Content based Component Retrieval based on Neural Network (NN) Classification Method

Rupali Garg
M.Tech. student,
Department of Computer Science, Punjabi University Patiala, India
Punjabi University, Patiala, India

Jagpuneet Kaur Bajwa
Assistant Professor,
Department of Computer Science, Punjabi University Patiala, India
Punjabi University, Patiala, India

ABSTRACT

CBIR systems mainly concentrates on reusability and development of software modules. This leads to less cost and enhancement of software module. Reusability is one of the factor that leads to good development of software. Software reusability is gaining interest because it leads to time reduction in software development too. So, reusability of variables has been adopted. Various methods has been developed for this, but in proposed work, the problem of component reusability is solved using neural network in addition to genetic algorithm, in which neural network will help out for retrieval of reused components.

General Terms

Component Retrieval, Genetic Algorithms, Best Component

Keywords

Components Retrieval, Neural Network, genetic algorithm, Software Engineering.

1. INTRODUCTION

1.1 Basic Description Of Software

Software is the set of trained tasks. Software is used in various applications e.g. application model i.e. words processor.

- Video Games
- Embedded Systems
- Operating Systems

All above mentioned applications need to be run without any error and must deliver QOS services. So, in this concern software has to be tested again and again correctly.

Software is the set of program code that is in executable form and fill various needs of computational model. Software when developed for particular application then it is called software product.

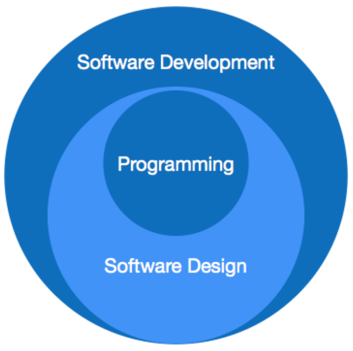

Fig 1: Software Paradigm

1.2 Need Of Software Engineering

The need of software engineering emerges as a result of higher rate of progress in client prerequisites and environment on which the software is working.

- **Large software -** It is simpler to fabricate a divider than to a house or building, in like manner, as the span of software turn out to be huge designing needs to venture to give it an investigative procedure.
- **Scalability-** If the software process were not taking into account exploratory and engineering ideas, it would be simpler to re-make new software than to scale a current one.
- **Cost-** As hardware industry has demonstrated its aptitudes and gigantic manufacturing has drop down the cost of PC and electronic equipment. In any case, the expense of software stays high if legitimate procedure is not adjusted.
- **Dynamic Nature-** The continually developing and adjusting nature of software immensely relies on the environment in which client meets expectations. In the event that the way of software is continually changing, new improvements need to be done in the current one. This is the place software engineering plays a worthy role.
- **Quality Management-** Better procedure of software improvement gives better and quality software product.

1.3 Software Reuse

Software reuse is the way toward making software frameworks from existing software as opposed to building them without any preparation. The product reuse perceived as having the capacity to enhancing software improvement profitability and software quality.

Reusability is one of the significant programming quality variables. Programming reuse is of interest since individuals need to manufacture frameworks that are greater and more mind boggling, more solid, less costly and that are conveyed on time. They have discovered conventional programming building techniques insufficient, and feel that product reuse can give a superior method for doing programming designing. There are two methodologies for reuse of code: build up the reusable code starting with no outside help or distinguish and separate the reusable code from effectively created code.

1.3.1 Component Based Software Engineering (CBSE)

It is a technique that plan to arrange and create software structures using reusable software sections. Software reuse is one of the standard motivations for CBSE. It focuses on reusing and changing existing fragments rather than making them with no arrangement. This diminishes both the change expense and effort and improves the way of the system.

The Component-Based Development, in like manner called change "with reuse", deals with the applications advancement. These applications are made by reusing existing segments. If necessary, new parts can be created, or even gained from outsider. An imperative issue in CBSE is that it is more attractive to reuse a current part than to build up another one, regardless of the fact that the prerequisites are not by any stretch of the imagination went to. This diminishes both cost and time to market, yet obliges a looking instrument to get to all the accessible parts. Along these lines, to be viable, a CBSE must give approaches to discovering, joining and adjusting existing segments [9, 10].

There are three stages in this procedure. These are given as:

- Component Qualification
- Component Adaptation
- Component Composition.

1.3.2 Reusability Benefits

1. *Increased dependability:* They are more reliable as they has been tested on various software's [6].
2. *Reduced process risk:* In case software development has to be made from scratch then more cost will be demanded.
3. *Effective use of specialists:* Doing of same projects with specific work and applicability will be done.
4. *Reliability and Safety:* Better reliability and safety for robustness.

1.4 Neural Network (NN)

Neural systems were begun around 50 years prior. Their initial capacities were misrepresented, throwing questions on the field in general. There is a late recharged enthusiasm for the field, be that as it may, as a result of new strategies and a superior hypothetical comprehension of their abilities.

1.4.1 Inspiration for neural systems

- Researchers are tested to utilize machines all the more adequately for errands right now comprehended by people.
- Typical Rules don't reflect forms really utilized by people
- Conventional processing exceeds expectations in numerous ranges, yet not in others.

1.4.2 Sorts Of Application

- Having a PC program itself from an arrangement of cases so you don't need to program it yourself. This will be a solid center of this course: neural systems that gain from an arrangement of cases.
- Advancement: given an arrangement of requirements and a cost capacity, how would you locate an ideal arrangement? E.g. voyaging salesperson issue.
- Characterization: gathering designs into classes: i.e. manually written characters into letters.
- Cooperative memory: reviewing a memory in view of an incomplete match.
- Relapse: capacity mapping
- Psychological science:
- Demonstrating more elevated amount thinking:
- dialect
- critical thinking
- Demonstrating lower level thinking:
- vision
- tryout discourse acknowledgment
- discourse era
- Neurobiology: Modeling models of how the mind functions.
- Arithmetic:
- Nonparametric measurable examination and relapse.
- Rationality:
- Can human souls/conduct be clarified regarding images, or does it require something lower level, similar to a neurally based model?
- Signal preparing: smother line clamor, with versatile reverberation scratching off, visually impaired source partition.
- Siemens effectively utilizes neural systems for procedure mechanization in fundamental commercial ventures, e.g., in moving plant control more than 100 neural systems carry out their occupation, 24 hours a day.
- Design acknowledgment, i.e. perceiving manually written characters, e.g. the present adaptation of Apple's Newton utilizes a neural net
- Prescription, i.e. putting away restorative records taking into account case data
- Discourse creation: perusing content out loud (NETtalk)
- Business, e.g. rules for home loan choices are separated from past choices made by experienced evaluators, bringing about a system that has an abnormal state of concurrence with human specialists.

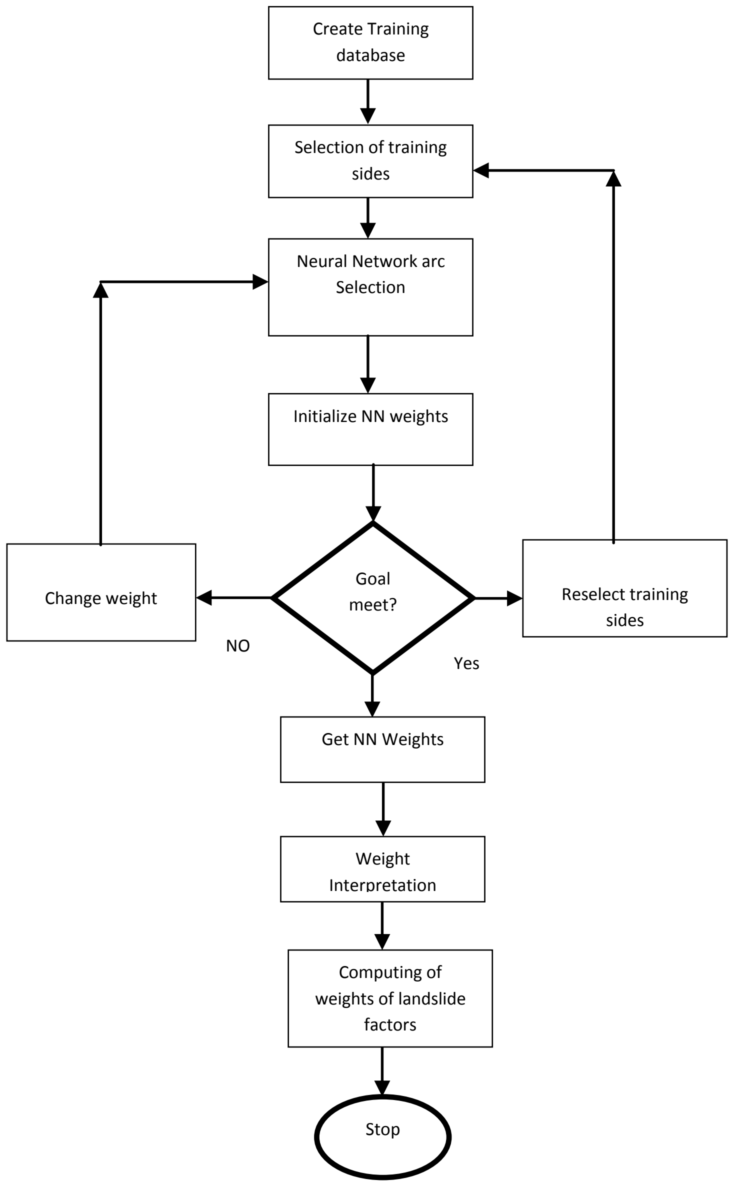

Fig 2: BPNN Working

2. LITERATURE SURVEY

This chapter describes the previous research work related to this topic.In this chapter different papers are studied and different types problems calculated.

N.Madaan, J.Kaur (2014) [4] presented the part of Requirement examination in segment choice process that helps the product engineers to model their necessities. It depicts the accompanying segment determination procedure that serves to choose the parts which can fulfill the necessities: **Cluster Based Selection Process:** It comprises of 3 stages: objective situated particular, reliance investigation, and bunch examination. Objective arranged particular uses an objective based necessities elicitation system to indicate Component based Selection (CBS) prerequisites).

A.Singh, Janhavi (2014) [2] proposed a software repository for storage and recovery of programming reusable segments. This vault executes two sorts of pursuit methods Keyword based hunt In this decisive word is entered to store and pertinent parts which coordinate the catchphrases are retrieved. User Priority-Based Component Retrieval: Select Component Type, Select Component Language, Select Component Domain and Select Component Name and the definite coordinating segment is recovered.

A.Bakhshi, S.Bawa(2013) [3] produced a product segment archive to store reusable component. Various look procedures are utilized to pursuit the obliged part and recovery of those parts. The procedures utilized for looking and recovering of parts are decisive word based hunt, mark based pursuit and Operational semantic catchphrase based recovery procedure is proposed which gives best recovery results. The outcomes acquired from the proposed method are contrasted and conventional magic word based strategy.

S.Singh (2013) [7] presented a meta-information show and faceted arrangement for capacity and recovery of programming parts that considers space semantic data in view of ontologies. As most existing storehouses recover a constrained arrangement of parts, the proposed meta-information model makes conceivable the proposal of interrelated segments, as philosophy attributes were consolidated. Three sorts of archives are used: Metadata vault stores data in unique area, give precise question terms, wipe out phenomena of same name with diff significance or distinctive name with same meaning. Describing storehouse give some data, for example, interfaces, capacities, authoritative levels, connected spaces, created dialects, connected situations, versions thus on so that hunt programming components. Component store -store segments and give a few administrations, for example, download etc.

P.Vohra, A.Singh (2013) [13] developed a product store which stores consequently, divided code on the premise of inputs and yields. Auto-Detect-Fragment-Store is an organized strategy which is utilized for the programmed stockpiling and recovery of source-code parts. Segment archive is developed which stores consequently, divided code on the premise of inputs and outputs.

S.Yadav, K.Kaur (2013) [5] proposed a calculation which is the mix of two most mainstream recovery strategies i.e. Essential word based methodology and Semantics based segment recovery technique. After this the recovered parts are positioned by past client requests. Thus, this paper proposed a proficient model for recovering the reusable parts in more streamlined way.

S.Gupta, A.Kumar(2013) [12] presented a meta-information display and faceted order for capacity and recovery of programming parts that considers space semantic data in light of ontologies and texonomies. Rather than most existing stores, which just recover a constrained arrangement of parts, the proposed metadata model makes conceivable the suggestion of interrelated segments, as metaphysics and scientific classifications attributes were joined. The product segment recovery taking into account feature order is a technique which has been broadly connected in programming part recovery, however the exactness of programming segment recovery is poor as an aftereffect of subjective consider faceted arrangement recovery. The structural engineering of programming part recovery framework and the model of programming segment recovery framework were composed, the relating match calculation was given. The consequences of use demonstrate that the new programming part recovery system can apparently enhance the segment recovery exactness and deal with the full-size of the seeking results.

C.Srinivas, V.Radhakrishna (2013) [8] proposed another methodology for bunching of reusable segments or archives by utilizing crossover XOR capacity which finds the level of comparability between any two parts or records. By applying mixture XOR capacity a framework named as comparability grid is characterized which is of request n-1 where n is number of parts in a given set. The proposed calculation takes the closeness network as information and gives yield as set of bunches of segments. The methodology can be supported as it completes extremely basic computational rationale and productive as far as handling with lessened hunt space and can be likewise be utilized as a part of general for report bunching or programming segment grouping.

A.Kaur, H.Monga (2012) [1] investigated auxiliary characteristics of capacity situated programming parts utilizing programming measurements and the measurements utilized are Cyclometric Complexity Using Mc Cabe's Measure, Halstead Software Science Indicator, Regularity Metric, Reuse recurrence metric, Coupling Metric. The estimations of these Metrics will turn into the info dataset for the distinctive neural system frameworks. Neural Network Based Approach is utilized to set up the relationship between diverse traits of the reusability and serve as the programmed apparatus for the assessment of the reusability of the methodology by ascertaining the relationship taking into account its training.

J.Sudhakarn, R.Vasantha(2011) [11] proposes another strategy for part grouping and recovery comprising K-closest Neighbor (KNN) calculation and vector space model methodology.

3. CONCLUSION

The software component retrieval storing system construct the model of software component retrieval, in the model, the software component can be store, searched and reutilized. A software component contains the entity, describing and metadata information in a software component repository. The entity, describing and metadata information can be stored together or discretely. The discrete scheme is more preferred as it is convenient for upgrade and maintenance, reducing burthen, improving openness, a component repository is partitioned into a describing repository and an entity repository.

The objective of this study is to classifying reusable software component using feed forward back propagation neural network as the training algorithm. The quickly convergence of training process as the solution approaches is the important reason for using NN algorithm. For this study, sigmoid, hyperbolic tangent functions are applied in the learning process. Feed forward back propagation neural network use to classify reusable software component according to reusable software component attribute and characteristic. Feature vector data cases are used to train the proposed network. After the completion of training process for the training data, the last weights of the network were saved to be ready for the testing procedure. Deciding reusable software component can be laborious. NN has been implemented for classification of reusable software component. The overall accuracy of classification in the training has been found to be 93.94%.

4. REFERENCES

[1] Anupama Kaur et al, "Identification and performance evaluation of reusable software components based Neural Network", International Journal of Research in Engineering and Technology 1.2 , 2012

[2] A.Singh and Janhavi, "Development of Component repository", International Journal of Advanced research in Computer Science and Software Engineering, 2014.

[3] Bakshi Amandeep, "Development of a software repository for the precise search and exact retrieval of the components", International Journal of Advanced Research in Computer Science and Software Engineering, 2013.

[4] Madaan Nitish and Jagdeep Kaur, "A Survey on Selection Techniques of Component Based Software".

[5] S.Yadav and K. Kaur, "Design of Rank Based Reusable Component Retrieval Algorithm", International Journal of Advanced Research in Computer Science and Software Engineering, 2013

[6] Sandhu Kanwaljeet and TrilokGaba, "A Novel Technique for Components Retrieval from Repositories", An international journal of advanced computer technology, 2014

[7] Singh Shekhar et al, "An experiment in software component retrieval based on metadata and ontology

repositor", International Journal of Computer Applications, pp. 0975–8887, 2013.

[8] Srinivas Chintakindi et al, "Clustering Software Components for Program Restructuring and Component Reuse Using Hybrid XOR Similarity Function", AASRI Procedia 4.

[9] Srinivas Chintakindi et al, "Clustering software components for program restructuring and component reuse using hybrid XNOR similarity function", Procedia Technology 12.

[10] Srinivas Chintakindi,et al, "Clustering and Classification of Software Component for Efficient Component Retrieval and Building Component Reuse Libraries", Procedia Computer Science 31.

[11] Sudhakaran et al, "A mixed method approach for efficient component retrieval from a component repository", Journal of Software Engineering and Applications , 2011.

[12] Suresh Chand Gupta and Prof. Ashok Kumar, "Reusable Software Component Retrieval System", International Journal of Application or Innovation in Engineering & Management, 2013.

[13] Vohra Pankaj and Ashima Singh, "Automatic Fragmentation and Storage of Code in Component Repository wrt their Input and Output Interfaces: A Tool", International Journal of Innovative Technology and Exploring Engineering, 2013.