Journal of Theoretical and Applied Information Technology
10th February 2014. Vol. 60 No.1
© 2005 - 2014 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

KERNEL OPTIMIZATION FOR IMPROVED NON-FUNCTIONAL REQUIREMENTS CLASSIFICATION

[1]K. MAHALAKSHMI, [2] Dr. R.PRABHAKAR [3] Dr. V. BALAKRISHNAN
[1]School of Engineering & Technology, Surya Group of Institutions, Department of Computer Science, Villupuram, Tamilnadu, INDIA
[2] Coimbatore Institute of Technology, Department of Computer Science, Coimbatore, INDIA
[3] Annamalai University, DDE, Department of Management Studies, Chidambaram, INDIA
E-mail: [1]mahalakshmik_2008@yahoo.com

ABSTRACT

Requirements Engineering (RE) refers to development of software-intensive systems. There are various kinds of software-intensive systems, and RE practice varies across this range. System requirements are descriptions of services provided by a system and its constraints. Functional requirements capture the system's intended behavior which may be expressed as services, tasks or functions, a system needs to perform. Unlike Functional Requirements, Non-Functional Requirements (NFR) state constraints to the system as well as particular notions of qualities a system might have.NFR analysis is an important RE activity. This paper uses Support Vector Machine (SVM) for classification of NFR; Radial Basis Function kernel is optimized using Hybrid Artificial Bee Colony (ABC).The hybrid ABC incorporates differential evolution.

Keywords: *Requirement Engineering (RE), Non-Functional Requirements (NFR), Support Vector Machine (SVM), Artificial Bee Colony (ABC), Differential Evolution*

1. INTRODUCTION

A requirement is a condition/capability to be fulfilled by a system to satisfy a contract, standard, specification, or other formally imposed documents (IEEE Standard 610.12-1990). Requirements for a system should be: consistent, correct, traceable and verifiable. RE is a process which elicits, understands, specifies and validates customers' and users' requirements. It identifies technological restrictions under which an application will be constructed and run. It is an iterative and co-operative process aimed at analyzing the issue, to document results in various formats to evaluate the results' precision [1].

When a software application is constructed, the development team needs certain knowledge about problem domains and application's requirements. Elicitation and specification of such requirements is a complex process to identify system functionality to satisfy users' and customers' needs. Though a standardized process supporting requirements handling and guaranteeing results quality is lacking, the best practices in general are software applications development providing techniques which are recommended by some Web methods for Web applications requirements specification. It is important that appropriate techniques selection is the responsibility of the development team and results success depends on this team, and the customers and users group participating in the process. RE's iterative process has three activities [2]:

- requirements elicitation
- requirements specification
- requirements validation

Requirements Engineering Activities

The following seven activities occurring during RE process are referred to in this investigation. They were used in a questionnaire to permit separate tasks identification, so preventing the issue of merged activities was identified by Houdek and Pohl [3].

Project Creation: The activity of setting up a project to develop a new product or modifying an existing one

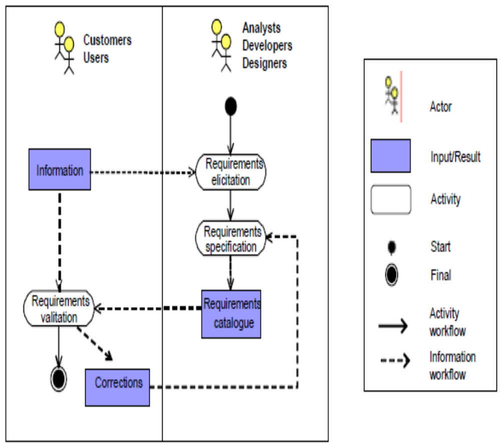

Figure 1: *The Requirements Engineering Process*

Requirements Engineering Activities

The following seven activities occurring during RE process are referred to in this investigation. They were used in a questionnaire to permit separate tasks identification, so preventing the issue of merged activities was identified by Houdek and Pohl [3].

Project Creation: The activity of setting up a project to develop a new product or modifying an existing one

Elicitation: refers to gathering system requirements from various stakeholders. Boundaries, stakeholder identification, goals and tasks performed are discovered now.

Interpretation and Structuring: Following requirements elicitation, they are interpreted, structured and analysed. Requirements are then documented. This phase is discussed separately; but it is usually, interwoven with requirements elicitation, as some analysis is invariable during elicitation.

Negotiation: This phase comprises of RE negotiating with stakeholders to agree on requirements definitions in requirements documentation

Verification and Validation: Verification and validation of requirements checks that requirements accurately represent system needs.

Change Management: this makes certain that same information is gathered for each change and that overall costs/benefits of proposed changes are reviewed

Requirements Tracing: This tracks each requirement's origins, so that when a change is necessary for a design component, original requirement is located [4].

Functional requirements capture the system's intended behavior which may be expressed as services, tasks or functions, a system needs to perform. In product development, it is good to differentiate between baseline functionality needed for a system to compete in that product domain, and features which differentiate a system from competitors' products, and variants in your company's own product line/family. A strategy to quickly penetrate a market, is to produce a core, or stripped down, basic product, and adds features to its variants to be released shortly. This release strategy also benefits information systems development, staging core functionality for early releases while adding features over many later releases.

Companies produce product lines with varied cost/feature variations in a line product in many industries. Product families include many product lines aimed at different markets or usage situations. What ensures that these product lines are part of a family are common functionality elements and identity. This commonality is leveraged by a platform based development approach that uses a set of reusable assets across the family.

Such strategies are important for software architecture. Specifically, it is not just functional requirements of first product or release that should be supported by architecture. Early (nearly concurrent) releases functional requirements need to be considered. Later releases are accommodated via architectural qualities like extensibility and flexibility. The latter are non-functional requirements [5].

Unlike Functional Requirements, NFRs also called Quality Requirements [6] state constraints to system and also specific quality notions a system might have. Usability, accuracy, performance, safety, reliability and security are examples, so it can be said, that while functional requirements state "what" a system should do, NFR's constrains "how" a system must accomplish "what". Consequently, NFRs are linked to a Functional Requirement.

Functional requirements address specific problems and so are implemented through specific localized modules/components. Although stated informally, they are formalized when necessary.

NFRs define a software system or subsystem's global constraints on a functional requirement, on development or deployment processes. They are global as they arise from all parts of a system and from interactions. There are notations to specify functional requirements and so they are usually stated informally in requirements documents thereby making it hard to enforce during development and difficult for evaluation by customers before delivery. NFRs cannot be easily evaluated by stakeholders as they can be interpreted differently under differing contexts.

Support Vector Machines (SVM's) are a new binary classification learning method, the idea being to locate a hyper plane separating the d-dimensional data perfectly into two classes. But, as sample data is usually not separable linearly, SVM's introduce the idea of a “kernel induced feature space" which casts data into higher dimensional space where it is separable. Casting into such space can cause problems computationally and also due to over fitting. SVM’s key insight is that higher-dimensional space doesn't need to be handled directly (as only formula for dot-product in that space is needed), thereby erasing the above concerns. Further, SVM's VC-dimension (a measure of a system's chances of performing well on unseen data) can be calculated, unlike learning methods like neural networks which lack such a measure. SVM's overall are intuitive, well-founded theoretically, and are successful practically. They have also been extended to solve regression tasks (where a system is trained to output a numerical value and not a “yes/no" classification) [7].

This study uses SVM for classification. The Radial Basis Function (RBF) kernel is optimized using Hybrid Artificial Bee Colony (ABC). The remainder of the study is organized as follows: section 2 literature survey, section 3: Methodology, Section 4: results and discussion and section 5: conclusion.

2. RELATED WORKS

A Survey of RE Interactions in Telecommunication Systems, challenged from a RE perspective was proposed by Alnafoosi [8]. Telecommunication systems can be abstracted to have an infrastructure layer which delivers services and a consumer layer where end user devices use such services. This investigation surveyed current telecommunication industry interaction landscape with RE. The aspects of change, complexity, internal and external components challenged RE to capture and present stable requirements with known and unknown limitations and accommodation for future use. Multiple RE approaches, on frameworks, tools and solutions in telecommunication industry are explored in this survey.

Ang, et al., [9] details RE techniques in developing expert systems. This study analyzed expert system attributes, RE processes in expert system developments and techniques applicable to expert system developments. Next, it suggested analysis based appropriate techniques for expert system developments.

A requirement Tool to Support Model Based RE was proposed by Lu, et al., [10]that presents a Model-driven Object-oriented Requirement editor (MOR Editor) and also supported requirement document modeling and model-driven document editing. MOR Editor supports, RE objectized requirement artifacts, link related requirement artifacts, ensured reusable requirement template and created a requirement document model that integrated artifacts in analysis, design and implementation phases. With MOR Editor’s support, requirement documents consistency, traceability, completeness and maintainability could be enhanced. A case study using MOR Editor demonstrates merit of proposed requirement editor.

An Effective RE Process Model for Software Development and Requirements Management was proposed by Pandey, et al., [11] which proposed an effective RE process model to produce software development quality requirements. The requirement management and planning phases were executed independently for effective requirements management. It was iterative in for better RE and maintenance later. The successful implementation of proposed RE process impacted positively on the software product production quality.

A RE practice research framework for public service organizations was proposed by Zakaria, et al., [12], and implemented without knowing by IT Personnel. This framework was designed to identify RE practice that was implemented during software project requirement in Malaysian Public Sector. This investigation determined Software Requirement System (SRS) flow from initial interview with a list of IT Personnel RE practice experience for software project from a pilot test. It identified problems that arose and the RE practice implemented from the survey. The identified RE practice will be validated by IT experts in the Malaysian Public Sector. The idea to develop standards, tools, and methods could improve current practice. The expectation was that such practice could be a guideline to IT Personnel using RE practice in software project requirements.

A survey on issues in non-functional requirements elicitation was proposed by Ullah, et al., [13], which left a serious flaw in system design to user satisfaction and lead to projects failure like the classic example of the London Ambulance System. Chances of software success can be maximized when dealing with NFR at requirements level. This literature survey was the first effort of its kind to seek key NFR challenges and issues at the elicitation level of RE. It also discovered approaches and methods suggested in literature to offset such issues.

Conflicts Analysis among Non-Functional Requirements using Integrated Analysis of Functional and Non-Functional Requirements was proposed by Sadana and Liu [14]. A framework was provided for this using functional and non-functional requirements integrated analysis. The framework identified and analyzed conflicts based on quality attributes relationships, functionalities and constraints. As poorly structured requirement statements lead to confusing specifications, they also developed canonical forms to represent non-functional requirements. The framework's output was a conflict hierarchy that refined conflicts level by level among non-functional requirements. Finally, a case study was provided where the proposed framework analyzed and detected conflicts among a search engine's non-functional requirements.

Handling non-functional Requirements in Information System Architecture Design proposed by Tsadimas, et al., [15] was complex depending on functional and non-functional requirements. As system architecture definition was related to system performance, non-functional requirements had a major role in enterprise information system design. A model based approach emphasizing non-functional requirements was proposed to explore effect of non-functional requirements on system design process. Requirement derivation process was discussed and a case study where proposed concepts were applied while redesigning a large scale organization's legacy system was presented.

Domain Independent Ontology for Non-Functional Requirements was proposed by Dobson, et al., [16]. Despite research on ontologies to represent requirements models, little progress was made in ontologies representing nonfunctional requirements. The latter defined the resulting system's overall qualities. As they restricted system services, nonfunctional requirements were of critical importance and functional requirements were sacrificed to meet them. Due to their diverse nature, nonfunctional requirements were expressed in non-standard domain specific ways. This investigation describes a non-functional requirements ontology that could structure and express constraints as part of a quality of service specification. The approach is illustrated through a case study.

Affleck and Krishna [17] suggested supporting quantitative reasoning of non-functional requirements a process-oriented approach as inadequate requirements elicitation, analysis, specification, validation and management was a long standing problem in software engineering. Lack of defined requirements was a major cause for project failure. Many known techniques and frameworks were developed to deal with RE's functional aspects. Recent years saw the emergence of frameworks incorporating nonfunctional requirements. This research presents a process orientated, lightweight, quantitative extension to NFR Framework focusing on providing quantitative support to decision process and how decisions affect systems.

Galster and Bucherer [18] proposed Taxonomy to Identify and Specify Non-Functional Requirements in Service Oriented Development which presented taxonomy for non-functional requirements in service oriented context. Taxonomy implemented three non-functional requirements categories; process requirements, non-functional external requirements, and non-functional service requirements. Taxonomy was applicable with individual services and service based systems as a whole. Taxonomy was considered a starting point and check list when handling non-functional issues in service oriented, specifically highly distributed environments.

3. METHODOLOGY

In Non-Functional requirements analysis is an important activity in RE. System requirements are descriptions of services provided by system and operational constraints. There is lop-sided emphasis on software functionality, though it is neither useful nor usable without necessary non-functional characteristics. This study uses SVM for classification, The RBF kernel is optimized using Hybrid ABC made up of ABC with differential evolution.

Inverse Document Frequency (IDF)

Assume there are N documents in a collection, and term ti occurs in ni of them. Hence, the measure proposed by Sparck Jones, as a weight applicable to term ti, is essentially

$$idf\left(ti\right)=\log\frac{N}{n_i} \qquad (1)$$

Actually this is not accurate – the original measure is an integer approximation to this formula, and logarithm is specifically to base 2. But, as seen below, the logarithm's base is not important. Equation 1 is a commonly cited form of Inverse Document Frequency (IDF) [19].

The framework laid out in the remainder of this section easily derives new IDF variants which are more flexible and robust than present variants and lead to improved retrieval effectiveness for various tasks.

$$RSV\left(q,d,j\right)=\frac{P\left(r,|d,q,J\right)}{P\left(\bar{r},|d,q,J\right)}$$

$$=\log\prod_{i=1}^{|v|}\frac{P\left(r,|d,q,J\right)p\left(q,r,J\right)}{P\left(\bar{r},|d,q,J\right)P(q,\bar{r},j)}$$

$$\overset{rank}{\Rightarrow}\log\prod_{i=1}^{|v|}\frac{p\left(q,r,J\right)}{P(q,\bar{r},j)}$$

$$\overset{rank}{\Rightarrow}\sum_{i:d_{i=1}}\log\frac{P\left(d_i|q,r,J\right)p\left(q,r,JP\left(\bar{d}_1|q,r,J\right)\right.}{P\left(\bar{d}_1|q,r,J\right)p\left(q,r,JP\left(d_i|q,r,J\right)\right.}$$

Q is a random variable representing query or information need and q a specific query in event space of Q, J is a set of relevant information. Here, J, which is novel to this derivation, stands for 'judgments', and is any relevance data source available, including (pseudo-) relevance feedback documents or click-through logs. The J set is partitioned into Jr and Jnr, relevant and non-relevant judgment data in J, respectively [20].

Support Vector Machine (SVM)

Support Vector Machine (SVM) is a computer algorithm which assigns labels to objects learning by example. For example, a SVM can recognize fraudulent credit card activity through examination of thousands of fraudulent and non-fraudulent credit card activity reports. Alternatively, it can recognize handwritten digits through examination of a large collection of scanned images of handwritten zeroes, ones and so on. SVMs are also successfully applied to an increasingly variety of biological applications. SVM is a mathematical entity, an algorithm (or recipe) to maximize a particular mathematical function regarding a given data collection [21].

This equation presents QP formulation for SVM classification. This is a simple representation.

$$\min_{\mathrm{f},\xi_i}\left\|\mathrm{f}\right\|_K^2+C\sum_{i=1}^{l}\xi_i$$

$$\mathrm{y}_\mathrm{i}\mathrm{f}\left(\mathrm{x}_\mathrm{i}\right)\geq 1-\xi_\mathrm{i},\ \text{for all i}\ \xi_\mathrm{i}\geq 0$$

SVM classification, Dual formulation:

$$\min_{\alpha_i}\sum_{\mathrm{i}=1}^{1}\alpha_\mathrm{i}-\frac{1}{2}\sum_{\mathrm{i}=1}^{1}\sum_{\mathrm{j}=1}^{1}\alpha_\mathrm{i}\alpha_\mathrm{j}\mathrm{y}_\mathrm{i}\mathrm{y}_\mathrm{j}\mathrm{K}\left(\mathrm{x}_\mathrm{i},\mathrm{x}_\mathrm{j}\right)$$

$$\sum_{i=1}^{l}\alpha_i y_i=0$$

$$0\leq\alpha_\mathrm{i}\leq\mathrm{C},\ \text{for all i;}$$

Variables ξ_i are called slack variables measuring the error at point (x_i,y_i). Training SVM is challenging when training points numbers are large.

A real world problem will not ensure a line dividing data within space. And there might be a curved decision boundary. There could also be a hyperplane which might separate data but this is not desirable if data has noise. It is better for a smooth boundary to ignore few data points than being curved or in loops, around outliers. This is handled differently; the term slack variables are introduced here. Havingy_i(w'x + b) ≥ 1 - S_k which allows a point to be a small distance S_k on the hyper plane's wrong without violating a constraint. It might end up having huge slack variables that permit a line to separate data, and so in such scenarios, introducing the Lagrangian variable that penalizes large slacks.

$$\min\ \mathrm{L}=\frac{1}{2}w'w-\sum\lambda k\left(yk\left(w'xk+b\right)+sk-1\right)+\alpha\sum sk$$

Where reducing α allows more data to lie on the hyper plane's wrong side and is treated as outliers giving a smoother decision boundary [22, 23].

Artificial Bee Colony (ABC)

Artificial Bee Colony (ABC) algorithm is a recently introduced swarm-based algorithm where the position of food source represents a possible solution to optimization problem and nectar amount in a food source corresponds to the associated solution's quality (fitness). The number of employed bees or onlooker bees is equal to number of solutions in a population [24].

Hybrid schemes are categorized into two types: staged pipelining type hybrid and additional-operator type hybrid. In the first type, optimization process is applied to all individuals in the population, followed by further improvement using DE search. In the second hybridization type, DE is applied as a standard genetic operator to a corresponding probability. In this hybridization scheme pipelining type hybrid method is used due to its advantages. In this method, a generation, after stochastic optimization process (Artificial Bee Colony) is applied to all population individuals. In the population, *n* best Differential vectors from current population are selected based on fitness values to generate initial population needed for local search via DE. The search continues till it reaches maximum generations or satisfies predefined criterion [25].

```
The steps of HDABCA are summarized as follows
(1) Initialization
(2) Move the employed bees onto their food sources
    and evaluate their nectar amounts.
(3) Place the onlookers depending upon the nectar
    amounts obtained by employed bees.
(4) Send the scouts for exploiting new food sources.
(5) Memorize the best food sources obtained so far.
(6) Select best (best fitness) n differential vectors
    from the population based on the fitness calculated
    to generate initial population for differential evolution
(7) Do
For $i = 1$ to number of particles
Do
{
Mutation
Crossover
Selection
}
END FOR
    (8) If a termination criterion is not satisfied, go
        to step 2; otherwise stop the procedure and
        display the best solution is obtained so far
```

The DE algorithm introduced by Storn and Price (1995) is a new parallel direct search method, using NP parameter vectors as population for every generation G. DE is categorized into a class of floating-point encoded, evolutionary optimization algorithms. Presently, there are many DE variants; the specific variant used in this investigation is DE/rand/1/bin scheme.

The ABC-DE algorithm has three types of bees: employed, onlooker and scout bees. One half of the colony includes employed bees and the other half onlooker bees. For each food source (solution), there is one employed bee i.e. the employed bee's number equals the number of food sources. The employed bees produce new solutions from current solutions by.

$$V_{j,1}^{(G+1)} = \left\{ \begin{cases} x_{j,r3}^{(g)} + FX\left(x_{j,r1}^{(G)} - x_{j,r2}^{(G)}\right), & if\ rand_j[0,1] < CR\ or\ j = K, \\ x_{j,1}^{(g)} & otherwise \end{cases} \right\} \quad (1)$$

Based on probability, onlooker bees choose solutions with better fitness producing new solutions by (1). Probability of choosing a solution is calculated by

$$p_i = \frac{fit_i}{\sum_{n=1}^{sn} fit_n}$$

The new solutions replace current solution, if new solution's fitness is equal or better than current one; this being done by

$$x_i^{(G+1)} = \begin{cases} V_i^{(G+1)} & {}^{if} V_i^{(G+1)} \leq \hat{u}\ \left(X_i^{(G)}\right) \\ X_i^{(G)} & otherwise \end{cases}$$

If a better solution cannot be produced by employed or onlooker bees, after some iterations, determined by control parameter limit, employed bee converts to scout and initializes solution randomly. As in DE, control parameters F and CR are ABC-DE algorithm's important elements. These control parameters play a big role to produce new solutions. As observed, the ABC-DE algorithm's structure is similar to ABC; whereas the way of producing a new solution, is similar to DE. In fact, ways to produce new solutions in DE are embedded in the ABC algorithm [26] itself.

4. EXPERIMENTS AND RESULTS

In this study SVM is used for the classification of NFR. The RBF kernel is optimized using Hybrid ABC which is made up of ABC followed by differential evolution. The results are shown from figure 2-7.

Figure 2 shows Best fitness showed by the proposed ABC-DE algorithm which is 0.200157578

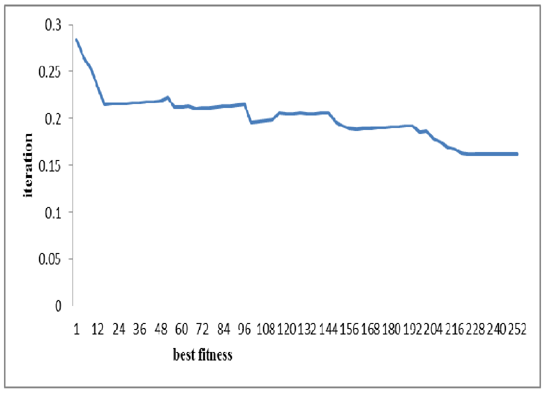

Figure 2: Best fitness

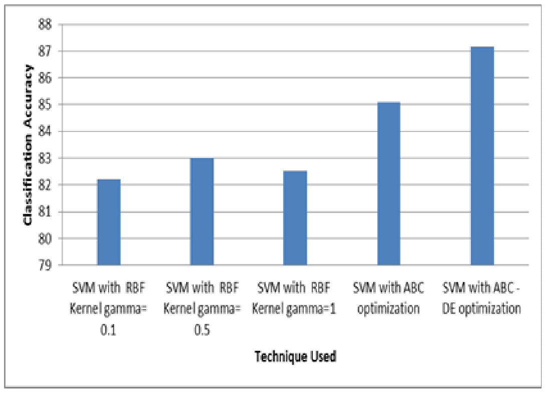

Figure 3: Classification Accuracy

Figure 3 shows that the proposed hybrid ABC-DE shows high classification accuracy of 87.17948718% compared to SVM with RBF Kernel gamma= 0.1,SVM with RBF Kernel gamma= 0.5,SVM with RBF Kernel gamma=1,SVM with ABC optimization methods.

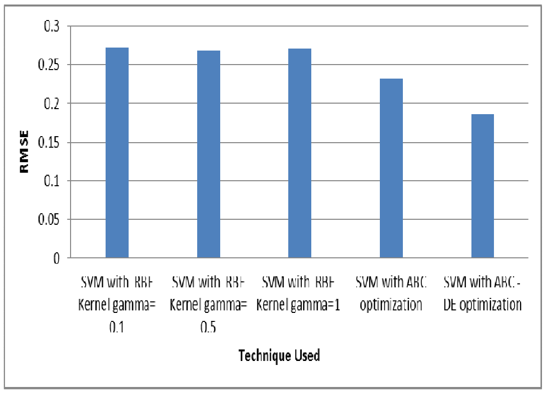

Figure 4: RMSE

Figure 4 shows that the proposed hybrid ABC-DE shows low RMSE value of 0.1862compared to SVM with RBF Kernel gamma= 0.1,SVM with RBF Kernel gamma= 0.5,SVM with RBF Kernel gamma=1,SVM with ABC optimization methods

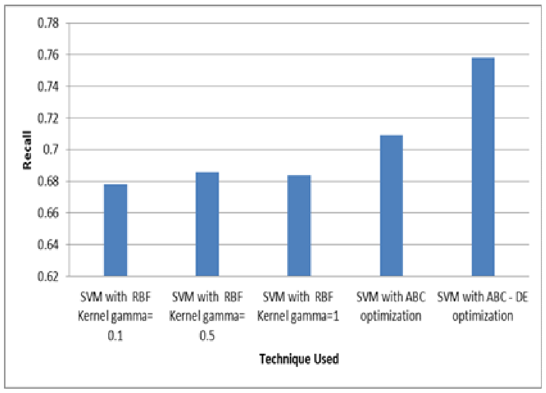

Figure 5: Average Recall

Figure 5 shows that the proposed hybrid ABC-DE shows high average recall of 75.7909% compared to SVM with RBF Kernel gamma= 0.1, SVM with RBF Kernel gamma= 0.5, SVM with RBF Kernel gamma=1, SVM with ABC optimization methods.

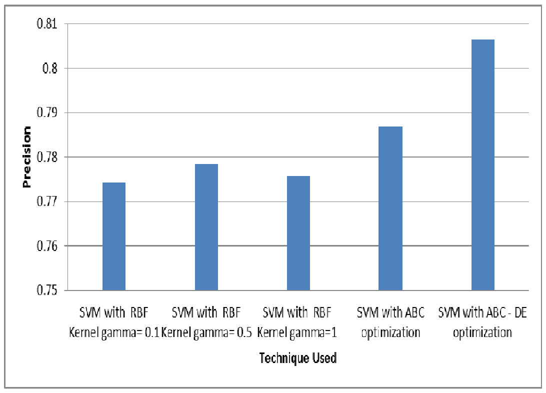

Figure 6: Average Precision

Figure 6 shows that the proposed hybrid ABC-DE shows high average Precision of 80.6363636% compared to SVM with RBF Kernel gamma= 0.1,SVM with RBF Kernel gamma= 0.5,SVM with RBF Kernel gamma=1,SVM with ABC optimization methods.

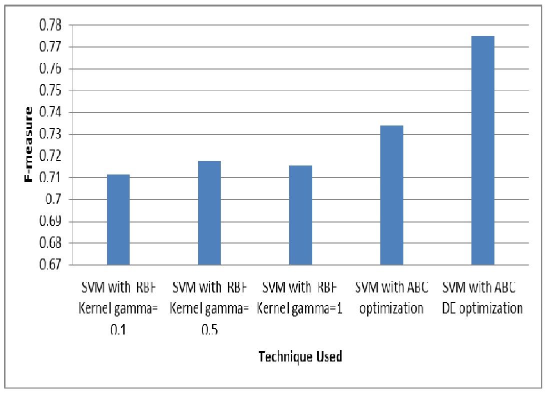

Figure 7: F-measure

Figure 7 shows that the proposed hybrid ABC-DE shows high F-measure of 77.5% compared to SVM with RBF Kernel gamma= 0.1, SVM with RBF Kernel gamma= 0.5, SVM with RBF Kernel gamma=1, SVM with ABC optimization methods.

5. CONCLUSION

RE process model is iterative and is a continual task through the project. RE activities occur across many phases, making process models appear iterative. The main step in determining non-functional requirements is by identifying NFRs type and categories. This study uses SVM for classification, the RBF kernel is optimized using Hybrid ABC made up of ABC followed by differential evolution. The hybrid ABC-DE reveals 75.7909% high average recall, 80.6363636% average Precision,87.17948718% classification accuracy, 77.5 % F-measure and 0.1862% low RMSE value compared to SVM with RBF Kernel gamma= 0.1, SVM with RBF Kernel gamma= 0.5, SVM with RBF Kernel gamma=1, SVM with ABC optimization methods.

REFRENCES:

[1] Ferreira, M.J., Loucopoulos, P. (2001). Organisation of Analysis Patterns for effective Re-use. Proceedings of the International Conference on Enterprise Information Systems. ICEIS 2001. Setubal,

[2] Escalona, M. J., & Koch, N. (2004). Requirements engineering for web applications-a comparative study. *J. Web Eng.*, *2*(3), 193-212.

[3] Portugal Houdek, F. and Pohl, K. (2000): Analyzing requirements engineering processes: a case study Proceedings of the 11th International Workshop on Database and Expert Systems Applications, Greenwich, UK, 6-8 September, pp.983-987

[4] Martin, S., Aurum, A., Jeffery, R., &Paech, B. (2002, December). Requirements engineering process models in practice. In *7th Australian workshop on requirements engineering. Deakin University, Melbourne, Australia*(pp. 41-47).

[5] Malan, R., Bredemeyer, D., & Consulting, B. (1999). Functional requirements and use cases. *functreq. pdf, 39k) June.*

[6] Cysneiros, L. M., & Yu, E. (2004). Non-functional requirements elicitation. In*Perspectives on software requirements* (pp. 115-138). Springer US.

[7] Boswell, D. (2002). Introduction to support vector machines.

[8] Alnafoosi, A. B. (2012, April). A Survey of Requirement Engineering Interactions in Telecommunication Systems. In *Information Technology: New Generations (ITNG), 2012 Ninth International Conference on* (pp. 885-885). IEEE.

[9] Ang, J. K., Leong, S. B., Lee, C. F., &Yusof, U. K. (2011, March). Requirement engineering techniques in developing expert systems. In *Computers& Informatics (ISCI), 2011 IEEE Symposium on* (pp. 640-645). IEEE.

[10] Lu, C. W., Chang, C. H., Chu, W. C., Cheng, Y. W., & Chang, H. C. (2008, July). A requirement tool to support model-based requirement engineering. In *Computer Software and Applications, 2008. COMPSAC'08. 32nd Annual IEEE International* (pp. 712-717). IEEE.

[11] Pandey, D., Suman, U., & Ramani, A. K. (2010, October). An effective requirement engineering process model for software development and requirements management. In *Advances in Recent Technologies in Communication and Computing (ARTCom), 2010 International Conference on* (pp. 287-291). IEEE.

[12] Zakaria, N. H., Haron, A., Sahibuddin, S., & Harun, M. Requirement Engineering Critical Issues in Public Sector Software Project Success Factor. *International Journal of Information and Electronics Engineering (fJ1EE), 201*(1).

[13] Ullah, S., Iqbal, M., & Khan, A. M. (2011, July). A survey on issues in non-functional requirements elicitation. In *Computer Networks and Information Technology (ICCNIT), 2011*

International Conference on (pp. 333-340). IEEE.

[14] Sadana, V., & Liu, X. F. (2007, July). Analysis of conflicts among non-functional requirements using integrated analysis of functional and non-functional requirements. In *Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International* (Vol. 1, pp. 215-218). IEEE.

[15] Tsadimas, A., Nikolaidou, M., &Anagnostopoulos, D. (2009, September). Handling non-functional requirements in information system architecture design. In *Software Engineering Advances, 2009. ICSEA'09. Fourth International Conference on* (pp. 59-64). IEEE.

[16] Dobson, G., Hall, S., &Kotonya, G. (2007, October). A domain-independent ontology for non-functional requirements. In *e-Business Engineering, 2007. ICEBE 2007. IEEE International Conference on* (pp. 563-566). IEEE.

[17] Affleck, A., & Krishna, A. (2012, June). Supporting quantitative reasoning of non-functional requirements: A process-oriented approach. In *Software and System Process (ICSSP), 2012 International Conference on* (pp. 88-92). IEEE.

[18] Galster, M., & Bucherer, E. (2008, July). A taxonomy for identifying and specifying non-functional requirements in service-oriented development. In *Services-Part I, 2008. IEEE Congress on* (pp. 345-352). IEEE.

[19] Robertson, S. (2004). Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of documentation*, *60*(5), 503-520.

[20] Metzler, D. (2008, October). Generalized inverse document frequency. In *Proceedings of the 17th ACM conference on Information and knowledge management* (pp. 399-408). ACM.

[21] Noble, W. S. (2006). What is a support vector machine? *Nature biotechnology*,*24*(12), 1565-1567.

[22] Support Vector Machines Lecturer: Shivani Agarwal

[23] Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and computing*, *14*(3), 199-222.

[24] Karaboga, D., &Akay, B. (2009). A comparative study of artificial bee colony algorithm. Applied Mathematics and Computation, 214(1), 108-132

[25] Abraham, A., Jatoth, R. K., & Rajasekhar, A. (2012). Hybrid differential artificial bee colony algorithm. *Journal of Computational and Theoretical Nanoscience*,*9*(2), 249-257.

[26] Alizadegan, A., Meybodi, M. R., & Asady, B. (2012). A NOVEL HYBRID ARTIFICIAL BEE COLONY ALGORITHM AND DIFFERENTIAL EVOLUTION FOR UNCONSTRAINED OPTIMIZATION PROBLEMS. *Advances in Computer Science and Engineering*, *8*(1).

.