NLP-based Automatic Answer Script Evaluation

Md. Motiur Rahman[1] and Fazlul Hasan Siddiqui[2*]

[1]Dept. of Physical and Mathematical Sciences, Chittagong Veterinary and Animal Sciences University, Chittagong, Bangladesh
[2]Dept. of Computer Science and Engineering, Dhaka University of Engineering & Technology, Gazipur, Bangladesh

ABSTRACT

The answer script evaluation is an important part of assessing students' performance. Typically, an answer script evaluation is done manually that sometimes can be biased. The evaluation depends on various factors like mood swing of the evaluator, the inter-relation between the student and evaluator. Additionally, evaluation is a very tedious and time-consuming task. In this paper, a natural language processing-based method is shown for automatic answer script evaluation. Our experiment consists of text extraction from answer script, measuring various similarities between summarized extracted text and stored correct answers, and then assign a weight value to each calculated parameters to score the answer script. For summary generation from the extracted text, we have used keyword-based summarization techniques. Here four similarity measures (Cosine, Jaccard, Bigram, and Synonym) are used as parameters for generating the final mark. Automatic evaluation of answer scripts has been found very useful from our experiments, and often the assigned marks is the same as manually scored marks.

Keywords: Automatic Evaluation, NLP, Text Summarization, Similarity Measure, Marks Scoring

1. INTRODUCTION

There are various assessment strategies that are used to evaluate a student's performance. The most widely used technique is a descriptive question answering. In this technique, a student expresses his/her opinion in response to the question in a long textual way. The automatic descriptive answer evaluation system will be very cooperative for various universities and educational institutions to assess a student's performance very effectively [1]. A student may answer a question by following different grammatical styles, and chooses different words similar to the actual answer. The motivation behind the automated answer script evaluation comes from less time consuming, less manpower involvement, prohibiting human evaluator's psychological changes, and very easy to keep record and extraction [2]. It also assures that mood swings or change in perspective of the human assessor will not affect the evaluation process.

The automatic answer script evaluation based on Natural Language Processing (NLP) will help us to overcome the difficulties faced in the manual evaluation. Here a student's written answer is provided as input and the system will automatically score marks after the evaluation. The system considers all possible factors like spelling error, grammatical error, and various similarity measures for scoring marks. The natural language processing technique is used to make the handling of used English language much easier.

The natural language processing is an area of artificial intelligence which deals with the interaction between human languages and computer [3]. The most challenging task in natural language processing involves speech recognition, natural language understanding, and natural language generation. The NLP is widely used in machine translation, question-answering, automatic text summarization, answer script evaluation, etc. [3-4]. Text summarization helps to find out precise data from a longer text document, and speeds up the evaluation process.

The text summarization is a process of creating a short, accurate summary of the longer text. It is very time wasting task to generate a summary of longer article manually. Hence an NLP-based automatic text summarization technique is used to facilitate and speed up the text processing. Two types of text summarization techniques are used for generating the summary. The extractive text summarization technique is used to select phrases and sentences from the source document, and generates a new summary [5]. The abstractive text summarization technique is the opposite of extractive technique. It generates entirely new phrases and sentences to hold the meaning of the source document [6]. The NLP-based strategies are very well suited for generating summary rather than the manual process. The summarized text will be fed as input to compute various similarity measures.

The similarity measure is a technique to find how much two sentences are similar in the sense of semantic,

[*]Corresponding author's email: fazlul.siddiqui@duet.ac.bd

syntactic and structure. Similarity measure will enable us to decide the scoring marks to a answer script [7]. For measuring similarity, different algorithms, like the cosine similarity, Jaccard similarity, bigram similarity and synonym similarity are used [8]. The individual similarity measure algorithm defines a separate meaning. The cosine similarity between two documents generates a metric which tells how two documents are related by looking at the angle as a substitute of magnitude. The Jaccard similarity defines the similarity between two set of documents and it is computed by dividing the length of intersection by the length of the union of two document sets. The bigram similarity deals with the structure of two sentences and tells whether two are similar or not in respect of structure [9]. The synonym similarity tells how much two sentences are similar in respect of synonyms.

To make ease the manual evaluation process, automatic marks scoring has become very popular. Automatic marks scoring can be accomplished with the help of machine learning. In machine learning approach, some parameters are used to train a machine learning algorithm, and after training it can automatically assign score [10]. Another approach is assigning a weight value to the respective parameter, based on importance, and then multiply the parameter value and weight value. The summation of the above multiplication defines the marks of the corresponding answer.

For making the answer script evaluation system faster and effective, a digital method, based on NLP, is presented in this paper as automatic answer script evaluation.

2. BACKGROUND

Answer script evaluation is a very crucial part of student assessment. A teacher follows various ways like short question answering, descriptive question answering and multiple choice question to assess students [11]. The evaluation of multiple choice question and short question is easy and less time consuming, while descriptive question answering takes more time to evaluate. Several methods have been developed for automatic answer script evaluation. Some of them are mentioned in the following subsections.

2.1 Automatic Short Question Evaluation System

A vector-based technique for short question evaluation was performed by Ahmed Magooda et al. They observed sentence representation techniques and the wide range of similarity measures for the automatically grading question. For similarity measures, they considered string similarity, knowledge-based similarity, and corpus-based similarity. They used two different datasets to perform their proposed method and their proposed method was able to evaluate short questions with accuracy up to 86 percentage [12].

A simple short question evaluation method was developed by Md Arafat Sultan et al. They gave the short question, its correct answer as input and find the only semantic similarity of student response with respect to the correct answer. They also focused on short text similarity and augmented similarity. They computed performance of their model with Mohler et al. dataset and simpler bag-of-words model. They witnessed that their proposed model works better with the bag-of-words model [13].

Michael Mohler et al. developed a model for automatic short answer grading. They used unsupervised techniques for automatic short answer grading. They considered knowledge-based and corpus-based similarity, and the effect of domain and size of corpus [14]. They added automatic feedback from student answer in order to improve the performance. Their developed model outperformed than the previously proposed model. However, they did not take into account the grammatical and spelling error for grading.

Jonathan Nau et al. described a method for automatic short question answering for the Portuguese language [15]. They combined latent semantic analysis and WordNet path based similarity measure using linear regression to predict the score for short questions. They compared the predicted scores to human scores, which was found very useful in their proposed combined method.

P. Selvi et al. introduced a method for automatic short answer grading system, which is based on simple lexical matching [16]. They performed some comparison with existing method and they found that their proposed model worked well in few cases. It can grade short question with 59 percent accuracy.

2.2 Automatic Descriptive Question Evaluation System

The evaluation of descriptive question is quite difficult in comparison with short question evaluation. It takes more time to evaluate, and accuracy depends on various factors [17]. Hence, many researchers have proposed many methods for automatic descriptive answer evaluation. Some are presented below.

Shehta et al. developed a model for automatic descriptive answer evaluation [18]. They divided their proposed system in student module and tutor module. Their model takes the student answer and tutor answer as input and calculates the semantic similarity between two answer that helps to score marks. They used full NLP to implement their model. Their developed model doesn't fit for all type of data since they focused only on semantic similarity. There were some other factors that influenced the scoring marks.

A pattern matching algorithm based method was proposed by Pranali Nikam et al. for the assessment of a descriptive answer [19]. In their study, they represented the student answer and true answer in the form of a graph, and then they matched the pattern between the two graphs. They match each word of student answer with the true answer. If any word does not match with the true answer, then find the synonym of that word, match that synonym with the true answer. If matching found, replace the original word with synonym and compute the similarity. Here if two sentences are out of order with the same word, it gets confused and provide wrong scoring.

A text similarity-based method for automatic scoring of descriptive type tests was developed by Izuru Nogaito et al [20]. They measured n-gram and word-level matching similarity BLUE and RIBES respectively. They also calculated Doc2Vec based cosine similarity. They found that the most effective similarity measure technique depends on the type of question. Based on the question, the effectiveness of similarity measurement techniques is varied.

Marcelo Loor et al. [21] proposed a method with a combination of LSA, BLEU, WMD and FUZZY LOGIC. They used LSA to find semantic similarity between two documents. They used WMD to calculate the cumulative distance that a word needs to pass to reach the reference word. The cumulative distance measure distance even if there is no common word. Finally, they used fuzzy logic to score the marks. They applied their proposed model on the various dataset and found that accuracy varies between 0.71 and 0.85.

Most of the researcher focused on semantic similarity for scoring marks. They did not consider all other similarity parameters for deciding score. In this experiment, a noble approach is proposed with different similarity measure and used this similarity measures as the parameter. Finally, assign a weight value to each parameter based on importance to calculate the marks of that question.

3. METHODOLOGY

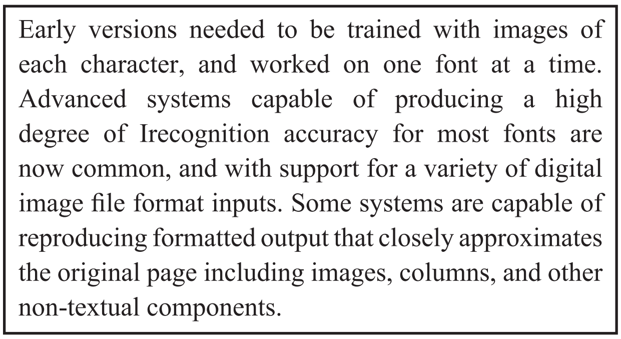
The aim of this study is to evaluate the descriptive answer script automatically and assign marks to this respective question. In order to accomplish this, we take answer script as input. Python programming language is used here for implementing every algorithm. Then NLP is used to extract text from the answer script and process the data. Various similarity measure has been calculated that is used as the parameter for assigning marks.

3.1 Text Extraction

The captured image from the answer script has been used as input for text extraction. For extracting text from the image, a python class pytesseract has been used. Before extracting text, the noise from the image is removed to increase the extraction accuracy. Pytesseract is a class based OCR and has Unicode (UTF-8) support, and can recognize more than 100 languages. The result of pytesseract is shown in Fig. 1. and Fig.2. The extracted text has been used for further processing and computes various similarity measures.

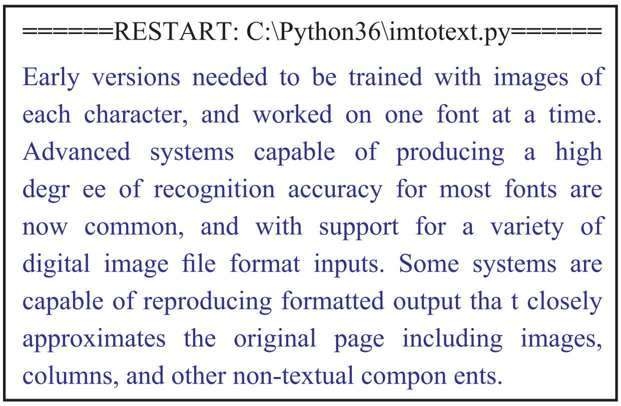
3.2 Summary Generation

From the image, the text is extracted as text format and natural language processing is used to make an automatic summary of the long text. Summary generation will help to speed up the text processing task by ignoring less important sentence from the long text document. Several techniques are available for generating auto-summary. In order to generate the summary of the long text, some keywords from the long text are selected based on the occurrence of the word. Here the average frequent words have been selected as keywords where the most frequent and less frequent word are ignored. Then the weight of each sentence in the text is calculated based on the number of keyword in sentence squared and divided by the window size. The window size is the maximum distance



Early versions needed to be trained with images of each character, and worked on one font at a time. Advanced systems capable of producing a high degree of Irecognition accuracy for most fonts are now common, and with support for a variety of digital image file format inputs. Some systems are capable of reproducing formatted output that closely approximates the original page including images, columns, and other non-textual components.

Fig. 1: Input image



```
======RESTART: C:\Python36\imtotext.py======
Early versions needed to be trained with images of
each character, and worked on one font at a time.
Advanced systems capable of producing a high
degr ee of recognition accuracy for most fonts are
now common, and with support for a variety of
digital image file format inputs. Some systems are
capable of reproducing formatted output tha t closely
approximates the original page including images,
columns, and other non-textual compon ents.
```

Fig. 2: Output text

between two significant words in a sentence. Then sort the sentence in descending order based on their weight value and finally take first n sentence as a summary of the long text.

Pseudocode of Text summarization algorithm

```
1. Take text as input
2. Tokenize the text into word
3. Remove duplicate from word list
4. Count frequency of each word
5. Calculate word percentage dividing word frequency by length of word list
6. Remove most frequent word and less frequent word by comparing word percentage with a max and min threshold value and select average frequent word as keywords
7. Count window size for each sentence with the help of keywords
8. Calculate weight of each sentence dividing square of no of keyword in sentence by window size
9. Sort the sentence in a descending order based on weight value and select first n sentence as summary
```

Another approach based on the bag-of-words ignoring keywords is also used. In order to find the effective technique for the summary generation, we have calculated Precision, Recall and F-score. The precision defines how much system summary (machine generated) is fact relevant?

$$Precision(P) = \frac{Number\ of\ overlapping\ Sentence}{Number\ of\ sentence\ in\ system\ summary} \quad (1)$$

The recall specifies how much of the reference summary (human generated) is recovering the system summary?

$$Recall(R) = \frac{Number\ of\ overlapping\ Sentence}{Number\ of\ sentence\ in\ reference\ summary} \quad (2)$$

F-score is the correlation measure that combines the precision and recall. The basic way to calculate F-score is to compute the harmonic average of precision and recall.

$$F\text{-}score = \frac{2.P.R}{P+R} \quad (3)$$

Here, the F-score of keyword based summary generation technique is greater than the bag-of-word based summary generation. Then the generated summary is compared with the true answer to find various similarity measure. Summary generation techniques and findings have discussed in result and discussion section in details.

3.3 Text Preprocessing

The Summarized text contains some word which carries less information and can be ignored to facilitate further text processing task. The way of converting data in a form that a computer can understand is known as preprocessing. The natural language processing is a very effective way to deals with the text preprocessing. Text preprocessing contains tokenize text into word, remove StopWord, lemmatize word, remove duplicate word etc. To accomplish this preprocessing using NLP, Natural Language Toolkit (NLTK) is a leading platform for building python program to work with human language data. It has the immensely built-in function to deals the text preprocessing by typing fewer commands. An NLTK built-in function word_tokenize is used to split the text into word and store in a list. The most important text preprocessing step is filter out the useless word. NLTK has a StopWord corpus which contains frequently occurred word those are useless to define the meaning of the sentence. The StopWord corpus has been used to filter out the unnecessary word.

Another text preprocessing step is word lemmatization. A word may appear in different form in many languages like a word walk may appear as walking, walked, and walks. Lemmatization is the process of converting the word into the base form which is known as the lemma. It will compress the length of the word list and save processing time. In order to lemmatize each word, an NLTK built-in function WordNetLemmatizer is used which convert all word into corresponding base form.

For carrying out some application over data, data need to be formatted in some common format. One kind of format is bigram or digram which is a sequence of two adjacent elements from the string of tokens. The bigram frequency distribution is commonly used to analyze the structural similarity of text. To generate bigram, bigram function of NLTK is used and it returns a list of bigram from all words. Here the frequency of each word also counts and stored in a dictionary where word used as key and store no of occurrence as data in the dictionary. Then the word dictionary with frequency and bigram are used for measuring various similarity.

3.4 Similarity Measure

In many cases, it is needed to define whether two sentences are similar or not. Similarity measures is a term which tells two sentences are similar or not by considering the different angle of similarity. Several similarity measure techniques are available that can be performed. In this experiment, cosine similarity, Jaccard similarity, bigram similarity, synonym similarity are performed.

Cosine similarity is very interesting similarity measure technique which looks at the angle of two documents and tells how much they are similar.

$$\textit{Cosine-similarity (A, B)} = \frac{(A, B)}{||A||.||B||} \qquad (4)$$

Where A and B are the word vector and each component of the vector contains word frequency or TF-IDF value. Here cosine similarity measure is carried out between student answer and true answer. Cosine similarity measure provides a very prominent result in terms of similarity. Cosine similarity has been implemented in this experiment in python language. The pseudocode is shown below.

Pseudocode of Cosine Similarity

```
1. Take dictionary of word and frequency as input.
2. Create two word vector where one for student answer and another for true answer. Length of each vector should be the length of total word list.
3. Calculate dot product of two vector
4. Compute norm of first vector
5. Compute norm of second vector
6. Multiply first and second norm
7. Divide dot product result by multiplication result and it will provide cosine similarity.
```

Jaccard similarity measure is another similarity measure technique which tells the degree of similarity by measuring the intersection and union of two word list.

$$\textit{Jaccard Similarity(A,B)} = \frac{|\mathbf{A} \cap \mathbf{B}|}{|\mathbf{A} \cup \mathbf{B}|} \qquad (5)$$

Where A and B are two word lists. Jaccard similarity is measured by dividing the intersection of two word lists with the union of that two word list. The intersection defines how much common word are between two word lists and the union defines total word in both lists.

Pseudocode of Jaccard Similarity

```
1. Take two word lists as input.
2. Perform intersection operation between two word lists. The AND (&) operation performs intersection.
3. Perform union operation between two word lists. Here add the length of two word list and subtract the length of intersection that is the union of two word list.
4. Divide intersection result by union result that will produce Jaccard similarity
```

In this study, the structural similarity between two documents is also taken into account. In order to compute the structural similarity, Bigram similarity measure has been performed. The pseudocode is presented in below.

Pseudocode of Bigram Similarity

```
1. Take two word lists as input
2. Generate bigram from two word lists. Bigram is the sequence of two adjacent tokens in string
3. Compute the no of common bigram in two bigram list.
4. Divide the no of common bigram by average bigram length of two bigram list.
5. Division will produce the bigram similarity.
```

In many languages, a word has many synonyms that hold the similar meaning. Hence, during the evaluation of the answer script, the synonym of the word have to consider for scoring marks. In this study, each word of student answer in matched with the true answer. If no matching word found in true answer, then retrieve all synonym of that word and again match with the true answer. To generate a synonym of a word, an NLTK wordnet function synsets is used. Synonym similarity is measured based on how much actual and synonym word of student answer is matched with the true answer and then divide it by average word length of two documents.

Pseudocode of Synonym Similarity

```
1. Take two word list as input
2. Match each word of student answer with true answer and count no of matching.
3. If there exist no matching word in true answer, then generate synonym of that answer.
4. Match each synonym on that word with true answer and count no of matching.
5. Divide the no of matching value by average length of two documents.
6. Division will generate synonym similarity value.
```

The efficient evaluation of answer script also depends on grammatical and spelling correctness. In this experiment, the grammatical and spelling mistake is also taken into consideration. To count the spelling and grammar error, a python package language check is used. The computed four similarity measure and grammatical-spelling error are used as the parameter for automatic marks scoring.

3.5 Marks Scoring

The one purpose of this study is to automatically score marks after evaluation. It is the final step of the experiment and the accuracy of this step will enhance the overall impact of this study.

Here, a weight value is assigned to each parameter based on the importance of the parameter. To improve the accuracy of assigning weight value, a survey study over 50 samples has been carried out. The average weight value estimation from the survey is accepted and applied.

$$marks = \sum_{k=1}^{n} P_k \, W_K \tag{6}$$

Where P_k is the kth parameter and $W_{k.}$ is the weight value of kth parameter. After assigning the weight value to each parameter, the weight value and the parameter value is multiplied. Then add all value of multiplication which is the final marks of that answer script.

In order to test our experiment, total thirty sample descriptive questions and the student answer to that question has been evaluated in a manual way. Three types of question in terms of marks are considered for this experiment. These are 5 marks question (M5), 10 marks question (M10) and 15 marks (M15) question It has been seen that most of the cases, our proposed method has scored score marks very near to manual judgment.

4. RESULT AND DISCUSSION

The goal of this study is to evaluate the descriptive answer script automatically and score marks. This will reduce the time for evaluating answer script and bring equality for evaluation. To satisfy those requirements, we used a weight parameter-based technique for automatic evaluation. The summary generation of extracted text plays an important role for the effectiveness of this experiment. For accepting an efficient technique for the summary generation, we have calculated F-score of the generated summary of two techniques with the comparison of reference summary. The estimated F-score of the keyword based summarization and bag-of-words based summarization are shown in Table I. Table I. indicates that the F-score of our used summarization technique is greater than the bag-of-words based summarization technique.

In this experiment, five parameters have been considered for scoring marks. These are synonym similarity, bigram similarity, grammatical-spelling error, cosine similarity and Jaccard similarity. These parameters are used to automatically evaluate three types of question (M5, M10, and M15) in terms of marks. The different weight value is assigned to each parameter based on question types. The weight assigned to each parameter are shown in Table II. The weight value is taken after averaging the survey value that for each parameter. From Table II, it is found in the survey that the importance of synonym parameter is more and grammatical-spelling error parameter is less for the evaluation of answer script. The high weight value indicates that the importance of that parameter is more for deciding the marks. The value of parameter comes between zero to one based on the similarity and presence of the error. The higher parameter value means the similarity between two documents is more and vice-versa. In this study, thirty answer script of three types of question is evaluated and marks are taken for testing the accuracy of the proposed model. Additionally, the above mentioned five parameters are calculated from that thirty answer script and used them for automatic marks scoring. The manual evaluated marks and auto-score marks are shown in Table III. From Table III, we see that our proposed automatic answer script evaluation system score marks very near to the manually scored marks. The comparison of automated scored marks and manually computed marks are shown in Fig. 3. From Fig. 3, we have found that there is a slight difference between automated scored marks and manually scored marks. Most of the cases the automated assigned marks and manually assigned marks are very close. When the student answer and the true answer contain more structural similarity as well as synonym similarity, the automated scored marks are very close to the manually scored marks. On the other hand, a notable difference between the automated scored marks and manually scored marks exist when the student answer and the true answer have less structural similarity while more Jaccard and Cosine similarity. It is also noticed from Table III. and Fig. 3 that the difference between the manually scored marks and automated scored marks is small for short question (M5) and opposite happens for descriptive question (M15).

Table 1: F-score Calculation

	Keyword-based Summarization	Bag-of-word based Summarization
Precision	0.9	0.83
Recall	0.83	0.41
F-score	0.86	0.53

5. CONCLUSION AND FUTURE WORK

In this experiment, we have developed a natural language processing-based method for automatic answer script evaluation and marks scoring. Our system consists of the following steps (1) text extraction from the image, (2) text summarization using keyword-based technique, (3) text preprocessing for further analysis, (4) finding various similarity measures, and (5) marks scoring. In the first step, the text is extracted using pytesseract which works based on OCR. Then the extracted text is summarized using keyword based summarize technique. Here we accept the average frequent word as the keyword and ignore most frequent and less frequent word. The summarized text is preprocessed with the aid of NLTK which is a leading platform for building python program. Here tokenization, stopword removal, lemmatization, bigram generation, and word frequency count are performed as a preprocessing. We also consider grammatical and spelling error for answer script evaluation. After preprocessing, four similarity measures – synonym similarity, bigram similarity, cosine similarity and Jaccard similarity measure are computed, which are used as the parameter for final marks scoring. In order to score marks, a weight value is assigned to each parameter after doing a survey on best weight estimation. The weight value is multiplied with parameter value to score final marks to that question. In this system, we have considered three types of questions based on marks, and the answer script based on that question is evaluated in a manual way. Manual marks are compared with automated scored marks to validate our developed method. In most cases, we have found that our proposed method scored marks similar to the manually assigned marks. It happens for a very few cases that the automated assigned marks are slightly higher or lower than the manually assigned marks. The limitation of our research is that we assign a weight value to each parameter manually by doing a survey. Therefore, our next goal is to introduce machine learning algorithm that will be trained by various calculated parameters, and algorithm will predict the marks of that answer script. Also in the future, we will introduce some new techniques for effective and precise summary generation.

REFERENCES

[1] V. Nandini, P. Uma Maheswari, “Automatic assessment of descriptive answers in online examination system using semantic relational features”, The Journal of Supercomputing, 2018.

[2] D. V. Paul and J. D. Pawar, “Use of Syntactic Similarity Based Similarity Matrix for Evaluating Descriptive Answer,” 2014 IEEE Sixth International Conference on Technology for Education, Clappana, 2014, pp. 253-256.

[3] K. Meena and L. Raj, “Evaluation of the descriptive type answers using hyperspace analog to language and self-organizing map,” 2014 IEEE International Conference on Computational Intelligence and Computing Research, Coimbatore, 2014, pp. 1-5.

[4] Y. Oganian, M. Conrad, A. Aryani, K. Spalek and H. R. Heekeren, “Activation Patterns throughout the Word Processing Network of L1-dominant Bilinguals Reflect Language Similarity and Language Decisions,” in Journal of Cognitive Neuroscience, vol. 27, no. 11, pp. 2197-2214, Nov. 2015.

[5] S. R. Rahimi, A. T. Mozhdehi, and M. Abdolahi, “An overview on extractive text summarization,” 2017 IEEE 4th International Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, 2017, pp. 0054-0062.

[6] P. K. Rachabathuni, “A survey on abstractive summarization techniques,” 2017 International Conference on Inventive Computing and Informatics (ICICI), Coimbatore, 2017, pp. 762-765.

[7] V. U. Thompson, C. Panchev, and M. Oakes, “Performance evaluation of similarity measures on similar and dissimilar text retrieval,” 2015 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K), Lisbon, 2015, pp. 577-584.

[8] A. R. Lahitani, A. E. Permanasari, and N. A. Setiawan, “Cosine similarity to determine similarity measure: Study case in online essay assessment,” 2016 4th International Conference on Cyber and IT Service Management, Bandung, 2016, pp. 1-6.

[9] Y. Oganian, M. Conrad, A. Aryani, K. Spalek, and H. R. Heekeren, “Activation Patterns throughout the Word Processing Network of L1-dominant Bilinguals Reflect Language Similarity and Language Decisions,” in Journal of Cognitive Neuroscience, vol. 27, no. 11, pp. 2197-2214, Nov. 2015.

[10] L. Gao and H. Chen, “An automatic extraction method based on synonym dictionary for web reptile question and answer,” 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA), Wuhan, 2018, pp. 375-378.

[11] T. Bluche, C. Kermorvant, C. Touzet, and H. Glotin, “Cortical-Inspired Open-Bigram Representation for Handwritten Word Recognition,” 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, 2017, pp. 73-78.

[12] A. Magooda , M. A. Zahran , M. Rashwan , H. Raafat, and M. B. Fayek, “Vector Based Techniques for Short Answer Grading, “ Proceedings of the Twenty-Ninth International Florida Artificial Intelligence Research Society Conference,2014,pp.238-243

[13] M. A. Sultan C. Salazar, and T, Sumner, “Fast and Easy Short Answer Grading with High Accuracy,” Proceedings of NAACL-HLT 2016, pp.1070–1075

[14] M. Mohler and R. Mihalcea, “Text-to-text Semantic Similarity for Automatic Short Answer Grading,” International Journal of Artificial Intelligence in Education 25 (2015), 118 – 125

[15] J. Nau, A. H. Filho, and G. Passero, “Evaluating Semantic Analysis Methods For Short Answer Grading Using Linear Regression, “PEOPLE:

International Journal of Social Sciences (2017), Volume 3 Issue 2, pp. 437 – 450.

[16] P. Selvi and A. K.Bnerjee, "Automatic Short – Answer Grading System (ASAGS)," InterJRI Computer Science and Networking (2010), Vol. 2, Issue 1, pp.18-23.

[17] S. K. Chowdhury and R. J. R. Sree, "Dimensionality reduction in automated evaluation of descriptive answers through zero variance, near zero variance and non-frequent words techniques - a comparison," 2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO), Coimbatore, 2015, pp. 1-6.

[18] M. A. G. Mohler, R. Bunescu, and R. Mihalcea, "Learning to Grade Short Answer Questions using Semantic Similarity Measures and Dependency Graph Alignments, "International Journal of Artificial Intelligence in Education 27 (2016), 83-89.

[19] P. Nikam, M. Shinde, R. Mahajan, and S. Kadam, "Automatic Evaluation of Descriptive Answer Using Pattern Matching Algorithm," International Journal of Computer Sciences and Engineering (2015) Vol.-3(1), pp.69-70.

[20] M. S. M. Patil and M. S. Patil, "Evaluating Student Descriptive Answers Using Natural Language Processing," International Journal of Engineering Research & Technology (IJERT) 2014, Vol. 3 Issue 3.

[21] M. Loor and G. De Tré, "Choosing suitable similarity measures to compare intuitionistic fuzzy sets that represent experience-based evaluation sets," 2015 7th International Joint Conference on Computational Intelligence (IJCCI), Lisbon, 2015, pp. 57-68.