ISSN No. 0976-5697

Volume 8, No. 3, March – April 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Checkpoint – An Online Descriptive Answers Grading Tool

Charusheela Nehete, Vasant Powar[#], Shivam Upadhyay*[#] and Jitesh Wadhwani[#]
Mentor, Student[#]
VESIT
Mumbai, MS India

Abstract: We live in an age of technology. Everything is automated. Even in the field of education, the use of technology is increasing largely. However, even today exams where theoretical questions need to be answered are taking place manually. This is because little work progress has been made in the field of grading theoretical answers written by students during examinations. Hence, we plan on creating an application that will help in evaluating answers. We call this application "Checkpoint". It is a natural language processing based descriptive answer checking and grading application. This application would simulate human thinking for assessing descriptive answers using natural language processing. NLP involves natural language understanding that is, enabling computers to derive meaning from human or natural language input, and produce the desired output. This application would first parse the answer that is given as an input to it. Taking into consideration for the presence of synonyms it will check how similar the given answer is to the ideal answer whose keywords will be provided by the teachers. Depending on the similarity, it will grade the answers.

Keywords: Exams, paper checking, assessment, online examination, grading, theoretical answers analysis

I. INTRODUCTION

Grading of descriptive answers is now done manually which is exactly what this project aims to change. Checkpoint aims at providing a way to grade descriptive answers so that even theoretical exams can be conducted online.

II. OBJECTIVE

To decrease workload of teachers, To provide a secure and easy way for students to write exams, To store marks of all students in an easy way, To emulate human way of assessment as much as possible.

III. NON-FUNCTIONAL REQUIREMENTS

- Server Interfacing
- User-friendly Graphic User Interface

IV. FUNCTIONAL REQUIREMENTS

- Accepting large inputs
- File processing
- Authentication
- Summarization
- Comparison

V. HARDWARE REQUIREMENTS

- Processor: Pentium II / III / IV
- Primary Memory: 64/128MB
- Hard Disk:1GB
- Monitor: Plug and play monitor
- Keyboard: 104 key
- Mouse: Logitech 3 button

VI. SOFTWARE REQUIREMENTS

- Operating System: Windows 7 /8 /10
- Web Browser: Mozilla Firefox, Google Chrome
- Database Server: SQL server 2008/ 2012
- Compiler: Python:3.5.2/ Python:2.7.12 , JDK 1.4/1.5/1.6/1.7

VII. WORKING OF THE PROJECT

The project will have four modules viz. Grammar Check, Stop Word Removal, Stemming, Comparison with keywords and Grading. This will be the core project which will be supported by other required modules like login authentication, admin panel which will be available only to teachers where they can enter questions and keywords. The core modules will be explained in the subsequent sections.

VIII. GRAMMAR CHECK

Students will write the answers through the GUI provided and it will be stored in the system. Each answer will have to go through a grammar check. Here spelling and grammar mistakes present in each answer will be checked. Following the general rule, we expect that the length of answer i.e. the sentences in the answer will be double the marks assigned to the question. For example, if the question is of five marks then the student will have to write an answer that has at least ten sentences. Our project allows one mistake per sentence. So if we encounter mistakes twice this threshold we mark this answer for review suspecting that the student has written something meaningless in the answer. Else we proceed towards the next module which is stop word removal.

IX. STOP WORD REMOVAL

Stop words are those words that are not relevant to the answer[1]. We searched online and found many lists that had stop words. We combined them and finally created our own stop words list. We excluded negative short forms like aren't, won't etc. so that we can perform contextual matching at later stage for which presence of negative words needs to be recorded. This module just went through the answer and deleted every occurrence of the stop words provided to it

through the list. This is also a crucial stage in learning whether the student has tried to exploit the system. It is possible that the student only writes keywords and not complete sentences. So if we don't find enough stop words in an answer, we mark it for review. Else, we proceed towards the next module which is Stemming.

X. STEMMING

Stemming a word means converting it to its root form, at least programmatically. Whatever remains in the answer is passed through stemmer provided by NLTK of PYTHON. Then all the words are present in their root form. We also stem the keywords given by the teacher. What this step does is ensures that all the keywords get matched irrespective of the way they are written [2]. For example, it is possible that the student writes "functional" while the provided keyword by the teacher is "functioning". After stemming, "functional" as well as "functioning" will be reduced to the same word "function" thus ensuring that the student will get marks for this particular keyword.

Before stemming, another factor needs to be considered. The keywords that are provided, it is necessary to first generate all possible synonyms and antonyms for them and then stem all of them. This is done to ensure proper comparison at a later stage. If we stem first, then we won't unable to find the synonym of the words as it's already been stemmed and it's structure has been changed. Hence first synonyms and antonyms are stored and then stemmed. We use python dictionary for this purpose. Why we do this, will be explained in the next stage i.e. Comparison with Keywords and Grading.

XI. COMPARISON WITH KEYWORDS AND GRADING

Now that we have all the words stemmed, in the answer as well as the keywords supplied, we have to compare them. We basically need to check for the presence of supplied keywords in the answer given by the student. Each keyword is associated with certain amount of marks as assigned by the teacher. According to that weightage, marks will be awarded. However, it's important to note that English being a versatile language, there are many possibilities of sentence construction and usage of words. So, not only we have to check for the presence of keywords but also for their synonyms. That is why we recorded the synonyms of the keywords in the previous stage. If we fail to locate a keyword, we proceed to find the presence of its synonyms. If that fails too, we next check for presence of antonyms. It is possible that the keyword maybe "advantageous" and the student has written "not disadvantageous". Hence, we check for presence of antonyms. If found, we check for the presence of a negative word like not, aren't, less, etc. in the same sentence where we found the antonym. If so, then again marks are awarded as per the weightage given.

XII. OBTAINED RESULTS

We created a sample paper and had a student named "Vasant" answer it. Then we processed those answers and had them graded. We also got a teacher to check the answers manually so that we could compare the results.

For example, the first question was "What is Java?"[3] to which the student had answered "Java is a programming language and computing platform first released by Sun Microsystems in 1995. There are lots of applications and websites that will not work unless you have Java installed, and more are created every day. Java is fast, secure, and reliable. From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere[4]. Java is an object-oriented language similar to C++, but simplified to eliminate language features that cause common programming errors. Java source code files (files with a .java extension) are compiled into a format called bytecode (files with a .class extension), which can then be executed by a Java interpreter. Compiled Java code can run on most computers because Java interpreters and runtime environments, known as Java Virtual Machines (VMs), exist for most operating systems, including UNIX, the Macintosh OS, and Windows."

The keywords that the teacher had supplied for this question were "programming language, fast, secure, reliable, object, bytecode, interpreter, independent, multithreaded, class" and the weightage assigned to each keyword was one. Since the question was of ten marks, it means that the each keyword will award the one mark. Since the student has only written seven of the keywords, he should have been awarded seven marks for this question as per the algorithm. The algorithm worked perfectly fine and the marks awarded were seven.

Similarly all the other answers written by the student were processed and graded as shown in figure 1 and the algorithm gave him 112 marks out of 120. Then we had a teacher grade the paper and the teacher gave him 110 marks. So we can say the algorithm produced an output which is sufficiently close to the actual output expected.

XIII. FUTURE WORK AND CONCLUSION

The algorithm proposed has a large execution time which needs to be reduced considerably for efficient execution. The answers are currently stored in text files. This can be replaced with something that takes less space. The logic of this algorithm is very naïve. It simplifies very complex tasks and does it in a very simple way. As a result, it takes too long to execute. Also, it cannot handle very complex sentence structures. English is a very confusing language to be analyzed completely. Same sentence can be constructed in a number of ways. So the method proposed for contextual matching, especially considering the antonyms, will not work for some complex sentence structures. Hence, the logic of contextual matching of this algorithm needs to be enhanced. Since, it was out of scope of the project, we haven't considered the fact that the students may be asked to write programming codes in the answers. To be able to check those, we would need to create a compiler and then compare the output and would require many other complex things. Also, students currently do not have any way, in this project, to draw diagrams. So a graphical user interface will have to be created and image processing will need to be used to check for the correctness of the diagrams drawn.

To conclude, this project forms only the basis of a functioning system. People can add onto it to create a fully functional grading system.

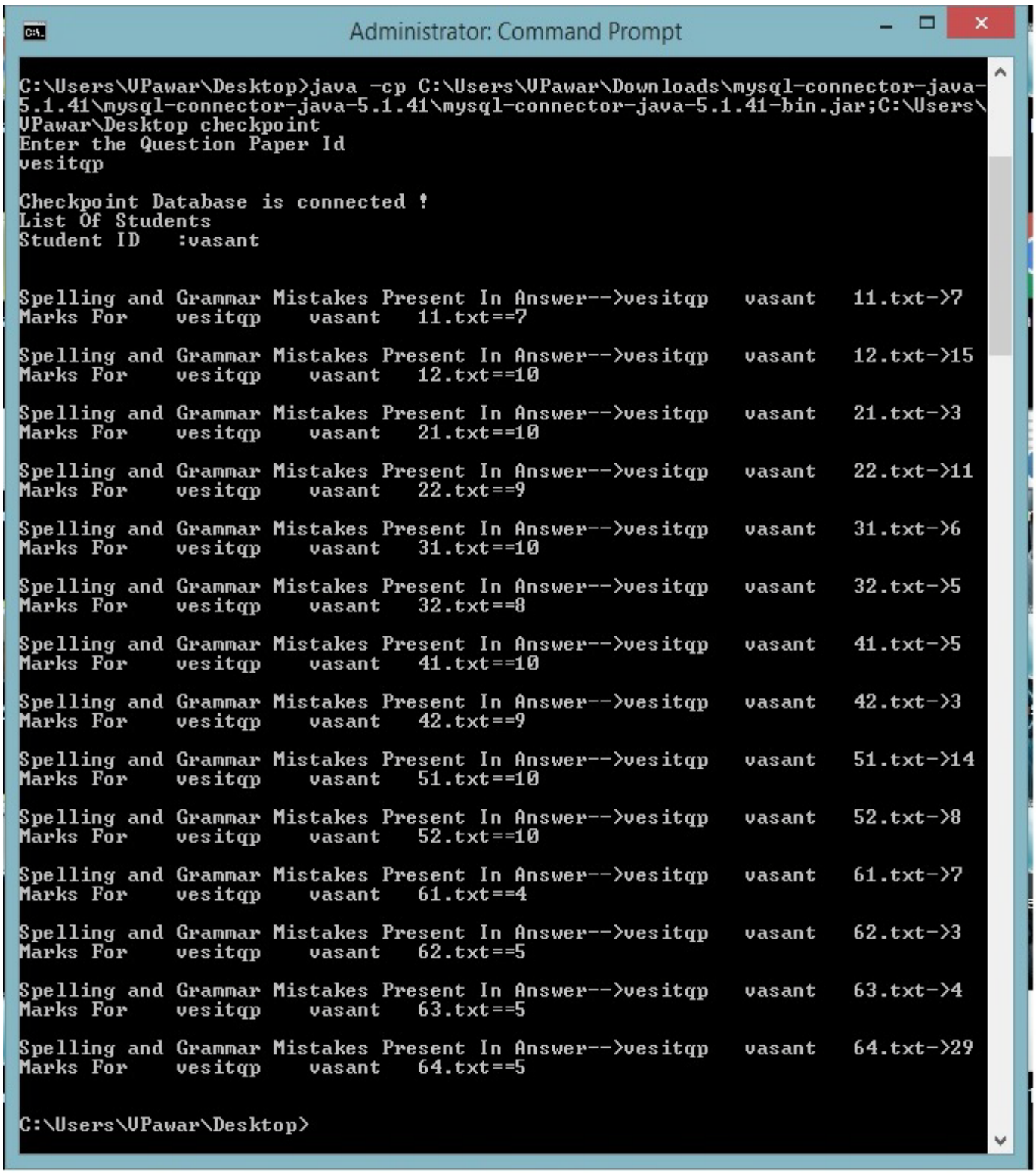

Fig. 1

XIV. ACKNOWLEDGMENT

We would like to thank our mentor, Mrs. Charusheela Nehete, who has been a great guide doing her best to help us out in every way possible. Right since the inception of this project idea, she has been with us ensuring we are on the right track so that we complete the project on time.

We would like to acknowledge the contributions of our head of department, Mrs. Shalu Chopra who ensured we had enough resources available so that we could work on the project.

Mrs. Vijaylaxmi, our vice-principal has also played a supportive part in completion of this project helping us in times of need.

XV. REFERENCES

[1] J. Talwar, S. Ranjani, Anwaya Aras, and M. Bedekar, Intelligent Classroom System for Qualitative Analysis of Students' Conceptual Understanding" , IEEE, 2013.

[2] HE Dehua, LU Yaobin, and ZHOU Deyi, Empirical Study of Consumers' Purchase Intentions in C2C Electronic Commerce, IEEE, 2008

[3] A. F. R. Rahman, H. Alam and R. Hartono. Content Extraction from HTML Documents, 1st Int. Workshop on Web Document Analysis(WDA2001), 2001

[4] Thorsten Joachims, Text Categorization with Support Vector Machines:Learning with many relevant features, IEEE,2005