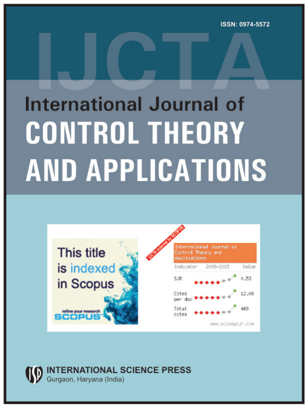

International Journal of Control Theory and Applications

ISSN : 0974-5572

Volume 9 • Number 43 • 2016

Framework for Determining Maturity Level of Test Organization Using RCNN

Kripa Joy[a], Ani Sunny[b] and Linda Sara Mathew[c]

[a]*Department of Computer Science and Engineering, M.A College of Engineering, Kothamangalam, Kerala, India. Email: kripa.joy89@gmail.com*
[b-c]*Asst. Prof. Department of Computer Science and Engineering, M.A College of Engineering, Kothamangalam, Kerala, India. Email:* [b]*lindasaramathew@gmail.com;* [c]*anisunny88@gmail.com*

Abstract: Testing is an important phase in software development. Many organizations have not fully realized and recognized their testing process that can effectively influence the improvement of their development process and doesn't have sufficient insight into the effectiveness of these testing process. If we don't consider these, it will lead to lack of maturity in a test organization and often leads to frequent conflicts between the business and IT. Improving the test process is essential for ensuring the quality of the system being tested. Many Maturity Models are available to find the maturity level of the test organization and suggest the improvements to their process areas, but they are focusing a limited set of testing process areas for determining that. The proposed system provides an effective approach to determine the maturity level of Test Organization by considering twenty four key process areas and seven vectors and provide test process improvement for achieving better maturity level. The framework has a set of questions, which will be answered by the testing staff working in the test organization. These answers will give the information about the current situations of the organization. Information collected from the testing staff will be evaluated by the NLP engine, which consists of Recurrent Convolutional Neural Network. RCNN is a deep neural network used to categorize the information.

Keywords: Software Testing; Maturity Model; Testing Maturity Model; Assessment Model; NLP; RCNN.

1. INTRODUCTION

Software testing is a process of executing an application in order to find out the software bugs. It validates and verifies the application to check whether the product meets the business and technical requirements that are explained in its design and development plan. To produce a good quality product required by the client, software testing is important. In order to deliver best quality products, the quality of the software testing process must be ensured. Test maturity models are the best way to ensure the quality of testing process in test organization.

Many maturity models are available to ensure and improve the software development process, but a little of them focus the improvement of testing process. Existing maturity models neither address the testing issues nor has the definition of the nature of a mature testing process. Most of them find the current situation of the organization by an assessment procedure and manually evaluate the situations to find out the maturity level of the organization. For finding the information from different testing staff, interviews are conducted

Because of the vital role of testing in software development, a new system, "Framework for determining the maturity level of a Test Organization using RCNN" is proposed. In this framework, mature testing is defined and natural language processing technology is added in assessment procedure. It will result in highly efficient Maturity Model. Artificial neural network is used to improve the efficiency of NLP engine for understanding the natural languages. Proposed Framework uses Recurrent Convolutional Neural Network (RCNN) in assessment procedure for classifying the text which is given by the testing staff. RCNN has a bi-directional recurrent structure for capturing the contextual information while learning process of word representations. A max-pooling layer automatically judges, what all features play key role in text classification, to capture the key component in the texts.

The remaining part of this paper is organized as follows: Section 2 presents related works; Section 3 describes our proposed framework, which includes the mature testing and components of this framework; Section 4 exhibits the output of the framework. Finally, the conclusions are shown in Section 5.

2. RELATED WORK

There are several Process Maturity Models available for finding out the maturity level of an organization and the road map for test process improvement. A few models are available for focusing the test activities. Mature testing is defined as the degree of optimization of process [1]. Capability Maturity Model Integration [2] is one of the main maturity models used for evaluating the maturity level of the software development organization. The CMMI provides guidance to software organizations on how to gain control of their processes for developing and maintaining software and how to rise towards a culture of software engineering and management excellence.

The CMMI is designed to guide software organization by determining the current process maturity and identifying the few issues most critical to software quality and process improvement and select process improvement strategies. This provides a framework for organizing the evolutionary steps into five maturity levels [3]. The maturity of an organization's software process helps to predict a project's ability to meet its goals.

CMMI doesn't consider the testing process separately and doesn't address all of the issues that are important for successful projects. Issues are not defined in this model. By focusing on a limited set of activities and working forcefully to complete them, an organization can steadily improve its organization-wide software process to enable continuous and lasting gains in software process capability

The Test Process Improvement (TPI) model has been developed on the basis of the practical knowledge and experiences of test process development [4]. A TPI model must observe a test process from different points of view, like use of test tools, and test specification techniques. In TPI model these are called Key areas. Every key area can be classified into Levels of maturity. In test process improvement, generally different steps are defined for determining the current situation of the testing organization and finding out the strength and weakness of the organization. Different checkpoints [5] are defined in order to cover the requirements on each level. Based on this determine the improvement strategies need to apply in the organization. Disadvantages of TPI is TPI model does not automatically lead to good analysis of the current and required situation and to improved test process.

Another test maturity model is Test Improvement Model (TIM) [6]. The Testing Improvement Model was developed because the developers thought that focus on test process improvement was necessary. According to the developers, TIM can be used to identify the current state of practice in key areas. TIM suggests methods to build the strengths and remove weaknesses. TIM consists of Maturity model and an Assessment Procedure.

Maturity Levels in TIM are Initial, Base lining, Cost effectiveness, Risk-lowering and Optimizing. It's mainly focusing five key areas they are Organization, Planning and tracking, Test cases, Test ware. Key areas determine the current maturity level of the organization

TMMI has a staged architecture for process improvement [7]. The development of the TMMI has used the TMM framework as developed by the Illinois Institute of Technology as one of its major sources. The key components are Maturity Model and Assessment Model. The TMM consists of 5 maturity levels that reflect a degree of test process maturity. Each maturity level has a number of Key process areas for defining the maturity level. A process area is a set of related activities within the test process [8].

The enterprise QA Transformation Model –A solution to enhance an enterprises' testing maturity [9], gives the answer to the need for a comprehensive test maturity model and defines the QA transformation roadmap and implements the changes suggested. It has been developed to help organization, selectively improves their testing capability based on key dimensions that contributes to testing maturity and can be customized to the business environment that the organization operates in.

In assessment procedure we can incorporate the Natural Language processing [10] using Neural Network [11]. Socher proposed the Recursive Neural Network (RecursiveNN) that has been proven to be efficient in terms of constructing sentence representations [12]. However, the RecursiveNN captures the semantics of a sentence by creating a tree structure. Its performance heavily depends on the performance of the textual tree construction.

The Recurrent Neural Network (RecurrentNN) model analyzes a text, word by word and stores the semantics of all the previous text in a fixed-sized hidden layer [13]. The advantage of RecurrentNN has the capability to capture the contextual information. This will be useful to capture semantics of long texts. However, the RecurrentNN prediction depends on the words which come later in the document, that are more dominant than earlier words.

To solve the problems occurred in RecurrentNN, the Convolutional Neural Network (CNN) is introduced to NLP tasks, which can fairly determine discriminative phrases in a text with a max-pooling layer [13]. Thus, the CNN may better capture the semantic of texts compared to recursive or recurrent neural networks. CNNs tends to use simple convolutional kernels such as a fixed window [14].When using such kernels, it is difficult to determine the window size [15]: small window sizes may result in the loss of some critical information, whereas large windows result in an enormous parameter space (which could be difficult to train).

Recurrent Convolutional Neural Network (RCNN)[16] can be used for the task of text classification. First, apply a bi-directional recurrent structure, to capture the contextual information to the greatest extent possible when learning word representations. Moreover, the model can reserve a larger range of the word ordering when learning representations of texts. Second, employ a max-pooling layer that automatically judges which features play key roles in text classification, to capture the key component in the texts [17]. By combining the recurrent structure and max-pooling layer, our model utilizes the advantage of both recurrent neural models and convolutional neural models.

3. PROPOSED WORK

The Proposed Model is an efficient model for evaluating the maturity level of a test organization. It has two components Maturity Model and Assessment Model. Maturity Model has 24 key process areas of testing organization that are mapped to seven vectors. Assessment Model has set of questions and an assessment procedure. Questions are related to different key process areas. Assessment procedure has a survey to collect the information from different staff, working at the organization in different roles. It also uses Natural Language Processing for categorizing the answers under different levels of maturity.

The objective of our approach is to introduce an efficient maturity model with NLP engine. Maturity model will find out the maturity level and current situations of the organization by evaluating the answers given by the testing staff in a test organization. For categorizing the answers into five levels and giving rating to the answers, we can use RCNN. By categorizing the answers we can find out the maturity level of each key process areas defined in the model. This provides an organization efficiently find out the problems related to each key process areas.

In order to achieve the best quality of software testing with Maturity, two components with artificial intelligence are introduced in our system Maturity Model and Assessment Model.

A. Maturity Model

Maturity Model has five well defined levels of maturity and twenty four key process areas. Every level describes maturity of testing process. Each level is characterized by their testing capability.

Three types of users are involved in the Maturity model for the calculation of the Maturity level test organization. Client who is going to test their products in the test organization is using this model for getting the maturity level based on their requirements. Test Assessor is initiating all the process internally. Testing staff is responsible for giving the information based on the questionnaire given in the assessment model.

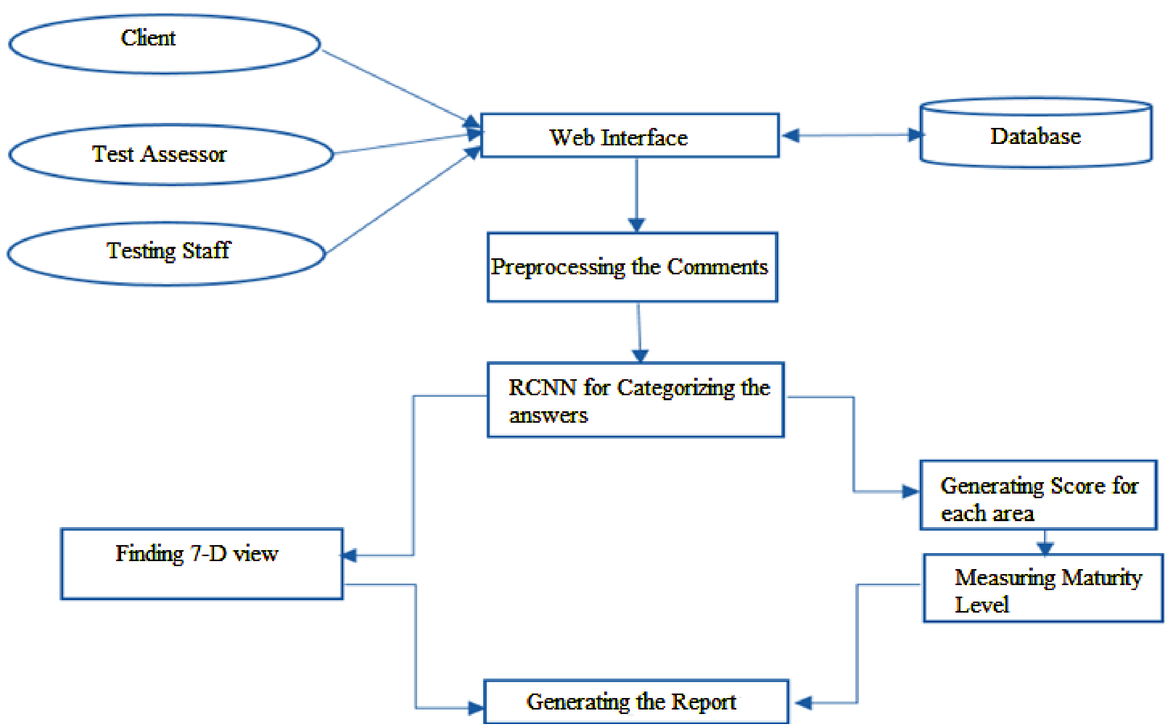

Figure 1: Architecture of Maturity Model

Figure 1 shows the Architectural structure of the Framework. Users are accessing the Framework through the web interface. Questions and answers are stored in the database. Answers submitted by the testing staff will categorize by the RCNN. Based on the category scores for each key process area is determined and Maturity level is measured. Dash board will show the 7-D view of the vectors and the report about the current problems and remedies are generated.

B. Assessment Model

Assessment model has a set of questions from each key process areas which are mapped to 7 vector (People, Process, Methodology, Tool, Technology, Domain, and Compliance) and the assessment procedure collecting the information about the current situations from the testing staff.

C. Assessment Procedure

Assessment procedure includes the submission of answers by each testing staff based on their roles and areas handling for the given questionnaire. Answers will be categorized by the recurrent convolutional neural network in the Framework.

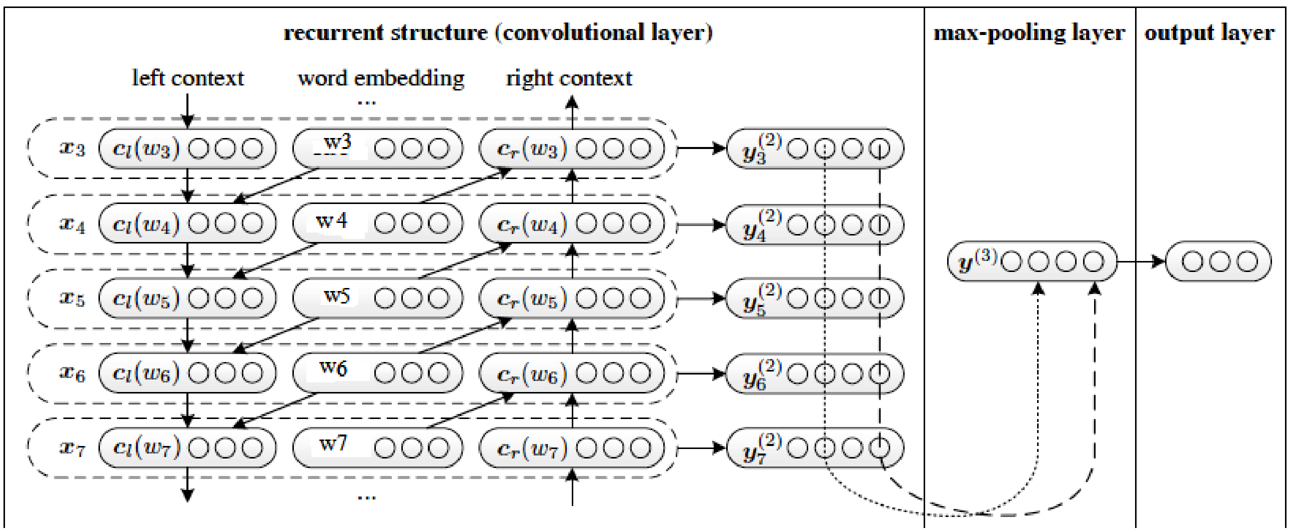

Figure 2: The structure of RCNN

Figure 2 shows the structure of RCNN. Different terms used in our neural network is $c_l(w_i)$ as the left context of word w_i and $c_r(w_i)$ as the right context of word w_i Both $c_l(w_i)$ and $c_r(w_i)$ are vectors. The left-side context $c_l(w_i)$ of word w_i is calculated using Equation (1), where $e(w_i w_{i-1})$ is the word embedding of word w_{i-1}, which is a dense vector with $|e|$ real value elements. $c_l(w_i w_{i-1})$ is the left-side context of the previous word w_{i-1}. The left-side context for the first word in any document uses the same shared parameters $c_l(w_i)$. $W^{(l)}$ is a vector that transforms the hidden layer (context) into the next hidden layer. $W^{(sl)}$ is a vector hat is used to combine the semantic of the current word with the next word's left context. f is a non-linear activation function. The right-side context $c_r(w_i)$ is calculated in a similar manner, as shown in Equation (2). The right-side contexts of the last word in a document share the parameters $c_r(w_n)$.

$$c_l(w_i) = f(W^{(l)} c_l(w_{i-1}) + W^{(sl)} e_{(wi-1)}) \tag{1}$$

$$c_r(w_i) = f(W^{(r)} c_r(w_{i+1}) + W^{(sr)} e_{(wi+1)}) \quad ((2)$$

Representation of word w_i in Equation (3), which is the concatenation of the left-side context vector $c_l(w_i)$, the word embedding $e(w_i)$ and the right-side context vector $c_r(w_i)$.

$$x_i = [c_l(w_i); c_r(w_i)] \tag{3}$$

The recurrent structure can obtain all c_l in a forward scan of the text and c_r in a backward scan of the text.

After we obtain the representation x_i of the word w_i, we apply a linear transformation together with the tanh activation function to x_i and send the result to the next layer.

$$y_i^{(2)} = \tanh(W^{(2)}x_i + b^{(2)}) \quad (4)$$

$y_i^{(2)}$ is a latent semantic vector, in which each semantic factor will be analysed to determine the most useful factor for representing the text.

When all of the representations of words are calculated, we apply a max-pooling layer.

$$y^{(3)} = \max_{i-1}{}^{n} y_i^{(2)} \quad (5)$$

The max function is an element-wise function. The k^{th} element of $y^{(3)}$ is the maximum in the k^{th} elements of $y_i^{(2)}$. The pooling layer converts texts with various lengths into a fixed-length vector. With the pooling layer, we can capture the information throughout the entire text. The max-pooling layer attempts to find the most important latent semantic factors in the document. The pooling layer utilizes the output of the recurrent structure as the input.

The last part of our model is an output layer. Similar to traditional neural networks, it is defined as

$$y_{(4)} = W^{(4)}y^{(3)} + b^{(4)} \quad (6)$$

Finally, the softmax function is applied to $y^{(4)}$. It can convert the output numbers into probabilities

$$P_i = \frac{\exp(y_i^{(4)})}{\sum_{k=1}^{n} \exp(y_k^{(4)})} \quad (7)$$

By using these probabilities we classify the text.

For training the recurrent convolutional neural network following algorithm is using.

Algorithm 1:

1. Read the dataset for training(five category of answers are there for training)
2. Tokenize each sentences
3. For each sentence
   (a) Find the left and right context of the word using linear activation function
   (b) Find the word embedding of the word
   (c) Find the word representation by concatenate the left and right context vector and word embedding
   (d) Apply linear transformation together with tanh activation function
   (e) Find text representation by using max function
4. Find the linear transformation by applying the basic neural network activation function
5. Find the probability value for classifying the sentence by using softmax function
6. Training network parameters

7. Maximize log likelihood with respect to parameters
8. Return the trained model.

For Classification following algorithm is using.

Algorithm 2:

1. Read the comments.
2. Tokenize the sentences.
3. For each sentence
   (a) Find the left and right context of the word using linear activation function
   (b) Find the word embedding of the word
   (c) Find the word representation by concatenate the left and right context vector and word embedding
   (d) Apply linear transformation together with tanh activation function
   (e) Find text representation by using max function
4. Find the linear transformation by applying the basic neural network activation function
5. Find the probability value for classifying the sentence by using softmax function
6. Return the class.

Once the answers are categorized we can start the scoring mechanism

D. Score Calculation

The calculation for finding the score given below:

We have seven vectors (People, Process, Methodology, Tool, Technology, Domain, and Compliance) each of them are mapped to any of the questions which are given in the database. For finding Score of each Vector:

Nv = Total No of Questions associated with that vector

Sv = Total Score given to that vector

$$Sv = (n0 \times 0) + (n1 \times 1) + (n2 \times 2) + (n3 \times 3) + (n4 \times 4)$$

$$\text{Rate of vector} = Sv/(Nv \times 4)$$

We have twenty four process areas and each questions are categorized under any of these area. For Finding Score for Key Area

Na = Total No of Questions in that process area

Sa = Total Score given to that Process Area

$$Sa = (q0 \times 0) + (q1 \times 1) + (q2 \times 2) + (q3 \times 3) + (q4 \times 4)$$

$$\text{Rate of Area} = Sa/(Na \times 4)$$

After calculating the score of vector, the proposed framework will create a seven dimensional view for the vectors in order to find out the percentage of influence of that vector in our test organization. From the score of each area we need to create a graph for finding the maturity level of each area and also find the maturity level of the entire organization. The proposed framework also generates one report mentioning the problems which are existing in the organization and gives improvement suggestions to remove them.

4. RESULTS

A set of questions and corresponding five category of answers are created for the development of an automatic system, with RCNN for the maturity evaluation of a test organization. Each area has different questions which are mapped to different vectors.

For the testing of the system we have added one user as client into our framework and selected all twenty four areas, different staff got emails to submit the answers based on these areas and roles. Since we have used RCNN for the evaluation process, each answers will be classified by the neural network. For this we have trained the RCNN by the answer set which we have already created under five category.

Framework analysis is carried out based on the report collected from a company by using the sample questions and answers given by them (Because of security purpose it is referred as company A).Now they are doing the assessment manually by a test assessment team and information collection is done by using a macro. Their report shows the maturity level of each process areas and 7-D view of vector. Report is generated based on the survey conducted internally in their organization in different area and a test assessment team valuate the answers submitted by the testing staff.

Table 1
Result Analysis

Process Areas	*Level in report*	*Level calculated*
Test Initiation	Level -1	Level -1
Change Management	Level-3	Level- 3
Test Execution	Level -2	Level- 2
Communication	Level- 2	Level -2

We have done the analysis by selecting the same areas and trained the RCNN with the previous answers they have given. Table 4 shows the selected areas and corresponding maturity level given in their report and calculated by our framework.

Figure 3 shows the 7-D view of the vectors of the company and Figure 4 shows the maturity level of each areas. Since we have given the same answers given by the testing staff in the company we got the same maturity level. Which means that our system works accurately. For the answers which are entirely different from the answers trained by the RCNN, then classification will be sometimes more difficult. So former usage we need to train the RCNN with more set of answers.

5. CONCLUSION

Proposed method give an efficient Maturity Model for evaluating the Maturity level of a testing organization. It is a web interface that anyone who are related to testing domain can interact with this model. Client can submit their requirements through this model. It will automatically select the answers and send them to the relevant testing staff in order to submit their answers for evaluating the current situations of the testing organization. Evaluation is done by the Assessment model. In which Recurrent Convolutional Neural Network has done the categorization of the data given by the testing staff. It is an efficient model that can accessible by anyone in the testing organization and evaluate the maturity level.

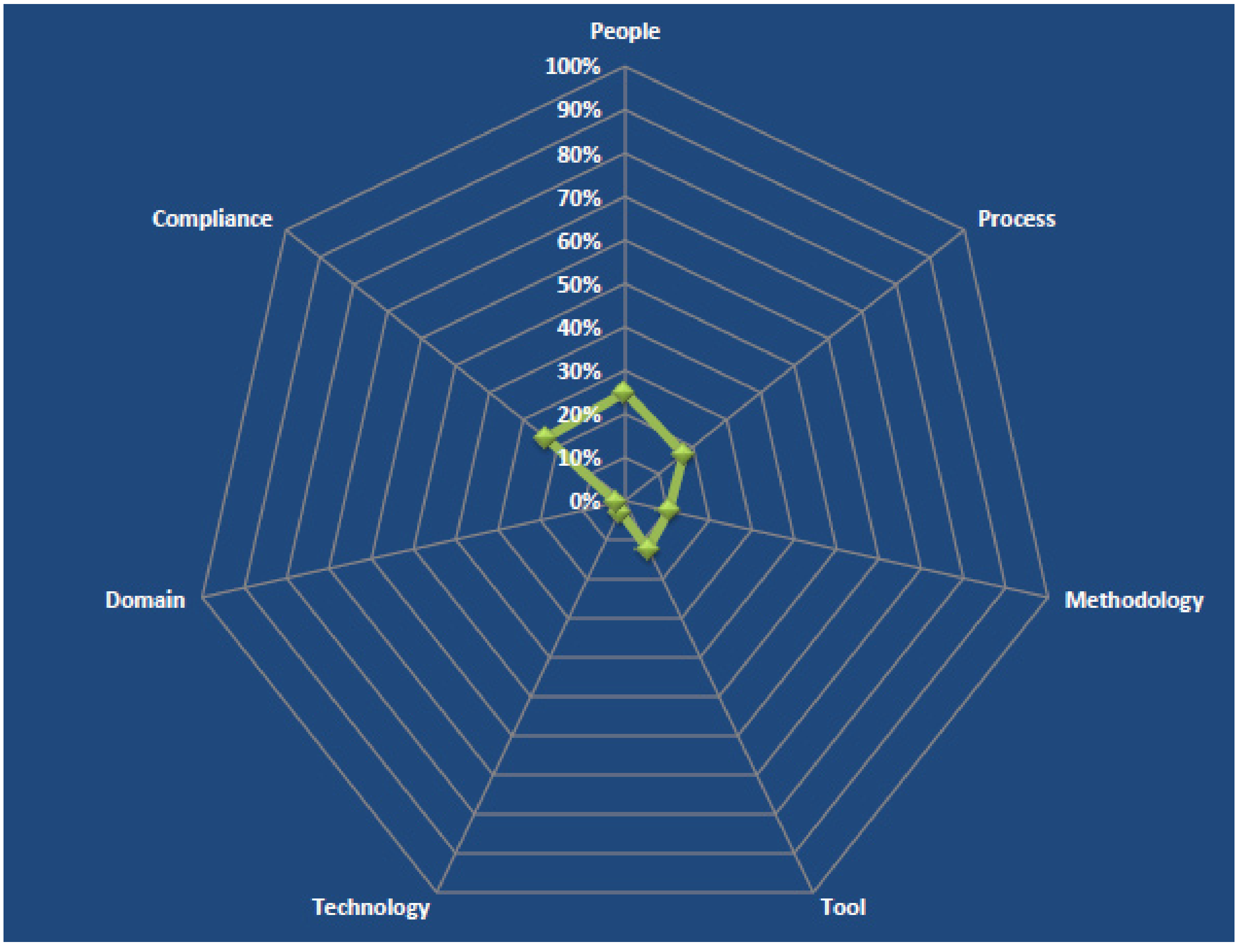

Figure 3: 7-D view of vectors

Figure 4: Testing Maturity Dashboard

REFERENCES

[1] Sukhdev Singh1, Gurmeet Kaur2, Parminder Kaur3 “Importance of Testing Maturity Model” Volume-8, Number-2 Jan-Jun 2015.

[2] Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, Charles V. Weber “Capability Maturity ModelSM for Software” February 2000.

[3] W.S. Humphrey “Characterizing the Software Process: A Maturity Framework”, Software Engineering Institute, CMU/SEI-87-TR-11, ADA182895, June 1987.

[4] Jari Andersin “TPI – a model for Test Process Improvement” 5th October 2004.

[5] Koomen, T. and Pol M., “Test Process Improvement: A practical step-by-step guide to structured testing”. ACM Press, London, England, 1999.

[6] Ron Swinkels, Frits Philips Institute, “A comparison of TMM and other Test Process Improvement Models.” November 2000.

[7] Burnstein I., Suwanassart T., and Carlson C., “Developing a Testing Maturity Model. Part 1”, Crosstalk, Journal of Defense Software Engineering, 9, No. 8, 21–24, 1996.

[8] Erik van Veenendaal “Testing Maturity Model Integration”, 1999.

[9] Reghunathan Balaraman, Harish Krishnankutty “Need for a Comprehensive Test Maturity Model”, External document 2 Human Computer Interaction and Data Mining015 Infosys Ltd.

[10] Abhimanyu Chopra, Athena Prashar, Chandresh Sain “Natural Language Processing” International Journal of Technology Enhancements and Emerging Engineering Research, Vol. 1, issue 4 131 issn 2347-4289.

[11] Ms. Shweta M. Patil, Prof. Ms. Sonal Patil” Evaluating Student Descriptive Answers Using Natural Language Processing” International Journal of Engineering Research & Technology (IJERT) Vol. 3 Issue 3, March – 2014.

[12] Socher, R.; Perelygin, A.; Wu, J. Y.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013. “Recursive deep models for semantic compositionality over a sentiment Treebank”. In EMNLP, 1631–Ť2.

[13] Elman, J. L. 1990. “Finding structure in time. Cognitive science “ 14(2):179–211.

[14] Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; and Kuksa, P. 2011. “Natural language processing (almost) from scratch”. JMLR 12:2493–2537.

[15] Yoon Kim “Convolutional Neural Networks for Sentence Classification” June 2014.

[16] Kalchbrenner, N., and Blunsom, P. 2013. “Recurrent convolutional neural networks for discourse compositionality”. In Workshop on CVSC, 119–126. Klementiev.

[17] Siwei Lai, Liheng Xu, Kang Liu, Jun Zhao “Recurrent Convolutional NeuralNetworks for Text Classification” Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence 2015.