Grenze International Journal of Engineering and Technology, June Issue

Subjective Answer Evaluation for Technical Subjects with NLP-Driven Advancements

Kavita R. Singh[1], Akshita Kamdi[2], Chinmayee Kakde[3], Dharti Bhuva[4], Nikita Rathod[5], Pooja Gotmare[6], Ogundele Oluwadamilare Sheriff[7] and Arvinder Kour[8]
[1-6]Department of Computer Technology, Yeshwantrao Chavan College of Engineering, Nagpur, Maharashtra, India
[7]Department of Surveying and Geoinformatics, Bells University of Technology, Nigeria
[8]Department of Humanities, Yeshwantrao Chavan College of Engineering, Nagpur, Maharashtra, India
{singhkavita19@gmail.com, Kamdi-akshitakamdi23@gmail.com, rutukakde123@gmail.com,dhartibhuva59@gmail.com, rathodnikita0603@gmail.com, poojagotmare2002@gmail.com, ogunsd2001@yahoo.com, aru.akm@gmail.com}

***Abstract*—This research paper addresses the challenge of manually grading subjective papers. It examines existing methods and suggests a new approach using computers and advanced tools to enhance the speed and accuracy of grading. Through experimentation, the study assesses the performance of this novel method. The results contribute to automated grading systems for subjective papers. Additionally, this paper proposes a novel approach utilizing natural language processing (NLP) techniques, including WordNet, cosine similarity, and TF-IDF, to automatically assess descriptive answers.**

***Index Terms*— Subjective paper evaluation, natural language processing, automated evaluation, cosine similarity.**

I. INTRODUCTION

The evaluation of students' performance and capabilities through subjective questions and answers is a crucial aspect of educational assessment. Unlike objective questions that limit responses to concise answers, subjective questions afford students the opportunity to articulate their thoughts and comprehension more expansively. These responses, characterized by greater depth and context, set them apart from their objective counterparts.

Nevertheless, the assessment of subjective answers presents formidable challenges for educators. Grading necessitates a meticulous examination of each word, considering factors such as the teacher's mental state, fatigue, and objectivity. As a result, subjective exams become intricate and daunting for both students and teachers alike. Given these complexities, it is imperative to devise an efficient system for evaluating subjective answers. While machines can readily evaluate objective answers by matching them with succinct, one-word responses provided by students, assessing subjective answers proves more challenging. These responses display considerable variability in length and incorporate a diverse range of vocabulary. Additionally, individuals often employ synonyms and convenient abbreviations, further complicating the evaluation process.

Addressing these challenges by developing effective systems and techniques for evaluating subjective answers is paramount in the field of education and assessment. Such advancements hold the potential to significantly improve the accuracy, fairness, and efficiency of assessing students' comprehension and performance in open-ended assessments. This research paper contributes to the field of educational assessment by addressing the challenge of manually grading subjective papers. It proposes a novel approach that utilizes computers and

Grenze ID: 01.GIJET.10.2.581

advanced tools, such as natural language processing (NLP) techniques, to enhance the speed and accuracy of grading. The organization of the paper is as follow. Section 2 presents literature review and section 3 presents the research gap and contribution followed by methodology description in Section 4. The experimentation and results are presented in Section 5 finally followed by conclusion in Section 6.

II. Literature Review

Bahel and A. Thomas [1] introduced a framework to assess subjective questions by utilizing text summarization, text semantics, and keyword summarization. They evaluated its performance against other methods and found that while their approach yielded an error of 1.372, Jaccard's similarity approach had a slightly lower error of 1.312. However, their method encountered difficulty in handling non-textual data such as diagrams and images. C. Xia, T. He, W. Li, Z. Qin, and Z. Zou [2] used a method called word2vec with legal documents to find similarities between different law texts. They measured the similarity between sentences using something called cosine similarity. This improved accuracy by 0.2 compared to another method called Bag of Words. And if they trained word2vec specifically on law documents, they could increase accuracy even more, by about 0.05 to 0.10. J.-E. Kim, K. Park, J.-M. Chae, H.-J. Jang, B.-W. Kim, and S.-Y. Jung Kim and colleagues [3] introduced a technique for grading brief descriptive responses using a lexico-semantic pattern (LSP), which worked well with the complex structure of the Korean language. LSP helps organize the meaning of the response, aiding in understanding the user's intentions. Additionally, a list of synonyms was employed to broaden the keywords, ensuring compatibility with various response styles. The dataset, collected from 88 students, was converted into LSP and compared with solution LSP to evaluate the response. Consequently, the system outperformed existing systems by 0.137. M. Oghbaie and M. M. Zanjireh [4] introduced a pairwise similarity measure to assess the similarity between two documents based on the keywords present in either document. Their work introduced a novel similarity measure termed PDSM (pair-wise document similarity measure), which was a modified version of the preferable properties approach. This measure was applied to text mining tasks such as document detection, k Nearest Neighbors (kNN) for single-label classification, and K-means clustering. They evaluated the accuracy using a measure of evaluation, and the results showed that the PDSM method outperformed other measures like the Jaccard coefficient by improving recall by 0.08. M. Alian and A. Awajan [5] investigated the influences of different factors on sentence similarity and identifying paraphrases. They utilized various word embedding models, clustering algorithms, and weighting methods to understand the context of sentences. The pre-trained embeddings they employed were AraVec and FastTex, both tailored for the Arabic language, and trained on a dataset consisting of approximately 77,600,000 tweets. Their findings showed that utilizing pre-trained embeddings along with labeled data from experts led to improved recall and precision, achieving 0.87 and 0.782 respectively for K-means and agglomerative clustering. K. Orkphol and W. Yang [6], used a method called word2vec to represent words in a fixed-sized vector space. They then compared sentence similarities using cosine similarity. They used Google's word2vec and averaged the word vectors in a sentence to get the sentence vector. To accept a score, it had to pass a certain similarity threshold between 0 and 1. They tested the system's performance using recall and accuracy measures. With the probability of sense distribution, the system performed at 50.9%, and without it, at 48.7%. G. Jain and D. K. Lobiyal [7] G. Jain and D. K. Lobiyal introduced a fresh method for assessing subjective questions by employing concept graphs. They generated these graphs for both the solution and the answer, then assessed the score using different graph similarity techniques. In a related study, M.

From the detailed literature it has been observed that earlier systems could assess only five students at once. However, the proposed system presents two-fold contribution; firstly, there is a notable advancement in the proposed system which can evaluate 15 students simultaneously. This enhancement reflects a significant boost in efficiency and scalability compared to current methods. Secondly, in proposed system, we tackle the issue of different words having the same meaning. We create a synonym dictionary using the teacher's answer. Then, we replace words in students' answers with their synonyms from this dictionary. This helps our system understand and grade answers better, even if they use different words with the same meaning.

III. Methodology

The proposed system is designed with a focus solely on software engineering. It comprises four key components: data collection, preprocessing, similarity measurement, and marking scheme. Initially, sample answers specific to software engineering are gathered from teachers, addressing a definitive set of 15 questions. Subsequently, students provide their responses, which are then analyzed for similarity against the predefined criteria.

A. Data Collection

System operates with two CSV files: one for students and one for teachers. System extract information from CSV files and create data structures named teacher answer and student answers, which contain the teacher's and students' responses to questions, respectively.

Teacher Solution The teacher/evaluator typically prepares the solution to the question. The solution is a subjective answer that is being used to map students' responses. This solution must contain all the keywords discussed in the answers.

Student Response

The response is what the student writes to answer the question. Answer can vary in length depending on question type and how the student writes. It usually It includes some or all the keywords provided.

B. Preprocessing Module

Before comparing the student's answer with the teacher's solution, we conduct several text preparations steps. Initially, we eliminate common words, known as stop words, from both the teacher's solution and the student's answer to focus on essential content. Next, we leverage a synonym dictionary to identify and replace words in the student's solution with teacher's answer's synonyms, enriching the text without altering its meaning. Once these steps are completed, we proceed to compare both texts to assess their similarity, aiding in evaluating the alignment between the student's response and the teacher's expectations.

Keywords

Keywords are important words unique to each question. They're crucial for providing accurate answers. When system measures similarity between answers, it pays close attention to these keywords. They greatly affect how scores are determined.

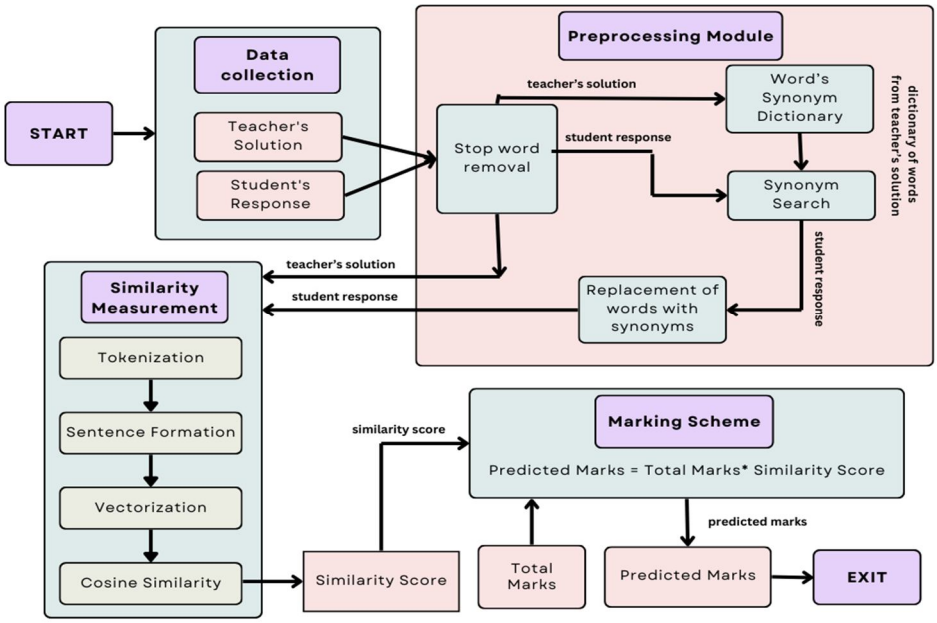

Figure 1. Subjective answer evaluation system

Stop Word Removal

Words, such as "the", "is", "and", "but", *etc.*, are frequently used but do not carry significant meaning in the context of text analysis. Therefore, in the preprocessing stage, stop words are removed from both the teacher's solution and the student's answer. By eliminating these words, we can focus on the essential content and improve the accuracy of subsequent analyses, such as similarity comparison between texts. This step helps streamline the text and enhance the effectiveness of the overall evaluation process.

Synonyms Search

In the preprocessing phase, a synonym dictionary is employed to enhance the student's answers. To create a synonym dictionary, we first extract keywords from the teacher's answers. These keywords serve as the

foundation for constructing the dictionary, which contains synonyms of these keywords. Then, when processing the student's answer, we utilize this dictionary to replace words corresponding to the keywords. For example, if the student's answer includes the word "happy" and the teacher's answer contains "joyful" then the synonym dictionary will replace "happy" in the student's response with "joyful". By aligning the student's response with synonymous terms extracted from the teacher's answers this method helps us evaluate the student's answer thoroughly and fairly.

C. Similarity Measurement

In similarity measurement phase we start by tokenizing the text, which means breaking it down into smaller parts like sentences and words. Then, we organize these words into sentences to maintain the text's structure. After that, we convert the words into numbers, a process called vectorization, so we can analyze them more easily. Finally, we calculate the cosine similarity between the numerical representations of the texts. This measures how closely related the texts are in terms of their meaning. By following these steps, we can effectively assess the similarity between different pieces of text.

Tokenization

Tokenization is a crucial step in text processing where data is divided into smaller units, such as paragraphs, sentences, words, or characters. This segmentation is essential for understanding the meaning of individual components in a text. Tokenization serves as one of the initial stages in text processing, providing the foundation for further analysis and understanding of textual content.

Sentence Formation

In data preparation for vectorization, tokenized words from each paragraph are reassembled into cohesive sentences. This preserves the original structure and meaning of the text, easing the subsequent vectorization process.

Vectorization

TF-IDF (Term Frequency-Inverse Document Frequency) is a method for converting text data into numerical vectors. It assesses the importance of terms in documents by considering both how frequently they appear in a document (TF) and how unique they are across the entire corpus (IDF). TF measures the local importance of terms within a document, while IDF measures their global importance across all documents. By combining TF and IDF, TF-IDF provides a numerical representation of documents, with each term's importance weighted relative to the entire corpus.

TF (term, document) = (Number of times term appears in document) / (Total number of terms in document)

IDF (term) = loge (Total amount of documents / Number of documents containing term)

Combining TF and IDF, TF-IDF produces numerical representations of documents, where each term's importance is weighted relative to the entire corpus. This approach enables effective analysis and comparison of text documents in various natural language processing applications.

Cosine Similarity Computation

Cosine similarity helps compare a student's answer with a model answer. First, both answers are turned into number representations using TF-IDF. Then, cosine similarity measures how similar these representations are. Higher scores mean closer matches, indicating better understanding by the student. Lower scores may suggest differences or incomplete understanding. This method helps teachers fairly evaluate student responses. The formula for cosine similarity measures the cosine of the angle between two vectors in a multi-dimensional space. The formula is given by: $(\mathbf{A}, \mathbf{B}) = \mathbf{A} \cdot \mathbf{B} / \|\mathbf{A}\| \|\mathbf{B}\|$

where $\mathbf{A} \cdot \mathbf{B}$ denotes the dot product of vectors **A** and **B**, and $\|\mathbf{A}\|$ and $\|\mathbf{B}\|$ denote the Euclidean norms of vectors **A** and **B** respectively.

Marking Scheme

In this module, marks are determined through a process utilizing similarity scores, incorporating methods like tokenization, sentence formation, and cosine similarity. The similarity score is multiplied by the total marks allocated to the question for calculating the expected marks.

*Predicted Marks=Similarity Score*Total Marks*

IV. EXPERIMENTATION AND RESULT

Utilizing potent tools like cosine similarity and aims to enhance the accuracy and efficiency of automated subjective paper evaluation. The experimental design employed in the study provides a rigorous evaluation of

the proposed approach, yielding significant insights. The results demonstrate a substantial improvement in automated evaluation, surpassing existing methods. Cosine similarity refines the identification of subtle language nuances, contributing to a more nuanced assessment. This research makes a noteworthy contribution to the field by offering a promising solution to the challenges of subjective paper evaluation, showcasing advancements in accuracy and efficiency within the realm of automated assessment processes. For understating of the reader, we provide a sample question with reference answer.

Faculty reference question and answer
Que 1: Explain the prototyping model in software development, detailing the requirement for building a working prototype before the development of the actual software. Describe the characteristics of a typical prototype and discuss its role in the development process.
Reference Answer: The Prototyping Model in software development involves creating a preliminary version of the software, known as a prototype, before proceeding with full development. This prototype aims to demonstrate key features and functionalities, allowing stakeholders to provide feedback and validate requirements. Typical characteristics of a prototype include basic functionality, a simplified design, and rapid development. While not fully functional, the prototype serves as a tangible representation of the final product. The role of a prototype in the development process is crucial. It helps in identifying potential issues early, gathering user feedback, and refining requirements. By allowing stakeholders to interact with a tangible representation of the software, prototypes facilitate better communication and alignment of expectations. Overall, prototypes serve as a valuable tool for mitigating risks, enhancing collaboration, and ensuring that the final product meets user needs effectively.

Ques 2: What is Component-Based Development in software engineering, and how does it facilitate software reuse?
Reference Answer: Component-Based Development (CBD) employs software components, offered by vendors as products, for building software systems. These components offer specific functionalities with clear interfaces, facilitating their integration into software projects. They can be designed as traditional software modules or object-oriented classes. CBD fosters software reuse, leading to numerous measurable benefits for software engineers. Reusability reduces development time and costs by leveraging existing solutions. It enhances maintainability by allowing updates or replacements of individual components without affecting the entire system. CBD also improves scalability, as components can be easily replicated across projects. Additionally, it enhances reliability and consistency by promoting the use of well-tested and established components. Overall, CBD streamlines development processes and enhances the quality of software systems through efficient reuse of proven components.

Que 3: Outline the principles of Agility in software development as described in the provided text. Highlight key concepts such as customer satisfaction, adaptability to change, frequent delivery of working software, collaboration between businesspeople and developers, emphasis on motivated individuals, and continuous improvement through reflection and adjustment.
Reference Answer: Agility in software development prioritizes customer satisfaction by delivering valuable software early and continuously, accommodating changing requirements for competitive advantage. It advocates for frequent delivery of working software, daily collaboration between businesspeople and developers, and building projects around motivated individuals. Face-to-face conversation is favoured for efficient communication. Progress is measured by working software, and sustainable development is promoted to maintain a constant pace. Attention to technical excellence and simplicity enhances agility, while self-organizing teams foster the emergence of optimal solutions. Regular reflection and adjustment ensure continuous improvement in team effectiveness and behaviour.

Que 4. How does a process framework contribute to effective software development?
Reference Answer: A process framework plays a pivotal role in effective software development by providing a structured approach to managing the complexities of the development lifecycle. It establishes a set of framework activities applicable to all software projects, ensuring consistency and standardization regardless of project size or complexity. Additionally, the framework encompasses umbrella activities such as project tracking and control, risk management, and software quality assurance, which are essential for project success. By delineating these activities, the framework facilitates better project planning, resource allocation, and risk mitigation. Moreover, it promotes collaboration and communication among team members, enhancing efficiency and reducing the likelihood of misunderstandings or errors. Ultimately, a well-defined process framework enables

teams to streamline development processes, deliver high-quality software products, and meet stakeholder expectations effectively.

Que 5. What are Negotiating requirements in requirement engineering?
Reference Answer: In requirement engineering, negotiating requirements involves identifying key stakeholders, understanding their "win conditions," and facilitating discussions to reach mutually beneficial agreements. Key stakeholders encompass individuals or groups directly impacted by the software project, whose involvement is crucial for negotiation. Understanding each stakeholder's goals, priorities, and constraints enables negotiators to tailor discussions and proposals, accordingly, fostering collaboration and consensus-building. Negotiation aims to reconcile divergent interests and find common ground among stakeholders, ultimately working towards a set of requirements that lead to a "win-win" outcome. By fostering open communication and constructive dialogue, negotiators navigate complexities inherent in requirement engineering, ensuring that the final set of requirements aligns with stakeholder expectations and project objectives.

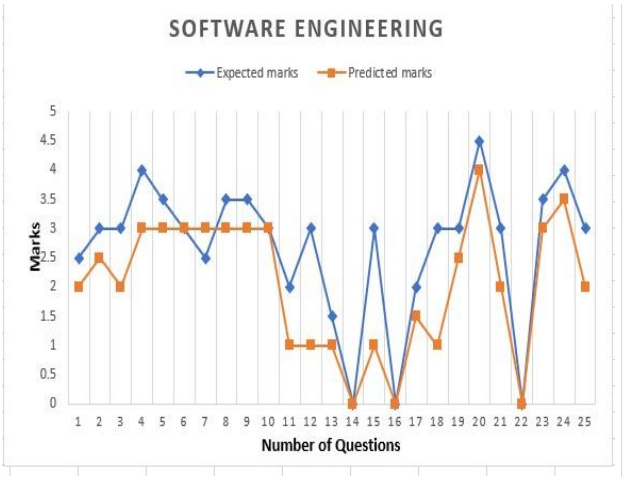

Figure 2. Software Engineering Marks Line Graph

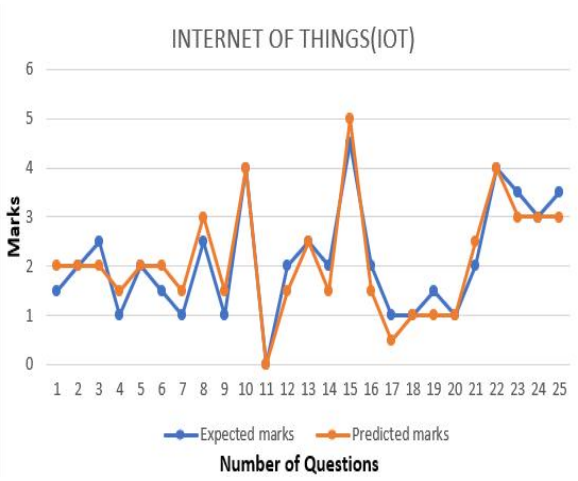

Figure 3. IoT Marks Line Graph

The system was tested on the Software Engineering (SE), Data structures (DS), Internet-of -Things (IoT) subject to evaluate its performance across different subjects. The results revealed that the system's accuracy is slightly higher in the IoT domain as shown in Table 2 and Figure 3 compared to software engineering results demonstrated in Table 1 and figure 2.. This can be attributed to the simplicity of most keywords in the IoT subject, which are already present in the dictionary. Consequently, the system produces more precise results in IoT-related contexts, showcasing its effectiveness in handling straightforward terminology. This enhanced accuracy underscores the system's adaptability and reliability across various subject domains. While evaluating system applicability, the system was tested with Data Structures (DS) but encountered an error. the system cannot evaluate pseudo code, indicating a potential area for future enhancement.

V. Conclusions and Future Scope

The proposed educational assessment system represents a transformative breakthrough, poised to revolutionize established procedures across diverse academic institutions. Cosine similarity quantifies vector similarity, enhancing the evaluation of subjective papers. These techniques elevate data quality, aiding in various text-based tasks. The system architecture encompasses tokenization, stop word removal, keyword extraction, synonym search, and stemming for text pre-processing. It follows specific rules for evaluating scores based on similarity checking (SC) and question marks. The system's inability to accommodate special characters and digits. The system exclusively accepts answers in paragraph format, precluding the use of bullet points or listed responses. If the synonyms provided by the teacher's dictionary do not align with the words used by the students, there may be discrepancies in the marks assigned to the students.
Despite the proliferation of online evaluations, the persistent challenges in effectively assessing subjective responses have underscored the limitations of multiple-choice questions in capturing the nuanced academic abilities of students.

TABLE I. EXPECTED AND PREDICTED MARKS OF STUDENTS

Question Number	Student Number	Expected Marks (Out of 5)	Similarity Score	Predicted Marks (Out of 5)	Error
1.	1.	2.5	0.33	2	0.5
	2.	3	0.51	2.5	0.5
	3.	3	0.44	2	1
	4.	4	0.56	3	1
	5.	3.5	0.59	3	0.5
2.	1.	3	0.60	3	0
	2.	2.5	0.52	3	0.5
	3.	3.5	0.64	3	0.5
	4.	3.5	0.67	3	0.5
	5.	3	0.69	3	0
3.	1.	2	0.21	1	1
	2.	3	0.18	1	2
	3.	1.5	0.2	1	0.5
	4.	0	0.0	0	0
	5.	3	0.2	1	2
4.	1.	0	0.0	0	0
	2.	2	0.29	1.5	0.5
	3.	3	0.13	1	2
	4.	3	0.49	2.5	0.5
	5.	4.5	0.78	4	0.5
5.	1.	3	0.41	2	1
	2.	0	0.0	0	0
	3.	3.5	0.58	3	0.5
	4.	4	0.7	3.5	0.5
	5.	3	0.43	2	1

TABLE II. EXPECTED AND PREDICTED MARKS OF STUDENTS IN THE SUBJECT OF IOT

Question Number	Student Number	Expected Marks	Similarity Score	Predicted Marks	Error
1.	1.	1.5	0.39	2	0.5
	2.	2	0.4	2	0
	3.	2.5	0.39	2	0.5
	4.	1	0.25	1.5	0.5
	5.	2	0.4	2	0
2.	1.	1.5	0.45	2	0.5
	2.	1	0.28	1.5	0.5
	3.	2.5	0.54	3	0.5
	4.	1	0.3	1.5	0.5
	5.	4	0.83	4	0
3.	1.	0	0	0	0
	2.	2	0.33	1.5	0.5
	3.	2.5	0.49	2.5	0
	4.	2	0.3	1.5	0.5
	5.	4.5	0.97	5	0.5
4.	1.	2	0.35	1.5	0.5
	2.	1	0.15	0.5	0.5
	3.	1	0.2	1	0
	4.	1.5	0.2	1	0.5
	5.	1	0.19	1	0
5.	1.	2	0.42	2.5	0.5
	2.	4	0.81	4	0
	3.	3.5	0.66	3	0.5
	4.	3	0.57	3	0
	5.	3.5	0.65	3	0.5

REFERENCES

[1] V. Bahel and A. Thomas, "Text similarity analysis for evaluation of descriptive answers," CoRR, vol. abs/2105.02935, 2021.

[2] C Xia, T. He, W. Li, Z. Qin, and Z. Zou, "Similarity analysis of law documents based on word2vec," in 2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C), pp. 354– 357, IEEE, 2019.

[3] J.-E. Kim, K. Park, J.-M. Chae, H.-J. Jang, B.-W. Kim, and S.-Y. Jung, "Automatic scoring system for short descriptive answer written in korean using lexico-semantic pattern," Soft Computing, vol. 22, no. 13, pp. 4241– 4249, 2018.

[4] M. Oghbaie and M. M. Zanjireh, "Pairwise document similarity measure based on present term set," Journal of Big Data, vol. 5, no. 1, pp. 1–23, 2018.

[5] M. Alian and A. Awajan, "Factors affecting sentence similarity and paraphrasing identification," International Journal of Speech Technology, vol. 23, no. 4, pp. 851–859, 2020.

[6] K. Orkphol and W. Yang, "Word sense disambiguation using cosine similarity collaborates with word2vec and wordnet," Future Internet, vol. 11, no. 5, p. 114, 2019.

[7] G. Jain and D. K. Lobiyal, "Conceptual graphs-based approach for subjective answers evaluation," Int. J. Concept. Struct. Smart Appl., vol. 5, no. 2, pp. 1–21, 2017.