values computed using Bayesian belief netwoks.

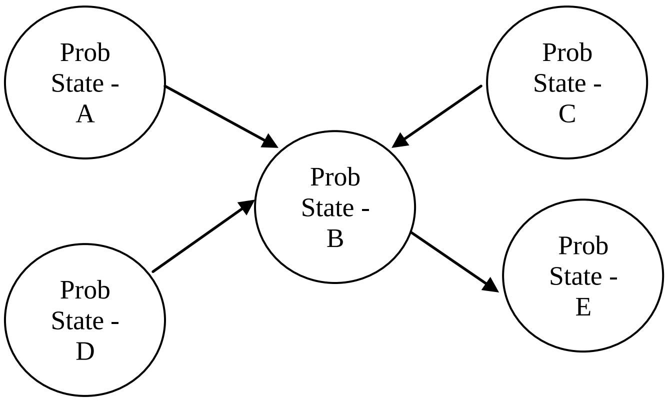


Fig 1: State Relationship in BNs

Underlying, Bayesian network is the Bayes theorem which formulates the posterior probability to predict the defects or errors in terms of prior probabilities as.

$$p(y_i = s / x_i) = p(\mathrm{x}_i / y_i = s).p(y_i = s) / p(x_i)$$

Where $p(x_i)$ denotes the normalized prior probability of the state s .

Many data mining approaches and statistical techniques have been proposed to predict the conditiaonal relationships between the random software metrics of the software development.For a predictive analysis of SDLC modules , many research implementations have been proposed, including association rules,regression analysis,decision trees,clustering analysis ,fuzzy logic and neural networks.Regression analysis can be used to find the dependency between the variables and decision patterns.Pattern rule mining can find relevant association patterns that can satisfy user given confidence and support measures on software metrics.Basically, Bayesian networks can be summarized as follows:

- This framework will predict uncertainties and estimates probabilistic measures.
- Integrates software data with knowledge decision patterns.
- Generates knowledge based risk analysis and planning.

- Explores the change in node behavior towards the software metrics.

In software project management ,causality and correlation has never been given sufficient attention.

2. CAUSALITY IN BAYESIAN NETWORKS

Bayesian networks based on the graph model ,prior probability estimations and are widely used to visualize uncertain knowledge and effective decision making process.A Bayesian network includes 4 types of local structures as shown below

$$softmetr1 \rightarrow softmetr2 \rightarrow softmetr3$$
$$softmetr1 \leftarrow softmetr2 \rightarrow softmetr3$$
$$softmetr1 \leftarrow softmetr2 \leftarrow softmetr3$$
$$softmetr1 \rightarrow softmetr2 \leftarrow softmetr3$$

Any two Bayesian networks are conditionally independence equivalent ,if they have the identical conditional probability for all random variables. In the above structures, first three patterns are independence equivalent Bayesian networks.Last pattern is not independence equivalent to the other three structures. All these structures represent causality and independence equivalent Bayesian networks, express the same dependency information to the joint probability distribution.

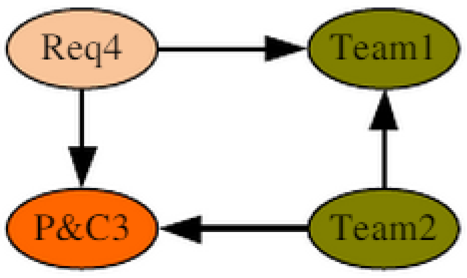


Fig 2: Simple Causality network without condition

Two structures such as Req→P&C←Team2 and Req4→ Team1←Team2 can be linked to the Bayesian network decision making as shown in Fig 2 and complex structures such as Team3← Product and User4← Req2← P&C1 are used for decision making process as shown in Fig 3.

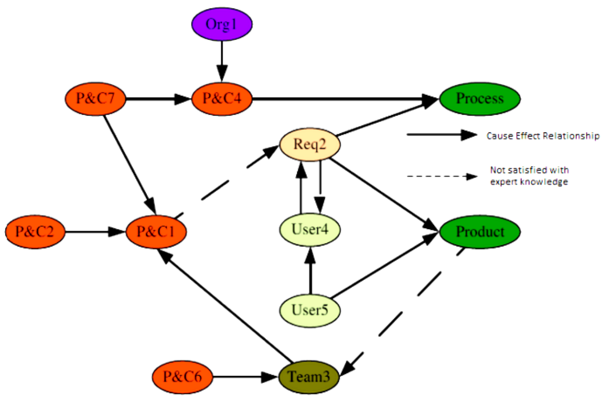

Org1
P&C7
P&C4
Process
Req2
Cause Effect Relationship
Not satisfied with expert knowledge
P&C2
P&C1
User4
Product
User5
P&C6
Team3

Fig 3: Complex Causality network without conditions

The basic idea of software defect analysis is based on Bayesian network with three basic steps: Firstly, each node expresses the software metrics which are more appropriate using causality dependency relationship.In the second step , each node determines the conditional probability distribution and find its parent node relationship.In the third step, the Bayesian reasoning model can be estimated to predict the decision on the software metrics.The basic criteria for choosing software reliability metrics are :

1. Correctness: This criterion includes justness,objectivity and recall.
2. Experience: This criterion provides the level in which this metric has been recognized.
3. Relevance: This factor reflects the correlation between the software reliability and metrics.
4. Practicality:This metric is used in the SDLC.
5. Feasibility: This metric supported by data collection ,tools and results are evaluated.

3. BAYESIAN NETWORK FRAMEWORK

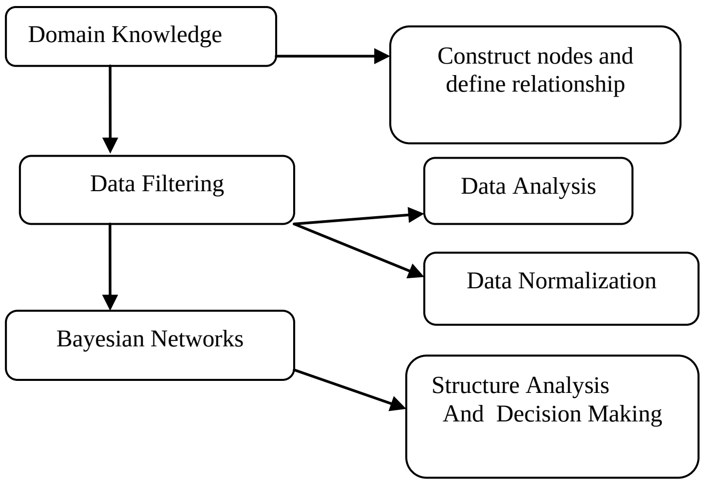


Fig 5: Bayesian Netowrks Frameworks

Domain Knowledge: The domain learning will support the training BN to a larger extent. The training of input data needs large amount of data.

Nodes Construction: The nodes in the Bayesian framework influence the SDLC schedule. In order to get more reliability,domain people , including software developer and quality engineer to survey the factors in each development phase. Relationships among the Bayesian nodes could be determined by the BNs algorithm. Some of the issues regarding the node relationship are:

- The programmer experience and domain knowledge will not affect the project modules.
- Staff income has no relevant dependency with the software coding .
- Coding procedure has an intense relationship to the developer's experience.

Based node designing types are: { Quality, coding metrics, process, coding complexity,requirements, team size, programmers experience,accuracy .etc}.

Data Preprocessing: Training data is the crucial part of the Bayesian networks. If the software, data is not reliable, the data collected from it will not get effective decision patterns in Bayesian networks. If all the metrics or factors are discrete, we need to transform it to continuous type.

Data Analysis: In order to optimize the accuracy of the Bayesian model, data consistency and noise filtering methods are used in the data analysis.

Data Normalization: In order to transform data distribution values to the unique format [0,1], data normalization method is used as shown in sample metric data. There are total 5 levels of severity of detects have been classified.

Severity #1: Very High

Severity #2: high.

Severity #3: Medium

Severity #4: Low

Severity #5: Very Low

Sno	Metric	Classes	Avg_Normalize
1	Coding process Quality	High,Medium,Low	[0.5,1]
2	Project size	High,Medium,Low	[0.4,0.8]
3	Requirements	High,Medium,Low	[0.2,0.9]
4	Developers experience	High,Medium,Low	[0,1]
5	Design complexity	High,Medium,Low	[0.6,1)

Prior Probability: Prior probability of the types of metrics is computed using the distribution of the each metric type of the Bayesian model .

$$P(T=m) = N_i / \mathrm{N}_T (1 <= i <= m.size)$$

$P(T=m)$ indicates the prior probability of the metric type m and it is computed by dividing the number of metrics in type I by the total number of metrics of same type regarding the all metrics.

4. SUMMARY OF BAYESIAN NETWORKS IN SOFTWARE ENGINEERING

Source	Problems Investigated
[2]	Coupling
[4]	Defects
[5]	Defects
[6]	Effort
[7]	Dynamic testing
[8]	Requirement review
[9]	Size,quality and effort

5. EXPERIMENTAL RESULTS

In the section, we have analyzed the different software metrics corresponding to the software development framework.`

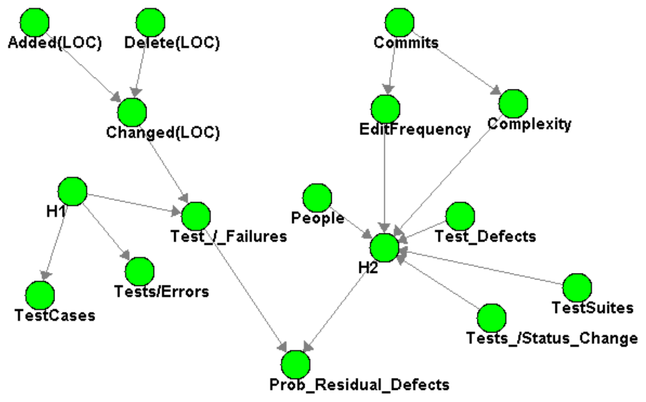


Fig 6: Model 2 Simulated Results

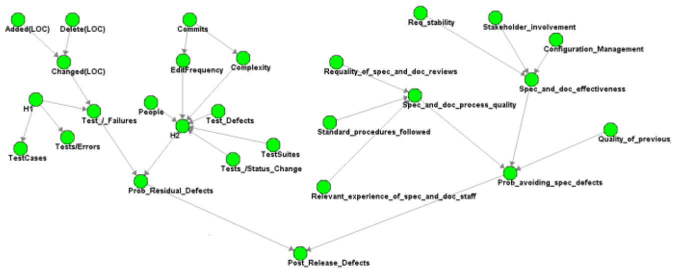


Fig 7: Model 3 Bayesian Network Simulation

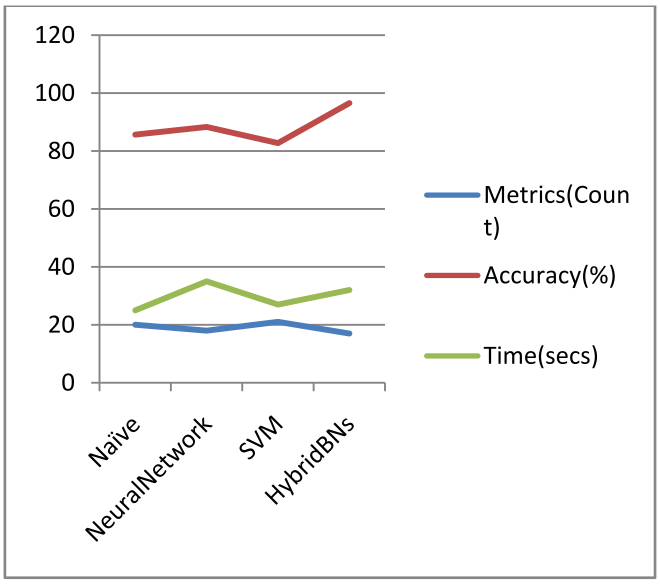
Bayesian Decision Patterns

```
Commits < 23.5
| TestSuits < 24.5
| | StandardProcedureFollowed < 25.5
| | | Spec_and_Doc_Effectiveness < 50.5
| | | | Added(LOC) < 30.5
| | | | | Delete(LOC) < 89 ==> 1
|  |  |  |  |  Delete(LOC) >= 89 ==> 6
|  |  |  |  Added(LOC) >= 30.5 ==> 6
|  |  |  Spec_and_Doc_Effectiveness >= 50.5
| | | | Prob_avoiding_spec_defects < 78.5 ==> 3
|  |  |  |  Prob_avoiding_spec_defects >= 78.5
| | | | | Delete(LOC) < 87 ==> 0
|  |  |  |  |  Delete(LOC) >= 87
| | | | | | Added(LOC) < 42.5 ==> 0
|  |  |  |  |  |  Added(LOC) >= 42.5 ==> 3
|  |  StandardProcedureFollowed >= 25.5
| | | StandardProcedureFollowed < 54.5
| | | | Added(LOC) < 69.5
| | | | | TestCases < 56.5
| | | | | | People < 51.5
| | | | | | | Spec_Doc_Proces_Quality < 7
| | | | | | | | Added(LOC) < 17 ==> 3
|  |  |  |  |  |  |  |  Added(LOC) >= 17 ==> 0
|  |  |  |  |  |  |  Spec_Doc_Proces_Quality >= 7
| | | | | | | | TestFailures < 28 ==> 0
|  |  |  |  |  |  |  |  TestFailures >= 28
| | | | | | | | | Added(LOC) < 17.5 ==> 3
|  |  |  |  |  |  |  |  |  Added(LOC) >= 17.5 ==> 4
|  |  |  |  |  |  People >= 51.5
| | | | | | | Delete(LOC) < 90.5
| | | | | | | | TestCases < 47.5 ==> 0
| | | | | | | | TestCases >= 47.5
| | | | | | | | | Added(LOC) < 25 ==> 0
| | | | | | | | | Added(LOC) >= 25 ==> 3
| | | | | | | Delete(LOC) >= 90.5 ==> 5
| | | | | TestCases >= 56.5
| | | | | | People < 90.5
| | | | | | | H1 < 79.5
| | | | | | | | Changed(LOC) < 97.5
| | | | | | | | | People < 55.5 ==> 3
| | | | | | | | | People >= 55.5
| | | | | | | | | | TestCases < 81 ==> 0
| | | | | | | | | | TestCases >= 81
| | | | | | | | | | | Spec_and_Doc_Effectiveness < 65
==> 3
| | | | | | | | | | | Spec_and_Doc_Effectiveness
>= 65 ==> 1
| | | | | | | | Changed(LOC) >= 97.5
| | | | | | | | | Req_Stability < 9 ==> 0
| | | | | | | | | Req_Stability >= 9
| | | | | | | | | | People < 78.5 ==> 3
| | | | | | | | | | People >= 78.5
| | | | | | | | | | | TestFailures < 88 ==> 3
| | | | | | | | | | | TestFailures >= 88 ==> 0
| | | | | | | H1 >= 79.5
| | | | | | | | Changed(LOC) < 83
| | | | | | | | | H2 < 40.5 ==> 9
| | | | | | | | | H2 >= 40.5 ==> 8
| | | | | | | | Changed(LOC) >= 83
| | | | | | | | | People < 37.5 ==> 9
| | | | | | | | | People >= 37.5 ==> 5
| | | | | | People >= 90.5
| | | | | | | TestFailures < 85.5
| | | | | | | | H2 < 57.5 ==> 0
| | | | | | | | H2 >= 57.5
| | | | | | | | | TestCases < 77.5 ==> 0
| | | | | | | | | TestCases >= 77.5 ==> 3
| | | | | | | TestFailures >= 85.5
| | | | | | | | Delete(LOC) < 76 ==> 0
| | | | | | | | Delete(LOC) >= 76 ==> 5
| | | | Added(LOC) >= 69.5
| | | | | Spec_Doc_Proces_Quality < 10.5
```

Performance Results

Approach	Metrics (Count)	Accuracy(%)	Time(secs)
Naïve	20	85.67	25
NeuralNetwork	20	88.32	35
SVM	20	82.76	27
HybridBNs	20	96.56	32

Traditional Algorithms Performance



6. CONCLUSION

In this paper, we studied Bayesian network framework on different types of software development features to explore the metric relationship and their decision patterns. Bayesian network model is used to predict the defect correction at various levels of the software development. This model reveals the high potential software efforts and metrics required to minimize the overall cost of the organization for decision support.The basic limitations of these traditional models are :1) unable to find the new patterns to the dynamic features.2) fail to load the data with a large number of instances. 3) Need to propose a new preprocessing technique to filter missing metrics.

7. REFERENCES

[1] Hearty, Peter; Fenton, Norman; Neil, Martin; Cates, Patrick,"Automated population of causal models for improved software risk assessment", Association for Computing Machinery — Nov 7, 2005.

[2] Xiaoxu Wang, Xiaoxu Wang; Chaoying Wu, Chaoying Wu; Lin Ma, Lin Ma,"Software project schedule variance prediction using Bayesian Network",IEEE, 2010.

[3] Hu, Yong; Zhang, Xiangzhou; Ngai, E.W.T.; Cai, Ruichu; Liu, Mei,"Software project risk analysis using Bayesian networks with causality constraints",Decision Support Systems , Volume 56 – Dec 1, 2013.

[4] Tim, Menzies; Gunes, Koru,"Predictive models in software engineering",Empirical Software Engineering , Volume 18 (3) – Jun 1, 2013.

[5] Jeet, Kawal; Rana, Yadvirender; Xin, Ruichi,"A Bayesian network based approach for software reusability prediction",ACM SIGSOFT Software Engineering Notes , Volume 37 (4) – Jul 16, 2012.

[6] Bencomo, Nelly; Belaggoun, Amel; Issarny, Valerie,"Dynamic decision networks for decision-making in self-adaptive systems: A case study",IEEE,2013.

[7] Alsri, Abdulrhman; Almuhammadi, Sultan; Mahmood, Sajjad,"A model for work distribution in global software

development based on machine learning techniques",IEEE,2014.

[8] Layman, L.; Shull, F.; Componation, P.; O'Brien, S.; Sabados,"A methodology for mapping system engineering challenges to recommended approaches",IEEE,2010.

[9] Otero, C.E.; Otero, L.D.; Weissberger, I.; Qureshi,"A Multi-criteria Decision Making Approach for Resource Allocation in Software Engineering",IEEE,2010.

[10] Fitzgerald, Brian; Musiał, Mariusz; Stol, Klaas-Jan,"Evidence-based decision making in lean software project management",Association for Computing Machinery — May 31, 2014.