IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

Building Automated Detection and Correction System for Syntax Program Error Using Development One Hot Vector Method

Dr. Buthainah F. AL-Dulaimi [1], Hameed Taha Khaleel [2]

Lecturer in Informatics Institute for Postgraduate Studies, work in the University of Information and Communication Technologies, Baghdad, Iraq [1]

Postgraduate in the Iraqi Commission for Computers and Informatics, Informatics Institute for Postgraduate Studies, Baghdad, Iraq [2]

Abstract: The aim of these papers is to propose a system that helps novice programmers to overcome the syntax errors by using automatic detection and correction technique. This technique is one of the important challenges facing researchers. The proposed system uses Hopfield neural network technic as an associated memory and the one hot vector as a method to represent the data. Where the number of iterations is adopted to reach of the stable state of the network to detect the syntax error and then correct them. The method of representation helps to increase the efficiency of the Hopfield technic and we will discuss this through two experiments. In the first experiment used the development one hot vector method as the matrix to represent the data. While used development one hot vector method as the one vector in the second experiment. The system is trained by using a case study containing a valid syntax statement. While it is tested by generating a set of errors in the statement for the case study. The case study is consists of several examples (Classes and Method) written in the Java programming language. These (Classes and Method) contains statement about the process of the multiplication and the arithmetic summation of the matrix. In the evaluation and testing process, random errors are created that related to a sequence of the tokens in the For-loop, If and While statements. The case study contains (205) statements that include (90) statements of the (For-loop statements) divided into (45) syntax error in the sequence of tokens and (45) correct statement.

Keywords: Correction syntax error, Detection syntax error, One Hot Vector Method, Java.

I. INTRODUCTION

The Programming is often cumbersome, especially if the person is a beginner and facing the syntax error. The interpreter or compiler of the programming language platform will display the error codes and determine their location. The beginner does not understand all the error messages. In our papers, we design a system capable of detecting and automatically correcting the syntax error for specific words such as for-statement without the need to understand error messages. This research focuses on syntax errors, to highlight the most common mistakes faced by the beginner [1]. The syntax error is pointing to the mistakes in order of token inside structure statement and punctuation. Frequently, messages syntactical error is not necessary to lead beginners to correct the error. Generally, most research converts the incorrect program to correct by using replacing, adding and/or deleting one or more symbols [2]. In our papers, we will compare the results of two experiments. Both experiments depend on Hopfield network technology and use the number of iteration for network stability as a measure of detection and automatic correction of errors. Network stability depends on the type of the encoding method. In these two experiments, one experiment uses one hot method and the other is used develop one hot vector method.

II. HOPFIELD NEURAL NETWORK (HNN)

The algorithm (1) describes the steps of network training, while algorithm (2) describes the network testing process [3,4]. Where training is done on a number of patterns and then the accuracy of the training is tested through the recognition of a pattern that has not been trained but is somewhat similar to the patterns that have been trained [5].

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

Algorithm (1) Training Hopfield Algorithm

- **Input:** fundamental memories (Y_1, Y_2 . . ., Y_p), The length of all vectors Y is equal (ith).
- **Output:** calculate weight matrix (W) between neurons , the weight matrix is symmetrical, main diagonal = 0.
- **Strategy:** calculated the weight matrix for (P) fundamental memories Y_p .
- **Steps of Storage:**
  1. IF i ≠ j then
     - $W = \sum_{i=1}^{p} Y_p * Y_p^T - P * I$ //I is n × n identity matrix
  2. ELSE $w_{ij} = 0$
- **END**.

Algorithm (2) Testing Hopfield Algorithm

- **Input:** unstable state of pattern (X_1, X_2 . . ., X_p), The length of all vectors (X) is equal (ith) that equal of the fundamental memory length (Y).
- **Output:** stable state pattern of (X_1, X_2 . . ., X_p).
- **Strategy**: the network recall fundamental memory Y_p when presented unstable input X_p or X is probe.
- **Steps of Testing:**
  - IF X_p=Y_p then // the number of patterns
  - X_k = Sign ((W.X_k) − θ) // k=1,2,…,P
  - Else // incomplete version or corrupted (probe)

a) Y(0) = Sign ((W.X(0) − θ)
// Initialize state X(0) is the element of the probe vector X and Y(0) is the state of neuron i , iteration (itr)= (0,1..itr) .

b) Y(itr+1) = Sign ((W.X(itr) − θ)
// update elements of stat vector (Y(itr)).

c) Repeat step (b) as long the state of the vector unchanged

- END.

III. THE ONE HOT VECTOR METHOD

In the neural network, the training requires data representation depending on the purpose of the training. In sequence classification, one hot vector is often used. One hot vector can be defined as a vector with only an element entry that corresponds to a particular letter is 1 excepting that all other inputs are zero [6,7].
Let $\{x^1, x^2 \ldots, x^\mu\}$ be a set of patterns and "μ" is index, the i-th element of each pattern is coded according to the following equation [8]:-

$$x_i^\mu = \begin{cases} 1 & If\ i = \mu \\ 0 & \text{otherwise} \end{cases} \quad \ldots\ldots\ldots (1)$$

In our papers, rather than having a vector containing one element of value (1), it is contained more than one value (1). So that each vector contains more than one variable depending on the syntax of the sentence and the element of matrix content from {1, -1} as illustrate in the algorithm (3).

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

Algorithm (3) Development one hot vector method Algorithm

- **Input:** the selective statement, number of parts and number of tokens without repetition μ and x_i represented the element in vector X
- **Output:** encoding matrix (encod_m) for statement
- **Strategy:** Converting the statement into a matrix consists of {1, -1}.
- **The Steps:**
  1. For i = 1 to (length of statement)
  2. For j = 1 to (length of part)
  3. IF $i = \mu$ then $x_i^{\mu} = 1$
  4. ELSE $x_i^{\mu} = -1$
  5. V+= element of vector X
  6. Next j.
  7. Save in encod_m[i][j].
  8. Next i.
- End.

IV. THE PROPOSED SYSTEM

The proposed system [4] will be implemented using two methods to represent the data as shown in figure (1) that illustrate the design of the system to perform the required operations according to the following steps:

1. Preprocessing data which as illustrated in tow steps.
   - Get the data from a notepad scripts.
   - Select the statement that starts with a specific keyword (For, IF, While).
2. Encoding the select statement based on developing one hot vector which illustrated in the algorithm (3) if the experiment1, while encoding the select statement based on equation (1) if experiment2.
3. Training Hopfield network as illustrated in the algorithm (1).
4. Generate potential syntax errors for each sentence of the case study.
5. Detection & correction syntax by apply testing Hopfield neural network is illustrated in the algorithm (2).
6. Decoding phase is illustrated in the algorithm (4).

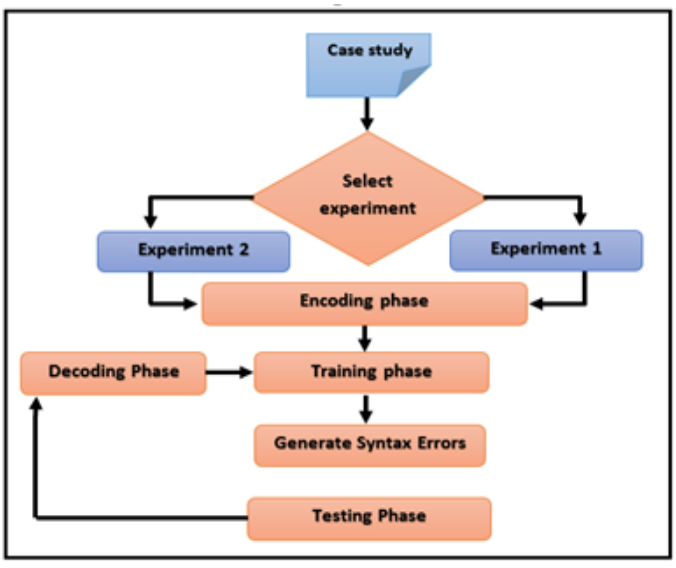

Figure 1: The structure of the propose system

In general, the number of tokens is calculated without repeating for the entire statement. Then determine of the number of bits that needed for encoding. That shown in figure (2).

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319-5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

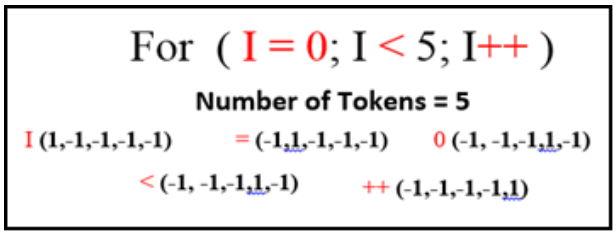

Figure (2): one hot vector method

In experiment (1), the part is the string between the two semicolons or between the arches and semicolon and the token is the word between two spices. Each part separated into three tokens. In the last block of the first phase, each part of the token will be converted into a sequence of {1, -1} depending on its position from the syntax and sequentially. Where are assigns (1,-1,-1,-1,-1,-1,-1,1,-1,-1,-1,-1,-1,-1,1,-1,-1) to the part1 from the statement, While assigns (1,-1,-1,-1,-1,-1,-1,-1,-1,1,1,-1,-1,-1,1,-1,-1) to the part2 from the statement, and the part3 of the token statement it (1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1, 1,-1,-1,-1,-1,-1,-1). While in experiment (2), encoding each token individually.

This process is repeated for each statement. The number of repetition is different from one sentence to another depending on the type of keyword that used in the code sentence. At last of pre-processing phase, returns matrix (15*3) diminution contains a sequence of {1,-1}in experiment(1) and returns matrix (5*9) in experiment(2) . These steps illustrated in the algorithm (3).

In the third phase, the algorithm (1) training Hopfield algorithm is applied for network training. While the test Hopfield algorithm (2) is applied and decode after generate potential syntax errors for each sentence of the case study.

V. THE DECODING DATA PHASE

The strategy of the decoding is based on the matrix that obtained from the developed test Hopfield algorithm (2) as an input. On the other hand to perform the matching operation and decoding [9]. In particular, the decoding process applied to each row individually. This process divides the row into (3) of sections to find the position of the token in the temp-table as shown in figure (3) by multiplying the element value of the column that is not equal to (-1) by the value of the row. Finally, all symbols are merge into the text variable (Whole-str) for each part.

Nu-line	Key-w	Id	Nu	Ar-op	Rel-op	Bit-op	Log-op	Ass-op	Split-token

Figure (3) structure of temp-table

VI. EXPERIMENT 1: IMPLEMENT HOPFIELD ALGORITHMS WITH DEVELOP ONE HOP VECTOR

In this experiment, For-statement is divided into three parts depending on the syntax of the For-statement. each part is separated by a semicolon. Where implement the develop one hot vector algorithm (3) to encoding statement. Each part is encoded as a row of the resulting coding matrix with (15*3) diminution contains a sequence of {1,-1}. Table (1) shown encoding training statement (for (x=0;x<=5; x++)).

Table (1) the encoding matrix for each part

Name of the part	Type of the Parts	Ex-part	the input matrix	weight matrix diminution
Part 1	Init-part	x=0	1,-1,-1,-1,-1, -1,-1,-1,-1,1, -1, 1,-1,-1,-1	(15*3)
Part 2	Con- part	x<=5	1,-1,-1,-1,-1,-1,-1,-1, 1,-1, 1,-1,-1,-1,-1	(15*3)
Part 3	Iter-part	x++	1,-1,-1,-1,-1,-1,-1, 1,-1,-1,-1,-1,-1,-1,-1	(15*3)

Where:

- Ex-part: indicates the example of the part statement.
- Init-part, Con- part, Iter-part: indicates initialization part, condition and,
- part, iteration part in respectively.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 7, Issue 2, February 2018

VII. EXPERIMENT 2: IMPLEMENT HOPFIELD ALGORITHMS WITH ONE HOP VECTOR AS ONE VECTOR

In this experiment, each tokens in For-statement encodes individually. Where implement the one hot vector method to encoding statement. Each token is encoded as a row of the resulting coding vector with one diminution contains a sequence of {1,-1}. Table (2) shown encoding training statement (for (x=0;x<=5; x++)).

Table (2)) the encoding matrix for each part		
Training example	**the input matrix**	**weight matrix diminution**
for (x = 0; x<=5; x++)	1,-1,-1,-1,-1,-1,-1,-1,-1,1,-1, 1,-1,-1,-1,1,-1,-1,-1,-1,-1,-1,-1,-1, 1,-1,-1,1,-1,-1,-1,-1,1,-1,-1,-1,-1,-1,-1,-1, 1,-1,-1,-1,-1,-1,-1,-1,-1,-1	[9]

VIII. RESULTS

To view the result of the proposed system, The case study contains (205) statements that include (90) statements of the (For-loop statements) divided into (45) syntax error in the sequence of tokens and (45) correct statement [10]. In particular, testing the "For keyword" syntax as the example for the proposed system.

In experiment (1), The pre-processing token will selective the whole sentence that starts with the "For as the keyword" successfully. Then, the implementation of the algorithm (3) development one hot vector method to produce a matrix with (15, 3) dimensions. While algorithm (1) training Hopfield is implementation for each row to result three weight matrix with (15, 15) dimensions. In the final phase, the test Hopfield algorithm (2) is applied and decode after generate potential syntax errors for each sentence of the case study.

While in experiment (2) , The algorithm (4) preprocessing token will selective the whole sentence that starts with the "For as the keyword" successfully. Then, the implementation of the algorithm (3) develop one hot vector method to produce a one vector with (45) dimensions instead of matrix with (15, 3) dimensions. While algorithm (1) training Hopfield is implementation for whole vector to result weight matrix with (45) dimensions.
In the final phase, the test Hopfield algorithm (2) is applied and decode after generate potential syntax errors for each sentence of the case study.

In the experiments, there are four important factors that depend on the efficiency of their performance: the number of the error statements that are detected, the number of correct statements that are detected, the number of error statements that are corrected and the number of correct statements that converted to the error statement. Table (3) shows the comparison of experiment 1 and experiment 2 and illustrated in figure (4).

Table (3) Comparison rate of correct recognition and failure recognition between experiments(1&2)					
No. experiment	No. statement	True detection rate	Failure detection rate	True correction rate	Failure correction rate
experiment 1	**90**	**100%**	**0%**	**80%**	**20%**
experiment 2	**90**	**78.88 %**	**21.11%**	**53.33%**	**46.66%**

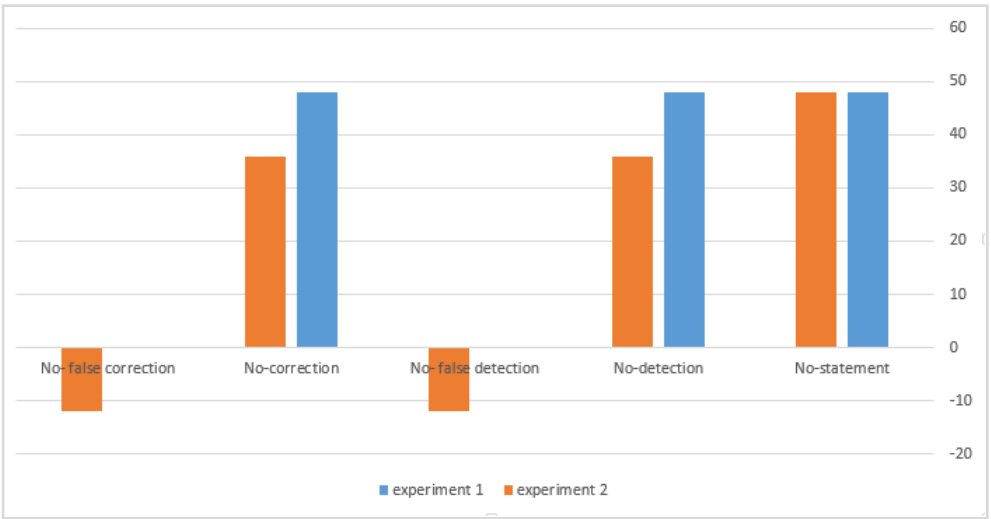

Figure (4) Comparison of experiment 1 and experiment 2

IX. DISCUSSION

The importance of our research is to help the students that trying to learn Java program language by designing a software agent capable of learning syntax. It can detect and correct syntax errors without trouble to understand the error messages from the compiler. Our system supports the use of neural networks to contribute to the detection and correction of error. This technique has the ability to remember knowledge from the learning stage and then address the problem within the limits of learning information. As can be seen, in the new methods of compiler design depend on machine learning techniques.

In the experiment (1), based on the concept of "divide and conquer" to the design of our system. The system is trained to break the syntax based on the symbol (;) and check the arrangement sequence of symbols for each part. We presented syntax of the statement "FOR" as an example because it consists of more than a syntax formula. In Table (3,4,5,6), the example consists of the basic syntaxes, which can be detected and corrected by so that all the arrangement sequence error of the tokens in the syntax are eliminated. Because of each part consists of three types of token {Identify, Logical comparison marks, numbers}, in this experiment the reach to the stable state from the one iteration indicates that the sentence is valid. While reaching the stable state after 2 of the iterations. This indicates that the sentence is wrong and has been corrected.

In the experiment (2), all statement encoding as one vector to testing it. This method affects the stability of the network, which leads to differences in the detection and correction results for the statement. The update of the value of any node in the network affects the stability of the network as a whole and therefore the greater the length of the vector is more likely to update the contract, leading to errors detection and correction.

X. CONCLUSION

The search is designed to correct the order of the keyword syntax sequence in the Java programming language by processing each token separately once and processing the sentence completely again.

Our system designed with two methods of encoding, One of them breaks down the sentence into several parts and encoding into the matrix and the other encodes the sentence as one vector.

The system is designed specifically to help the beginner programmers. Through, detection and correction syntax error according to Hopfield technique, regarding arrangement sequence of the tokens in the program code. We used the Java program language in building agent and as a case study. Because of, It is one of important language and most popular that deals based on object-oriented, class-based. The much of its syntax derives from C, C++ language. Ultimately, the beginner programmers in Java program language understand how to correct the some of the syntax error. Without wasting time to understanding the error messages of the compiler.

REFERENCES

[1] P. Denny, A. Luxton-Reilly, and E. Tempero, "All syntax errors are not equal," Proc. 17th ACM Annu. Conf. Innov. Technol. Comput. Sci. Educ., p. 75, 2012.
[2] M. Hristova, A. Misra, M. Rutter, and R. Mercuri, "Identifying and Correcting Java Programming Errors for Introductory Computer Science Students," *ACM SIGCSE Bull.*, vol. 35, no. 1, pp. 153–156, 2003.
[3] A. H. N. Network, "A Modified Hopfield Neural Network for Solving TSP Problem," pp. 1775–1780, 2016.
[4] M. Negnevitsky, Artificial Intelligence AGuide to Intelligent Systems, Second Edi. England: Pearson Education Limited, 2010.
[5] Dr. Buthainah F. AL-Dulaimi, Hameed Taha Khaleel " A Hopfield Neural Network Based Building Agent for Detection and Correction of Programming Errors", International Journal of Science and Research (IJSR), Volume 6 Issue 7, July 2017.
[6] Y. Goldberg, "A Primer on Neural Network Models for Natural Language Processing," arXiv.org, vol. cs.CL, p. 726, 2015.
[7] X. Hinaut, X. Hinaut, R. Neural, and S. Learning, "Recurrent Neural Network for Syntax Learning with Flexible Representations To cite this version : Recurrent Neural Network for Syntax Learning with Flexible Representations," IEEE ICDL-EPIROB Work. Lang. Learn., 2016.
[8] A. V. Uriarte-Arcia, I. López-Yáñez, and C. Yáñez-Márquez, "One-hot vector hybrid associative classifier for medical data classification," PLoS One, vol. 9, no. 4, 2014.
[9] C. Hillar, J. Sohl-Dickstein, and K. Koepsell, "Efficient and optimal binary Hopfield associative memory storage using minimum probability flow," arXiv Prepr. arXiv1204.2916, vol. 441170, no. 1, p. 5, 2012.
[10] R. Sedgewick, K. Wayne, J. Holcomb, C. Melville, and G. Entremont, Introduction to Programming in Java An Interdisciplinary Approach. Greg Tobin, 2007.