Automated Exam Paper Marking System for Structured Questions and Block Diagrams

R.R.A.M. P. Jayawardena, G.A. D. Thiwanthi, P. S. Suriyaarachchi, K. I. Withana, and C. Jayawardena

Faculty of Computing, Sri Lanka Institute of Information Technology, Malabe 10115, Sri Lanka.
Email: chandimal.j@sliit.lk

***Abstract*—Exam paper marking has always been an exhaustive process which requires a lot of time and effort. We intend to address this issue by automating the marking of structured type questions and three diagram type questions, i.e. block diagram, logic circuit, and flowchart questions. A web-based system was developed to collect answers, mark and provide feedback to students. We investigated the application of Natural Language Processing (NLP) when marking structured questions. Graph matching using Depth-First Search (DFS) and Breadth-First Search (BFS) was employed in marking diagram type questions. The accuracy of logic circuit and flowchart diagram marking were tested by simulating and converting the diagrams into Python programs respectively. The implemented system has been made available to users as a plug-in in the opensource learning management system Moodle.**

***Index Terms*—Automated answer script marking, structured answers, block diagram answers, logic circuit answers, flowchart answers**

I. Introduction

Marking a large number of exam papers within a short period of time is a challenging task for teachers. Most answer scripts in schools and universities are currently marked manually by the examiners as marking of papers always require human expertise. Different answers can be written to express the same meaning and there may be subjective answers as well. If the answer script marking can be automated, the workload of examiners and the time consumption can be immensely reduced. It will reduce the total time taken for releasing results as well. It will also increase the marking accuracy as the accuracy of manual marking may go down due to tiredness, especially if the number of papers are large.

The system presented in this paper provides a solution to the above problem, within the scope of two types of questions: structured type questions and diagram type questions. Answers to structured type questions are short text answers, written inside a provided space. The diagram types considered in the system presented in the paper are block diagrams, logic circuits, and flowcharts.

The proposed system was implemented as a web-based system that provides exam paper creation and marking facilities to teachers. Teachers can create the four types of questions mentioned above using a web interface. For structured type questions, students get a pre-defined area to write the answer. In diagram type questions, both teachers and students are provided with a canvas for drawing the diagram. Structured type questions are marked based on the semantic similarity between the teacher's and student's answers. In diagram type questions, marks are given based on the availability of the required blocks as well as the logical relationships & connections between them.

In addition to the above features, the proposed system has been implemented as a plug-in to the Learning Management System, Moodle. These plug-in allows Moodle users (teachers) to create the above types of questions within the Moodle environment, in addition to the already supported question types of Moodle.

II. Literature Review

Various techniques have been used for marking structured type questions. There are already existing systems for marking answers given to simple questions, which do not require answers in the form of sentences. In the work presented in [2], a short answer marking method based on Inductive Logic Programming (ILP), Decision Tree Learning (DTL), and Bayesian Learning(BL) was presented. This work was successful, but it was limited to simple classifications similar to the "bag-of-words" approach, which is not sufficient for marking answers written in different styles using the same set of words. In this method, different sentences written using the same set of words, i.e. sentences with different meanings, will not be recognized, leading to low marking accuracy.

In the work of Siddiqi [3], a region-based approach was used to mark short textual answers. The authors have used an open source spell checker named Jortho to perform spell-checking and a statistical parser named Stanford Parser was used to parse the student's answer text. The output of the Stanford Parser, i.e. part-of-speech tagged text, was chunked into noun phrases and verb groups. The tagged and chunked text was compared with the syntactic structures specified in the Question Answer Language (QAL). In the work presented in [4], the semantic similarity score for a pair of sentences is calculated using WordNet.

Although there are some existing methods for marking textual answers, surprisingly, there are not many known existing systems for marking the three types of diagrams mentioned

978-1-5386-9418-3/18/$31.00 ©2018 IEEE

above and other diagrams, as pointed out in [5]. CourseMaster [6], [7] is one of the existing systems that can mark Logic Circuits and Flowcharts. However, there are no widely known systems for marking block diagram type answers.

There are several algorithms mentioned in literature which are used for graph matching. Ullmann's algorithm, VF2 algorithm are some of the exact graph matching algorithms [8], [9] and A-star algorithm ([10]-[14]) is one of the inexact error-tolerant graph matching algorithms. Binary Decision Diagrams, Ordered Binary Decision Diagrams, Simulation, and implication-based methods have been used in former work related to logic verification of logic circuits [15], [16].

III. Methodology

A. Structured type answer marking

Structured answer marking is done by comparing the teacher's answer and the student's answer by calculating the semantic similarity between the two. First, the student's answer and the teacher's answer are tokenized. Stanford Core NLP provides the ability to split the text into sentences and separate the words into parts of speech using Stanford CoreNLP Parts-Of-Speech (POS) tagger. Nouns, verbs, adjectives, adverbs, subject, object and the relationship between them can be identified using this POS tagger. The system ignores be, is, are, was, were, am, being, has, have, do, does, been, would, should, did, will, shall verbs when calculating Verb Similarity feature as the priority should be given to effective verbs in sentences.

Stanford Named Entity Recognizer is used for identifying the named entities which belong to following types; Person, Organization, Location, Currency, Percent, Date and Time. Stanford CoreNLP Dependency Parse Annotator (”depparse”) is used here to identify subjects and objects. As the next step, synonym sets are generated for all the nouns and verbs for both answers. Then the calculation of the semantic similarity score is done by continuously checking the words with each word in the related synonyms set separately until it gets a maximum score. To get this relatedness we used WordNet which is a large English lexical database. This method returns a value between 0 and 1. The Allocated marks for a particular question is changed depending on this returned value. The keywords must be provided by the teacher when creating a structured typed question.

Equation 1 is used to calculate the above mentioned semantic sentences similarities.

$$score = \frac{\sum_{i=1}^{n} NounScore}{n} + \frac{\sum_{j=1}^{m} VerbScore}{m} \quad (1)$$

Cosine similarity is used to check whether the students have included these keywords in their answer. In the process of checking cosine similarity, the system will check whether the exact word is there in the student's answer. The following cosine similarity values are calculated for a given sentence pair:

- Word Similarity
- Noun Similarity
- Verb Similarity
- Adjective Similarity

Equation 2 is used to calculate the above mentioned cosine similarities.

$$CosineSimilarity = \frac{\sum_{i=1}^{n} FV_{S,i} * FV_{T,i}}{\sqrt{\sum_{i=1}^{n}(FV_{S,i})^2} + \sqrt{\sum_{i=1}^{n}(FV_{T,i})^2}} \quad (2)$$

In Eq. 2, $FV_{S,i}$ and $FV_{T,i}$ represent frequency vectors of source sentence and target sentence respectively. The target sentence is the teacher's answer and the source sentence is the student's answer. Overlap ratio is used to get how many times a word appeared in the answer and length difference between the answers are also calculated to get the final mark for a particular answer. Finally, feedback is generated providing the missing key points for the student's answer.

B. Diagram type answer marking

To enable teachers and students to enter their diagram answers to the system, the proposed system provides a drag-and-drop canvas made using the GoJS diagramming pack under evaluation license. The answer diagrams are marked by constructing the graphs in the Neo4j database, and performing graph matching based on a novel approach. Mark allocation and feedback generation are done during graph traversal.

In Block diagram question answering, both the teacher and the student has to add a ‘start’ block and ‘end’ block to their answer. This ‘start’ block is used by the system to identify the point where answer graph traversal must start, thus, reducing the computational complexity otherwise required.

After discovering the start nodes, using DFS, both the teacher and student graphs are traversed simultaneously. Node by node is inspected in iterations where the child nodes of the current teacher node under analysis are compared against the set of child nodes of the current student node under analysis to find a match. Node matching is done by checking text similarity using WordNet from Natural Language Toolkit (NLTK). It is done by tokenizing the set of the words and part-of-speech tagging to identify Nouns, Verbs, Adjectives, and Adverbs recognized by WordNet when retrieving synonym sets for each word. Afterwards, the similarity value of the most similar word according to path similarity in WordNet hierarchy is taken for each word to form the final averaged score between zero and one. This score is used to match the nodes. The threshold value to be taken for the identification of a correct match was decided to be kept at 0.55 and above.

The additional, deleted, and substituted (graph edit - GE) nodes, and incorrect connections in the student answer are identified by our system.

A similar approach is taken to mark Logic Circuits using BFS by identifying the correct, and GE nodes. The inputs are processed in the same order. A gate is taken as a match only if the number of times it has been visited is equal to the number of inputs to it. In order to ensure that the next gate connected to the current gate is the exact gate as in the teacher's answer, we check whether the number of parents of the current node with and without any other children are the same. Additionally, to improve performance by reducing the looping involved, all the childless parents of the current node are removed, so that they are not analyzed again.

Simulation of a Logic Circuit is also done by using BFS. A binary combination list is generated by using itertools library in Python according to the number of inputs to the circuit. All inputs are processed in the same order for each binary combination, and when simulating both teacher and student answers. For each binary combination, simulation is run to generate the output for that particular combination. Current inputs to a gate are stored in the respective node in Neo4j. The gate is only taken for further graph traversal, when the number of stored inputs at the gate is equal to the number of input connections to the gate. The binary inputs stored in a gate are evaluated as per the gate to decide the output of the gate, and store it as an input in its child nodes.

Flowchart answers are marked by converting the answer and generating a Python program file with the help of graph traversal, and this file is used to test the program logic by providing the inputs given by the teacher to observe whether the desired output is given by the program. Marks are allocated for steps using DFS. Unlike in Block diagrams, taking the flowchart rules into consideration, the symbols will not be expected to be in a fixed position similar to the teacher's answer. For instance, if a teacher has taken an input before a decision, it is expected for the student to take an input before a decision, but the location of the input need not be fixed as long as it is before the decision. If the converted program does not return the expected output, marks will be given only for the correct input and output steps. In such cases, half marks will be awarded for decision steps if the student has identified the expected structure correctly, i.e. condition or loop.

Moodle is a well-known opensource learning management system widely used in universities and schools. Moodle plug-ins have been developed to integrate the proposed system with Moodle. Moodle plug-ins were developed using PHP for the existing Moodle users for our main features, enabling them to access the system's core functions. We provide the capability to generate API keys as well. PHP and Javascript are the languages we used in developing the front-end of our application while Flask and Spring Boot are used in making the Python and Java Web Application Programming Interfaces (API). MySQL is the database we used in holding all the application-related data and records. Neo4j database was used for the construction and easy traversal of graphs in graph matching of the diagram marking module. Py2neo and Pymysql are the Python libraries used to connect to Neo4j and MySQL databases respectively.

IV. Results

In order evaluate the validity of the proposed system, tests were conducted as follows.

A. Structured answer questions

For example, consider the question *What is a database?*. For this, teacher's and students' answers can be as follows:

- *Teacher's Answer*: A database is a collection of information that is organized so that it can be easily accessed, managed and updated.
- *S1*: A database is a collection of information which can be easily accessed, managed and updated.
- *S2*: A database is a collection of information.
- *S3*: An organized collection of data.

The semantic similarity, as generated by the system, of the three answers are shown in Fig. 1, Fig. 2, and Fig. 3. *S1* has got a similarity of 0.88 as it is very similar to the *Teacher's Answer* with same key words. In this case, the student should get closer to full marks for the question. Student answer *S2* has got a semantic similarity score of 0.55. This is an average answer and the student should get closer to half marks. In the case of *S3*, the semantic similarity score is 0.38 since it does not contain any key words.

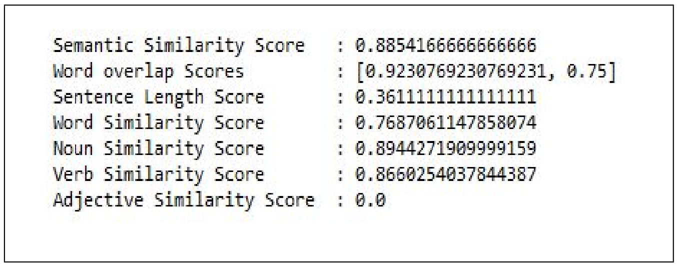
```
Semantic Similarity Score   : 0.8854166666666666
Word overlap Scores         : [0.9230769230769231, 0.75]
Sentence Length Score       : 0.3611111111111111
Word Similarity Score       : 0.7687061147858074
Noun Similarity Score       : 0.8944271909999159
Verb Similarity Score       : 0.8660254037844387
Adjective Similarity Score  : 0.0
```

Fig. 1. Text Semantic Similarity Results for First Student Answer

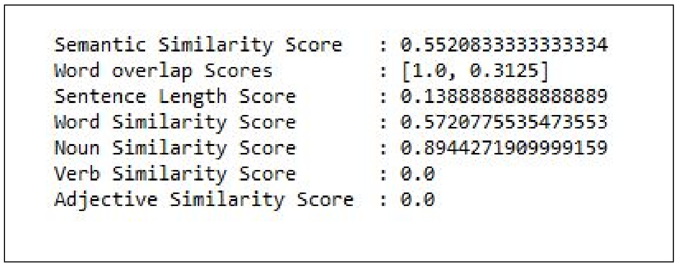
```
Semantic Similarity Score   : 0.5520833333333334
Word overlap Scores         : [1.0, 0.3125]
Sentence Length Score       : 0.1388888888888889
Word Similarity Score       : 0.5720775535473553
Noun Similarity Score       : 0.8944271909999159
Verb Similarity Score       : 0.0
Adjective Similarity Score  : 0.0
```

Fig. 2. Text Semantic Similarity Results for Second Student Answer

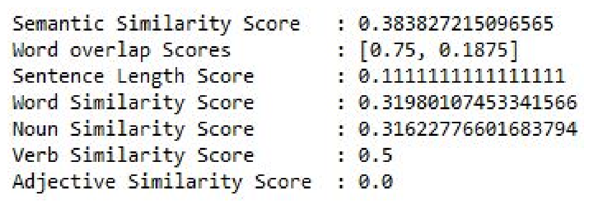
```
Semantic Similarity Score   : 0.383827215096565
Word overlap Scores         : [0.75, 0.1875]
Sentence Length Score       : 0.1111111111111111
Word Similarity Score       : 0.31980107453341566
Noun Similarity Score       : 0.31622776601683794
Verb Similarity Score       : 0.5
Adjective Similarity Score  : 0.0
```

Fig. 3. Text Semantic Similarity Results for Third Student Answer

B. Diagram type questions

We tested our diagram type question marking implementation (block diagrams and logic circuits) by hosting the web-based solution, and allowing the students to answer a set of questions. After the student answers were collected, the answers were marked using the system as well as by three teachers. We were able to collect 25 answers for the Block Diagram Questions and 85 answers for the Logic Gate Questions. Out of the 85, we were able to mark only 66 answers manually at the time of writing this paper. For both type of questions, difference of marks given by the teachers and by the system were calculated. These results are shown in Tables I and II.

Range of Difference (in marks)	Number of Answers
0 (same mark)	13
0.5	5
1 <mark <= 2	3
3 <mark <= 5	2
5 <mark <= 7	2
Mark >7	0

TABLE I
COMPARISON OF SYSTEM-ALLOCATED MARKS VS ACTUAL EXAMINER-ALLOCATED MARKS FOR BLOCK DIAGRAM MARKING

Range of Difference (in marks)	Number of Answers
0 (same mark)	47
0 <mark <= 1	3
1 <mark <= 2	5
2 <mark <= 3	3
3 <mark <= 5	4
5 <mark <= 7	4
Mark >7	0

TABLE II
COMPARISON OF SYSTEM-ALLOCATED MARKS VS ACTUAL EXAMINER-ALLOCATED MARKS FOR LOGIC CIRCUIT MARKING

If we disregard the 0.5 differences or 0 – 1 differences in marks, we can see that around 72% of the marks allocated for block diagram answers and 75.8% marks allocated for logic circuit answers using the system are equivalent to the marks allocated by the actual lecturer or teacher. It is also noticeable that while most of the rest of the marks fell under the ranges 1 - 2, 2 – 3, and 3 – 5, none of the marks generated by the system had a huge variance, or completely differed from the actual mark allocated by the marker. The test results did manifest a few scenarios where the system-allocated marks diverge from actual examiner allocated marks by considerable amounts as well.

Although the system is not 100% accurate in marking these diagram-type answers, it can be seen that a majority were marked accurately whilst the rest of the marks have not deviated by very huge amounts.

When it comes to the performance of the Logic Circuit answer marking, we monitored the time taken by the system for marking the answer using graph matching, and figured that it was much less compared to simulation. When graph matching took a time period around one second to mark an answer, simulation took around six seconds to mark the same answer. Therefore, we decided to allow the teacher to select whether the teacher wants an exact match done using graph matching, or whether alternative answers which will be marked using simulation are also to be allowed if an exact match could not be found, and this helped us improve the marking performance. It is also possible for the teacher to decide in using simulation to guarantee correct mark allocation regardless of the alternative answers, as graph matching may tend to give slightly different marks at times.

V. CONCLUSION AND FUTURE DIMENSIONS

This paper presents an automated answer script marking system for structured type textual questions and diagram type questions. For the structured question marking we used the semantic sentence similarity approach for language processing using various word senses of WordNet. This method depends upon all the semantic relation of word senses across the different synsets using WordNet for different part of speech of words. In addition to that, this method can be used to identify the semantic similarity among positive and negative sentences. In the diagram marking module, we have explored the application of DFS and BFS in graph marking for marking diagram-type questions. Nodes of Block diagrams were matched by checking semantic similarity using path similarity in WordNet hierarchy. Logic Circuits were simulated, and flowcharts were converted to programs to verify their logic. It is evident that we have achieved accuracy above 70% for both block diagrams and logic circuits. We intend to test flowchart marking after completion, and block diagram marking will be tested with more data. The current system can be expanded to support other types of diagrams like Entity-Relationship and Class diagrams as few systems exist in the diagram marking area. Additionally, we can strive in improving accuracy of Block diagrams and Logic circuit marking by investigating other scenarios that can enable more graph matching.

The proposed system has been made available as a Moodle plug-in. Including the other type of questions and improving the accuracy is an on-going research work of the authors.

REFERENCES

[1] Features: Automatic Test Marking - Course Toolkit LMS. [online] Available at: , *http://www.coursetoolkit.com/features?feature=automated-test-marking.* [Accessed 23 Mar. 2018].

[2] S. G. Pulman and J. Z. Sukkarieh, "Automatic Short Answer Marking.,"

[3] Raheel Siddiqi, "A Region-based Approach to the Automated Marking of Short Textual Answers," *SSU Res. J. Engg. Tech,*, vol. 1, no. 1, p. 7, 2011.

[4] M. A. Tayal, M. Raghuwanshi, and L. Malik, "Word net based method for determining semantic sentence similarity through various word senses," *in Proceedings of the 11th International Conference on Natural Language Processing,*, 2014, pp. 139–145.

[5] R. Stone, F. Batmaz and C. Hinde, "Drawing and Marking Graph Diagrams," *15 December 2015.[Online].Available: https://www.tandfonline.com/doi/full/10.11120/ital.2009.08020045,*, [Accessed May 2018].

[6] C. A. Higgins, P. Symeonidis, and A. Tsintsifas, "The marking system for CourseMaster," *ACM SIGCSE Bull.,*, vol. 34, no. 3, pp. 46–50, 2002.

[7] C. Higgins, P. Symeonidis, and A. Tsintsifas, “Diagram-based CBA using DATsys and CourseMaster,” *Proc. - Int. Conf. Comput. Educ. ICCE 2002,*,pp. 167–172, 2002.

[8] C. Djeraba, M. Revenu, A. Baskurt, and C. Wolf,“T hèse de l ’ U niversité de L yon I nexact graph matching techniques : Application to object detection and human action recognition,” *Proc. - Int. Conf. Comput. Educ. ICCE 2002,*,vol. 33, 2010.

[9] V. Carletti, P. Foggia, and M. Vento, “Performance Comparison of Five Exact Graph Matching Algorithms on Biological Databases,”*New Trends Image Anal. Process. - ICIAP,* , 2013.

[10] H. Bunke and G. Allermann, “Inexact graph recognition matching for structural pattern,” *Pattern Recognit. Lett,*, vol. 1, no. May, pp. 245–253, 1983.

[11] M. Neuhaus, K. Riesen, and H. Bunke, “Fast Suboptimal Algorithms for the Computation of Graph Edit Distance,” *no. Im,*, pp. 163–172, 2006.

[12] Z. Abu-Aisheh et al., “An Exact Graph Edit Distance Algorithm for Solving Pattern Recognition Problems,” *4th Int. Con- ference Pattern Recognit. Appl. Methods 2015,*,pp. 271–278, 2015.

[13] K. Riesen, “Structural Pattern Recognition with Graph Edit Distance,’ , pp. 29–45, 2015.

[14] R. Dijkman, M. Dumas, and L. García-Bañuelos, “Graph matching algorithms for business process model similarity search,” *th Int. Conf. Bus. Process Manag.,*, vol. Business P, pp. 48–63, 2009.

[15] W.Kunz, ““HANNIBAL: An Efficient Tool for Logic Verification Based on Recursive Learning”

[16] C.L. Berman, and L.H. Trevillyan, “Functional Comparison of Logic Designs for VLSI Circuits,’

[17] START”.[Online].Available:*http://start.csail.mit.edu/index.php.,*, [Accessed May 2018].

[18] Á. Rodrigo, J. Perez-Iglesias, A. Peñas, G. Garrido, and L. Araujo, ““A Question Answering System based on Information Retrieval and Validation,” *Noteb. Pap. CLEF 2010 LABs Work.,*,pp. 22–23, 2010.

[19] B. Ojokoh and P. Ayokunle, ““Online Question Answering System,” *Int. J. Comput. Sci. Res. Appl. Int. J. Comput. Sci. Res.*,Appl. ISSN, vol. 3, no. 3, pp. 2–9, 2013.

[20] pdfx v1.9”.[Online].Available: *http://pdfx.cs.man.ac.uk/.,*,[Accessed May 2018].