ProCAss: An Intelligent Assessment for Computer Programming Corpus

Kartinah Zen, Seleviawati Tarmizi, Dayang Nurfatimah A.I, Inson Din, Chua Sui Soon

Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak
94300 Kota Samarahan, Sarawak
E-mail: kartinah@fit.unimas.my

ABSTRACT

Electronic assessment (e-assessment) becomes more popular in educational tools especially in e-learning environment. This is because it has some advantages such as reducing the staffs needed for assessment tasks, automated marking is not prone to human error and it gives instant feedback to the students. However, the e-assessment existed only more on assessing essays and lack of implementation in assessing computer program in computer science environment [Haley, et.al, 2003]. This paper will briefly describe on the techniques that have been used to implement e-assessment for essays such as Project Essay Grade (PEG), E-rater, Intelligent Essay Assessor (IEA) and Latent Semantic Analysis (LSA). Mainly, the discussion will concentrate on LSA to build the e-assessment for computer programming corpus. One great advantage of LSA over the others is its ability to make absolute relative comparisons, where set of documents can be compared to each other, or to an answer schema. Another reason is the computer program produces programs in a subset of English-like words, a bit similar to an essay [Haley, et.al, 2003]. This paper will propose system design integrated with LSA method to assess student's programming assignments. Then, the ability of LSA algorithm in grading computer program corpus will be evaluated. The grading process will not limited on certain programming languages, but on any programming languages.

1.0 INTRODUCTION

The aim of this paper is to propose a design of automated marking computer program assignments using Latent Semantic Analysis (LSA). For many years researchers have done in-depth study in accessing the quality of free text questions especially essays. There are many approaches suggested by researchers regarding essay assessment since assessing students' writing is extremely labor intensive and time consuming. Some of the approaches are Project Essay Grade (PEG), E-rater, Intelligent Essay Assessor and LSA-based measurement (Miller, 2003).

LSA is chosen over the other approaches because of its ability to make absolute relative comparisons, where set of documents can be compared to each other, or to an answer schema by expert domains. This is a good criterion in building the model for assessing the computer program. Another strong reason to choose LSA is the computer programs written in high-level language produce programs in a subset of English like words, a bit similar to an essay [Haley, et.al, 2003]. However, the structure and logic in computer programs are different from an essay. Therefore, there is a need to design an approach using LSA to assess computer programs.

This paper is divided into four parts where the first part is an introduction, the second part is the description of other techniques, which done by other researchers to assess essays. These techniques are important in order to compare them with LSA. The technique of LSA, its algorithm and how LSA incorporated in computer programming assessment will be discussed in the third part of this paper. Finally, the conclusion and the problems raised will be presented.

2.0 PROJECT ESSAY GRADE (PEG)

Project Essay Grade (PEG) is developed by Ellis Page in 1965 to give scores to essays by computer (PAREonline.net, 2001). He expressed an intrinsic variable of interest and an obvious quantifiable variable, which correlates or approximates with one or more intrinsic variable. He named those two variables as trin, which represent intrinsic, and prox, which represent approximates. Specific attributes of writing style, such as average word length, number of semicolons, and word rarity are examples of proxes.

For any given trin, multiple regression analysis is performed on a randomly drawn sample of human-graded essays to determine the extent to which the proxes predict the human scores. The derived weights for each prox may be adjusted to maximize their power in multivariate prediction. The score for the trin in a previously unseen essay can then be predicted with the standard regression equation

$$\text{score} = \alpha + \sum_{i=1}^{k} \beta_i P_i$$

where α is a constant and $\beta_1, \beta_2, \ldots \beta_3$ are the weights associated with the proxes $P_1, P_2, \ldots P_k$.

Evaluation of the essay is often based on a comparison to others. PEG uses group of information to generate a statistical model. To assess essay content, PEG counts topic-specific keywords and their synonyms, which must be manually compiled for each essay set. The disadvantage of this system is it needs to be trained for each essay set used. Users must have access to many samples of pregraded essays. For a given sample of essays, human raters grade a large number of essays (100 to 400), and determine values for up to 30 proxes. (Rudner, et.al.,2001)

3.0 E- RATER

Jill Burstein from Educational Testing Service developed e-rater in 1990s. E-rater's basic technique is identical to PEG. The difference from PEG is e-rater uses vector-space model to measure semantic content. The vector-space model starts with a co-occurrence matrix where the rows represent terms and the columns represent documents. Term is usually a word and document is a sentence or a paragraph. The value in a particular cell is presented by a number indicating the frequencies of the terms occurred. Typically, each cell value is adjusted with an information-theoretic transformation. The weight of terms is calculated based on a formula. After the weighting, document vectors are compared with each other using some mathematical measure of vector similarity. (Miller, 2003).

4.0 INTELLIGENT ESSAY ASSESSOR (IEA)

Intelligent Essay Assessor (IEA) was first introduced for essay grading in 1997 by Thomas Landauer and Peter Foltz. (Rudner, et.al.,2001).
IEA arranged each calibration document as a column in a matrix. The technique of arrangement rows and columns is same as e-rater. Each essay to be graded is converted into a column vector, with the essay representing a new source with cell values based on the terms (rows) from the original matrix. A similarity score is then calculated for the essay column vector relative to each column of the rubric matrix. The essay's grade is determined by averaging the similarity scores from a predetermined number of sources with which it is the most similar.

5.0 LATENT SEMANTIC ANALYSIS (LSA)

Latent Semantic Analysis (LSA) is. a statistical corpus-based natural language processing technique that infers meaning from documents. It supports semantic similarity measurement between texts. By given a set of documents in a domain, LSA uses the frequency of occurrence of each word in each document to construct a word-document co-occurrence matrix (Kanejiya, et al).

Originally, LSA is being used for retrieving information before evolving to the system that are more fully exploiting its ability to extract and represent meaning (Haley, et al., 2003).

6.0 TECHNIQUES AND ALGORITHM IN LSA

In this section, LSA technique and algorithm is presented. Generally, the LSA model will be discussed which the model comprises of four steps (Landauer, T.K., et al., 1997, Jonathan, L.M., et al., 1999)

Step 1: A large body of text is represented as an occurrence matrix(***i x j***) in which rows stand for individual word types (terms), columns for meaning bearing passages such as sentence or paragraphs, that is <u>term x document</u>. Each cell then contains the frequency with which a word occurs in passage.

Step 2: Cell entries *frequency(i,j),* are transformed to:

$$\frac{\log(frequency_{i,j}+1)}{-\sum_{1-j}\left(\left(\frac{frequency_{i,j}}{\sum_{1-j} frequency_{i,j}}\right)*\log\left(\frac{frequency_{i,j}}{\sum_{1-j} frequency_{i,j}}\right)\right)}$$

a measure of the first order association of a word and its context.

Step 3: The matrix is then subject to Singular Value Decomposition (SVD). This means the document will be divided into three smaller matrices of a particular form (Miller, 2003). The first of these matrices has the same number of rows as the original matrix, but has fewer columns. These n columns correspond to new, specially derived factors such that there is no correlation between any pair of them. The third matrix has the same number of columns as the original, but has only m rows. In the middle is a diagonal n x n matrix of what are known as singular values. Its purpose is to scale the factors in the other two matrices such that when the three are multiplied, the original matrix is perfectly recomposed(Miller, 2003)

Step 4: Lastly, all but the *d* largest singular values are set to zero. Pre-multiplication of the right-hand matrices produces a least squares best approximation to the original matrix given the number of dimensions, d, that are retained. The SVD with dimension reduction constitutes a constraint satisfaction induction process in that it predicts the original observations on the basis of linear relations among the abstracted representations of the data that it has retained.

7.0 PROPOSED DESIGN

The proposed design in implementing technique and algorithm in LSA to a programming corpus is as shown in Diagram 1.

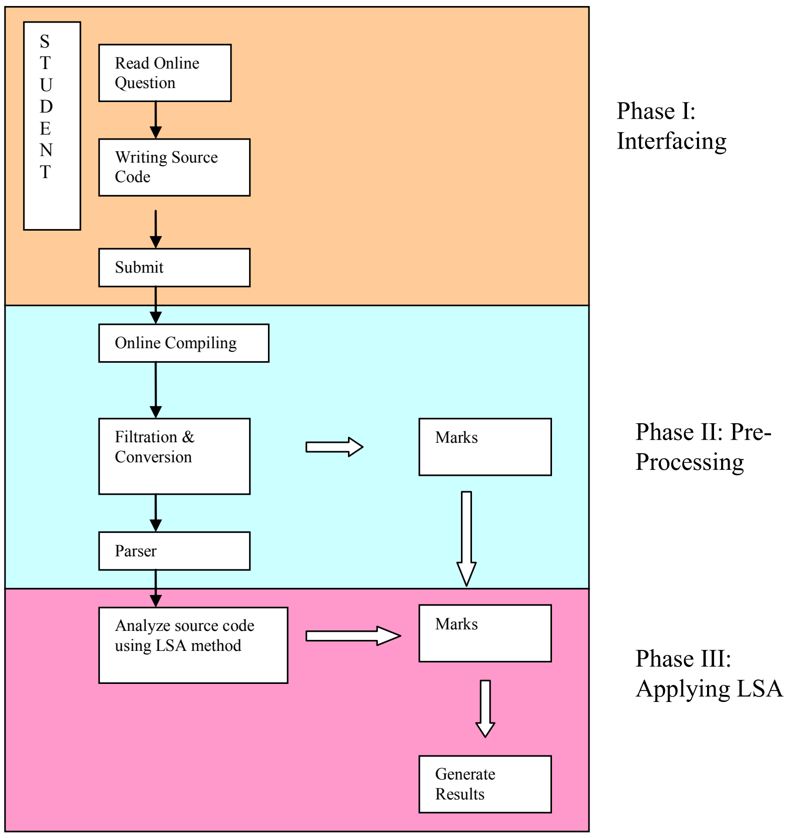

Diagram 1: Three phases in implementation of the technique namely interfacing, pre-processing and applying LSA technique

There are three main phases in implementation of the technique namely interfacing, pre-processing and applying LSA technique.

The first phase is where students will read, answer and submit online questions. This phase is the interfacing part, which captures and stores answers submitted by students.

Second phase of the implementation is Pre-processing. In this part, the corpus sent by students will be compiled to check for syntax errors. If document contain syntax errors it will be sent back to the student. Only corpus that is clear from syntax errors is further checked.

Next process in Phase II is Filtration & Conversion where the programming corpus will be filtered and convert to fulfill the requirement for LSA technique. Filtration and conversion process include clearing symbols used in programming such as / and []. Symbols required will then be examined by comparing with answer schema and will be marked. Further process is to compare and convert the variable declare and used in the answer piece. Variable name used in the answer may differ from what is in the schema. Therefore, this problem will be sorted out by converting all the variable names to suit the schema.

Next process is to solve the variable position problem that is public and private variables. The position of each variable declared will be examined and mark will be reduced for each variable declared incorrectly. The last process in filtration and conversion is to see the flow of the code whether logically put correct or not. After filtration and conversion process, filtered and converted corpus will be sent to the parser to LSA analyzer.

Some mark will be given to the programming pieces after phase II. The rest of the mark will be given after Phase III: Applying LSA Technique.

In Phase III, LSA technique will be applied to the code. Mark then will be determined based on the overall piece of the program such as BEGIN, END and so forth. At the end of the process marks from Phase II and III will be add up to come up with the total mark. Then, result is generated to students.

8.0 DISCUSSION

At the current stage of this research, feasibility study on using LSA in assessing computer programs is concerned. The ability of LSA technique to compare text similarity has been proved in grading essays. Therefore, LSA is expected to be able to perform the same technique on grading the computer programs. However, realizing that the nature and the structure of computer programs are different from essays, some aspects in perceiving computer program structure will be taken into consideration.

A few special aspects of computer programs structure are listed below:

- The symbols (i.e. $, @, &, etc.) might have a specific meaning in computer programs. So, it should not be eliminated. However, in grading essays using LSA technique, symbols are considered unnecessary and will be eliminated during text parsing. Therefore, using LSA on assessing computer programs should add a process to filter and convert computer programs to LSA's input texts that preserve the meaning of computer programs special characters or symbols.
- The nature of one computer program assignment to have different versions of answer is identical to an essay assignment. LSA can be fed with possible answer documents in the training corpus to evaluate an essay. Due to this fact, LSA is expected to accept a number of sample answers of computer program for training corpus in order to assess a computer program.
- It is a worry that in computer programming, strings, which are declared by a programmer or student, are different with the others including the assessor. This matter should not be overlooked since LSA is comparing and finding text similarities. In this situation, a string declared by a student which is written in different set of characters with the assessor will perceived to have a different meaning with the assessor's string, although they represent the same thing. As an initial suggestion, a parsing stage should be included to filter or convert the students' strings to which they can be perceive and accepted by LSA before text comparison.
- The variables' scope in computer programs can create problem in assessing computer programming. One variable name may be found in different procedures and functions in a computer programs and conceive different values. Yet, they have the same name. Logically, they will be considered the same by LSA, but actually they are not. Another problem is the local variable obviously different from the public variable, but LSA could not differentiate them because they are different by their position in program.
- The position of certain source code in a computer program is important in certain cases. On the other hand, LSA do not taken into account the position of a term in assessing an essay. Thus, this contradiction should be considered before using LSA to assess computer programs.

9.0 CONCLUSION

The nature of the computer programs and essays are a bit different regarding their semantic and logic. To overcome several problems mentioned, the proposed design was included with the filtration and conversion. The advantages of LSA where it can supports semantic similarity measurement between texts and the ability to extract and represent meaning rather than just comparing the texts could make the assessing process more accurate than the other techniques.

REFERENCES

Haley, Thomas, Nuseibehm Taylor, Lefrere (2003). *"E-Assessment Using Latent Semantic Analysis".* 3rd International LeGE-WG Workshop. Last retrieved on 26 March 2004 from http://ewic.bcs.org/conferences/2003/3rdlege/session3/paper3.htm

D. Kanejiya, S. Prasad, *"Automatic Evaluation of Students' Answers Using Syntactically Enhanced LSA".* Last retrieved on July 2004 from http://acl.ldc.upenn.edu/W/W03/W03-0208.pdf.

Landauer, T.K., Laham, D., Foltz, P.W., *"Introduction to Latent Semantic Analysis. Discourse Processes"*, 25, p259 – 284, 1998.

Landauer, T.K., Laham, D., Foltz, P.W., *"The Intelligent Essay Assessor^{tm} : Putting Knowledge to the Test".*

Miller, T. (2003), *"Essay Assessment with Latent Semantic Analysis".* A Journal of Educational Computing Research. Last retrieved on 26 March 2004 from http://www.cs.toronto.edu/~psy/essay_lsa.pdf

Lawrence Rudner, Phill Gagne, *"An Overview of Three Approaches to Scoring Written Essays by Computer"*,University of Maryland, College Park. http://www.pareonline.net/