International Journal of Science and Research (IJSR)
ISSN: 2319-7064
SJIF (2020): 7.803

Cognitive Computing in Fault Tolerance of Software

Kaushik I

Department of Computer Science and Engineering, National Institute of Technology, Karnataka, Surathkal, India
ORCID - 0000-0003-4258-2048

Abstract: ***This paper intends to give a brief description of cognitive computing research in the field of fault tolerance. We have referred over ten articles and research papers of various ongoing projects in this context for this paper to be authored. As always, we have tried to mention all the specific details of the algorithm that we intend to develop to tolerate the faultiness of the system and provide correction to the same.***

Keywords: Cognitive Computing, Fault Tolerance, Deep Learning, Neural Networks, Perceptron

1. Introduction

Fault tolerance, as we know, is the ability of the system or the software to acknowledge the error in the system itself to either ignore or correct the fault before it becomes a bigger problem to the system leading to the malfunctioning or the crashing of the system. In this era of fast-moving and fast development of software and systems, fault tolerance has be-come an essential development criterion. For this reason, this concept has become one of the most essential and problematic parts of the developers. On the lighter side, fault tolerance and its concepts are the reason for high-performance systems and software. Another important field of computer science is that of cognitive computing and deep learning. Cognitive comput-ing is the phenomenon of using human thinking capability to develop an algorithm that can lead to the implication of the artificial duplication of the same process in computing. These two vast concepts of computer science can be used to build a better algorithm that helps in a highly effective system for fault tolerance.

Cognitive computing deals with only two processes in the system. As in the brain neuron, the chemical process that happens only gives out either of the two results, “yes or no”. In deep learning, we use this procedure to develop the algorithm to proceed with the tolerating service. The concept of perceptron can be put to use here. As we have mentioned earlier, the result of the process in the neuron is either yes or no, so in this case, scientists have developed a virtual machine called a perceptron that does the same process as that of the neuron but with a bias as the weights of the quantities to which we compare and say yea or no, as in the case of perceptron its “case 1 or case 2”. The perceptron is a built concept from the human brain and uses much similar action of the neuron. The deep learning algorithm is solely based on perception, and its implementation has been regarded as one of the best yet. The algorithm focuses solely on the development of the system such that it can give us the correct and the possible choice from the previous cases of study by the algorithm. We are using this concept in the implementation and development of our algorithm.

2. Methodology

So, the question that comes to our mind is how this abstract concept can be implemented and, most importantly, where it can be implemented. In this field of software development, fault tolerance is a vast concept of development and error correction, as is cognitive computing and deep learning. As in this case, we develop a deep learning algorithm that inspects the system to process the implementation and provide us with the best solution to the system. Now, let us assume algorithm A is created with the same concepts as above; it now gains access to the system software implementation and processes the code or the UX to check for errors and bugs. This also happens only in the application of the system as the design is only in the application part and nowhere near the implementation part of the development of the system. Once it finds or comes across errors and bugs, it reports immediately to the user and asks for authentication to process the error to provide the solution.

The concept of fault tolerance is to try to tolerate or ignore the error if possible else correct it. The algorithm then processes the error and provides the user with a solution to either remove the implementation or continue with the code aspect. To develop such an algorithm, we need to provide the algorithm with errors so that the algorithm learns at a specific rate and can correct and detect faults in the system. As mentioned above, the concept of perceptron has two cases with bias on each based on the algorithm’s previous supervised learning. In this problem of detection and correction of the error, we use two cases is the case. One is to detect the error, but it can be ignored or tolerated due to its severity. Case two is that the algorithm discovers that the error or the fault cannot be fixed by the average user and has to be removed or quarantined immediately to prevent serious errors.

So, how does the algorithm know to proceed with case 1 or case 2? Well, the answer to this lies in the development stage of the system itself. In the development stage of the system, the algorithm is being connected to the main module of the system. Here in every stage of the system development, the algorithm goes through every stage of error the developers are encountering during

implementation. This includes the errors in the code, UX and UI problems, bugs, sematic errors, and various other errors in the system that occur. During this recording and learning of the errors, the algorithm also notes the developers' solution to the errors. This helps the algorithm build bias for each type of error occurring and, in the end, generate a report that contains all types of errors and might again occur in the future. The algorithm then proceeds to be in the module of the system as though it is a part of the system and continues to monitor the system's errors. So, if an error occurs, as mentioned in the above paragraph, the algorithm quickly registers it into the database and comes with a solution to the user and corrects the system in the process itself. Though the algorithm here is allowed to take its own decisions authentication system can be implemented to allow safer correction.

3. Theory

The theory behind this procedure is mainly the perceptron and its features. In the fault tolerance algorithm, we use the algorithm based on the perceptron handling request, which is the fault occurring in the system. So, in this section, we will deal with understanding perceptron and its purpose in this problem. A perceptron is an algorithm for supervised learning of binary classifiers. A vector of numbers or functions represents these. In mathematics, the field of ML is also contained by perceptron and its calculations.

A. Visualization of Perceptron

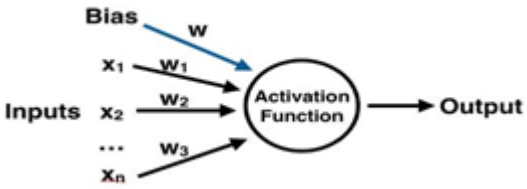

Figure 1: Function

In Fig-1, the set of inputs x1, x2, x3 are the faults that are occurring in the systems along with their weights (severity) to the function in the perceptron.

Function:

```
f(x) = ax + b
func NewPerceptron(n int32) *Perceptron {
        var i int32
        w: = make([]float32, n, n)
        for i = 0; i < n; i++ {
                w[i] = rand.Float32()*2 - 1
        }
         return &Perceptron{
                 weights: w,
                bias:rand.Float32()*2 - 1,
        }
}
```

The above algorithm states the perceptron's output based on the threshold of the weighted values of each fault that enters as a request into the system of the algorithm.

This algorithm is used mainly by all the perceptrons that run under the influence of bias-based systems and thresholds.

Here in the case of our problems, the bias of the fault is given based on the extensive study of the algorithm during the system's development process and learning the severity of each fault. Also, the algorithm learns the solution that the developers take action over the fault that has occurred. This allows the perceptron to build up a base and accurately predict the current fault's solution.

4. Efficiency

Now that we have implemented the algorithm, the question arises this process effective to some extent? The answer lies in the implementation itself. As we have mentioned that the algorithm goes through all the previous errors of the system that occurs in the development stage and also keeps track of all the processes that are happening in the system; the algorithm has a complete detailed description of the system in the form of a database generated by it. In this case of fault tolerance, it will contain all the type of errors and their severity of the error. Also, in the real-time application of the code system, there is a possibility of contradictions of errors that can be resolved using this algorithm.

So, as for the efficiency, though we are not in the stage of giving the numbers, the efficiency of the deep learning algorithm is very high and very much appreciated in this field of technological sciences. Deep Learning can be optimized using various accelerator algorithms and provide much better algorithmic outcomes if needed. Nevertheless, the algorithm itself is very efficient and can be in this context of fault tolerance.

5. Feasibility and Scope

This algorithm is just a concept yet to implement. But. Since the concept mainly revolves around the importance of fault tolerance and combining this with the algorithms in deep learning, this concept can be implemented in big organizations. Moreover, it can be used to its utmost efficiency by using various acceleration methods to optimize the perceptron's output. Also, the algorithm is very well known to scientists, and our algorithm needs only a bit of tweaking and changes to the previously existing algorithm.

The scope of this implementation is excellent. As mention already, fault tolerance is an essential component of develop-ment systems and cannot be ignored at all times as a whole. So, this implementation can help developers think of new ways to optimize the way we look at fault tolerance and help in the increase generation of better high-performance systems, and lead to the new evolution in the field of deep learning and cognitive computing.

6. Conclusion

After discussing the new concept of fault tolerance, we can find out that this method of tolerating errors is one of the best ways to correct and detect the software. Moreover, this can be implemented very quickly since it uses deep learning algorithms and some concepts of the perceptron to analyze the system and find the correct solution for the problem to the detected errors.

References

[1] J. Octavio Gutierrez-Garcia, Emmanuel Lopez´-Neri, Dept. of Com-puter. Science., Inst. Technology. Autonomo de Mexico, Mexico City, Mexico, Cognitive Computing – a brief Survey and open research challenges, IEEE Okayama, Japan, 12-16 July 2015
[2] Haluk Demirkan, Seth Early, Robert R. Harmon, University of Washing-ton Tacoma, Cognitive Computing, IEEE computer Society, 17 August 2017.
[3] Deepali Mittal, ASET, Amity University, Delhi, INDIA, Neha Agarwal, A review paper on Fault Tolerance in Cloud Computing, IEEE New Delhi, India,04 May 2015.
[4] S.Malik, M.J.Rahman, Fac. of Eng. Sci., Mohammad Ali Jinnah Univ., Islamabad, Pakistan, A framework for fault tolerance in distributed real time systems, Islamabad, Pakistan, 18-18 Sept. 2005.
[5] A. Manzone Centro Ricerche FIAT, Torino, Italy,A. Pincetti, D. De Costantini, Fault tolerant automotive systems: an overview, Taormina, Italy, Italy, 9-11 July 2001