Journal of Soft Computing Exploration
Homepage: htttps://shmpublisher.com/index.php/joscex
p-ISSN: 2746-7686
e-ISSN: 2746-0991

.

A comparative analysis of five textual similarity methods for automatic short answer grading

Imam Rangga Bakti[1], Handaru Jati[2], Nurkhamid[3], Yola Permata Bunda[4]
[1]Department of Informatics Engineering, Universitas Pasir Pengaraian, Indonesia
[2]Department of Electronics and Informatics Engineering Education, Universitas Negeri Yogyakarta, Indonesian
[3]Department of Informatics Engineering Education, Universitas Negeri Yogyakarta, Indonesian
[4]Department of Information System, Universitas Tjut Nyak Dhien, Indonesia

Article Info

Article history:

Received Mar 11, 2026
Revised Mar 19, 2026
Accepted Mar 22, 2026

Keywords:

Text mining
ASAG
Comparison
Five methods
Textual similarity

ABSTRACT

This study investigates the application of text mining techniques in Automatic Short Answer Grading (ASAG) by comparing five textual similarity methods: Cosine Similarity, Jaccard Similarity, Dice's Coefficient, Overlap Coefficient, and Matching Coefficient. The dataset consists of five definition-based questions answered by 25 students in a Human–Computer Interaction course. The data were preprocessed using case folding, tokenization, stop word removal, and stemming. The results show that Cosine Similarity achieved the highest similarity score of 67.00%, followed by Overlap Coefficient (66.67%) and Dice's Coefficient (63.16%), while Jaccard Similarity and Matching Coefficient produced lower scores of 46.15%. These findings indicate that vector-based similarity methods are more effective in handling variations in sentence structure and keyword usage compared to set-based approaches, particularly for definition-based short answers. This study provides a comparative evaluation of multiple lexical similarity methods within a unified experimental setting, offering practical insights for selecting appropriate techniques in ASAG applications.

Corresponding Author:

Imam Rangga Bakti,
Department of Informatics Engineering, Rokan Hulu, Indonesia,
University Pasir Pengaraian,
Email: imamranggabakti@gmail.com
https://doi.org/10.52465/joscex.v7i1.11

1. INTRODUCTION

Text mining is a multidisciplinary process that transforms unstructured textual data into a structured format, enabling organizations to extract valuable insights from various sources such as emails, social media, and customer feedback [1],[2]. This process involves a wide range of techniques and has been applied in diverse fields, from music therapy to scientific research, highlighting its versatility and importance in data analysis [3],[4]. One of the domains that increasingly benefits from text mining techniques is educational assessment, particularly in the automatic evaluation of student responses.

Automatic Short Answer Grading (ASAG) is a rapidly growing field that employs machine learning and natural language processing (NLP) to evaluate student responses efficiently [5],[6], [7]. The integration of advanced approaches, such as deep learning and neuro-symbolic methods, has significantly improved the

performance of ASAG systems, making them valuable tools for educational institutions [9], [10], [11]. These systems are designed not only to provide accurate assessments but also to deliver constructive feedback that can enhance learning outcomes [12], [13], [14].

ASAG technology offers a promising solution for improving assessment efficiency in educational settings, particularly in large classrooms and online learning environments [15], [16], [17], [18]. However, one major challenge is accurately measuring the degree of alignment between student responses and answer keys, given the considerable variation in language use, writing style, and response length [19]. To address this challenge, textual similarity analysis has emerged as one of the most widely adopted approaches in ASAG systems.

Several studies have explored the use of textual similarity methods in ASAG. Wahyuningsih et al. [20] compared three methods, Cosine Similarity, Jaccard Similarity, and Dice's Coefficient, and found that Cosine Similarity and Dice's Coefficient produced the best performance in grading short answers. Lubis et al. [21] investigated semantic similarity based on word embedding and reported improved grading accuracy compared to traditional lexical methods. Putnikovic and Jovanovic [22]conducted a scoping review of embedding-based approaches for ASAG and highlighted that lexical similarity methods remain relevant as baseline comparisons due to their simplicity and interpretability. Ayaan and Ng [23]further demonstrated that NLP-based automated grading systems achieve competitive performance when combined with semantic analysis techniques. Meanwhile, Goenka et al. [24] proposed an explainable short answer scoring system that emphasized the importance of method selection based on answer characteristics. While machine learning and deep learning approaches have shown strong performance in ASAG, they often require large annotated datasets and higher computational resources. In contrast, lexical similarity methods remain widely used due to their simplicity, interpretability, and efficiency, particularly in small-scale or resource-constrained settings.

Despite these advances, most existing studies either focus on a limited subset of similarity methods or rely on English-language datasets, leaving a gap in systematic comparative evaluations across a broader range of lexical similarity methods within the same experimental framework. In particular, no study has systematically compared Cosine Similarity, Jaccard Similarity, Dice's Coefficient, Overlap Coefficient, and Matching Coefficient within the same experimental setting for ASAG. Furthermore, the relative performance of these methods on definition-based short answers, which are common in university-level examinations, has not been thoroughly examined.

These five methods were selected because they represent commonly used lexical similarity approaches with different underlying principles, including vector-based, set-based, and overlap-based techniques, making them suitable for a comprehensive comparative evaluation.

This study therefore aims to fill this gap by comparing these five textual similarity methods within the context of ASAG to determine the most effective and adaptive approach for grading definition-based short answers. The findings are expected to strengthen the scientific basis for selecting appropriate similarity methods in ASAG systems and to provide practical guidance for developing more effective and accurate ASAG applications in educational contexts.

2. METHOD

In this study, the researchers used the Cosine Similarity, Jaccard Similarity, Dice’s Coefficient, Overlap Coefficient, and Matching Coefficient methods. Python was used to calculate the similarity between the answer texts, while Microsoft Excel was used to calculate the correlation and MAE values. This research consisted of several stages or steps, which are presented in Figure 1 as the research flowchart.

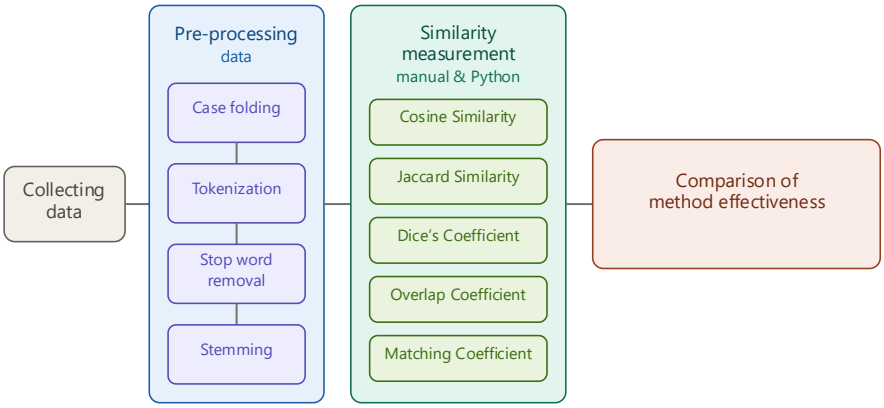

Figure 1. Research method and procedure

Collecting Data

Data collection was the first stage of this study. The dataset was obtained from the midterm examination of the Human Computer Interaction course at Universitas Pasir Pengaraian. The data were limited to definition-based questions, as this type of question produces short, structured responses that are suitable for lexical similarity analysis. The dataset consists of five questions, each answered by 25 students, resulting in a total of 125 answer pairs for evaluation. Each pair consists of a student response and a corresponding answer keys provided by the course lecturer. The answer keys were formulated as concise sentences representing the ideal definitions for each question. The questions, answer keys, and samples of student answers used in this study are presented in Table 1.

Table 1. Questions, answer keys and student answers

Questions	Answer key	Student Answer
Apa itu Interaksi Manusia dan Komputer (IMK)?	IMK adalah studi tentang cara manusia berinteraksi dengan komputer melalui antarmuka pengguna.	IMK adalah studi tentang cara manusia berinteraksi dengan komputer melalui antarmuka pengguna
Apa tujuan utama dari desain antarmuka pengguna (UI)?	Tujuannya adalah membuat interaksi lebih mudah, efisien, dan menyenangkan.	Tujuannya yaitu melakukan interaksi menjadi lebih mudah, efisien serta menyenangkan
Apa yang dimaksud dengan "usability" dalam konteks IMK?	Usability adalah seberapa mudah antarmuka digunakan oleh pengguna.	Usability merujuk pada seberapa sederhana antarmuka untuk digunakan oleh pengguna.
Apa perbedaan antara antarmuka berbasis teks dan antarmuka grafis?	Antarmuka teks menggunakan perintah teks, sementara grafis menggunakan ikon dan tombol visual.	Antarmuka berbasis teks memanfaatkan perintah teks, sementara antarmuka grafis menggunakan elemen visual seperti ikon dan tombol.
Mengapa penting untuk memperhatikan aksesibilitas dalam desain antarmuka pengguna?	Agar semua pengguna, termasuk yang memiliki keterbatasan, bisa mengakses aplikasi dengan mudah.	Aksesibilitas penting untuk memastikan semua pengguna, termasuk yang memiliki keterbatasan, dapat dengan mudah mengakses aplikasi.

Preprocessing Data

Preprocessing is applied to raw data, which usually contains elements that are not meaningful for text mining, such as stop words. In order for the data to be processed effectively, it must first go through the preprocessing stage. Data preprocessing is the process of preparing raw data before it undergoes further analysis. The preprocessing steps carried out in this study are as follows [25], [26], [27]:

Case Folding

Case folding is used to convert all text into lowercase letters. This step helps facilitate the search process, since text documents are not always consistent in their use of uppercase and lowercase letters.

Tokenization

Tokenization is the process of splitting text from a paragraph or sentence into smaller units called tokens. For example, the phrase “a process of buying and selling goods” can be divided into the following tokens: a, process, buying, selling, and goods.

Stop Word Removal

Stop word removal (filtering) is a step that removes words that frequently appear but do not carry significant meaning, such as conjunctions, prefixes, and other common function words.

Stemming

Stemming is the process of transforming words into their root or base form.

Cosine Similarity, Jaccard Similarity, Dice Coefficient, Overlap Coefficient, and Matching Coefficient

This study employs five methods, namely Cosine Similarity, Jaccard Similarity, Dice Coefficient, Overlap Coefficient, and Matching Coefficient, to measure text or sentence similarity. In general, these methods determine the level of similarity based on the common words found in the sentences being compared. The similarity calculations in this study follow commonly used textual similarity measures in text mining and information retrieval research [28], [29], [30] . The formulas for Cosine Similarity, Jaccard Similarity, Dice's Coefficient, Matching Coefficient, and Overlap Coefficient are presented in Table 2.

Table 2. Algorithm formula

Metode Similarity	Formula
Cosine Similarity	$\text{Cosine Similarity} = \frac{\text{A. B}}{\|\|\text{A}\|\|\|\|\text{B}\|\|}$
Jaccard similarity	$\text{Jaccard Similarity (A, B)} = \frac{\text{A} \cap \text{B}}{\text{A} \cup \text{B}}$
Dice Coefficient	$\text{Dice's Coeficient} = \frac{2 \cdot \|\text{A} \cap \text{B}\|}{\cdot \|\text{A}\| + \|\text{B}\|}$
Matching Coefficient	$\text{Matching Coeficient(A, B)} = \frac{\text{A} \cap \text{B}}{\text{A} \cup \text{B}}$
Overlap Coefficient	$\text{Overlap Coeficient} = \frac{\text{A} \cap \text{B}}{\min(\|\text{A}\|, \|\text{B}\|)}$

3. RESULTS AND DISCUSSIONS

Text mining, also known as text data mining or text analysis, is the process of extracting meaningful information from unstructured textual data. It has become an essential tool for handling the vast amount of text generated in today's digital world. Text mining techniques are designed to transform unstructured text into structured data, enabling the discovery of patterns and trends that support decision-making. These techniques are supported by advances in natural language processing (NLP), artificial intelligence (AI), and machine learning algorithms.

Some basic techniques in text mining include text preprocessing, such as tokenization and case folding. Commonly used words that do not contribute significantly to the meaning of the text are removed in order to reduce noise in the data. Other important preprocessing techniques are stemming and lemmatization. These processes reduce words to their base or root form, which helps normalize textual data. Stemming truncates words to their basic form, while lemmatization considers the context in order to derive the correct root form.

Furthermore, after the preprocessing stage, similarity measurement was carried out manually using the Cosine Similarity, Jaccard Similarity, Overlap Coefficient, and Matching Coefficient methods. This study also presents the calculation results of each method using a simple program written in the Python programming language, so that the results of the manual calculations and the program-based calculations can be compared clearly. The following section presents an example of the calculation of the similarity between student answers and the lecturer's answer key using the five methods. The examples are taken from the answer key and student responses shown in Table 3.

Cosine Similarity

The first step in this method is tokenization and the construction of word vectors (Bag of Words). At this stage, both the answer key and the student's answer are broken down into individual words, and the frequency of occurrence of each word is recorded. The results of this process can be seen in the following table:
Answer Key: Tujuannya adalah membuat interaksi lebih mudah, efisien, dan menyenangkan.
Student Answer: Tujuannya yaitu melakukan interaksi menjadi lebih mudah, efisien serta menyenangkan

Table 3. Tokenization

Word	Answer Key	Student Answer
Tujuannya	1	1
adalah	1	0
membuat	1	0
interaksi	1	1
lebih	1	1
mudah	1	1
efisien	1	1
dan	1	0
menyenangkan	1	1
yaitu	0	1
melakukan	0	1
menjadi	0	1
serta	0	1

From the table above, the Vector for Answer Key and Answer can be made:
Answer Key Vector 1 (A) = (1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0)
Answer Vector 1 (B) = (1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1)

Next, the second step is to calculate cosine similarity where the cosine similarity formula is:

$$Cosine\ Similarity = \frac{A.\ B}{||A||||B||} \quad (1)$$

Dot Product (A - B) is the sum of the multiplication of the corresponding elements between two vectors.

$$A \cdot B = (1*1) + (1*0) + (1*0) + (1*1) + (1*1) + (1*1) + (1*1) + (1*0) + (1*1) + (0*1) + (0*1) + (0*1) + (0*1)$$

$$A \cdot B = 1 + 0 + 0 + 1 + 1 + 1 + 1 + 0 + 1 + 0 + 0 + 0 + 0 = 6$$

The norm (||A|| and ||B||) is the length of the vectors, calculated by the formula:

$$||A|| = \sqrt{(1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 0^2 + 0^2 + 0^2 + 0^2 + 0^2}} = \sqrt{9} = 3$$
$$||B|| = \sqrt{(1^2 + 0^2 + 0^2 + 1^2 + 1^2 + 1^2 + 1^2 + 0^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2 + 1^2}} = \sqrt{9} = 3$$

Now we can calculate Cosine Similarity:

$$cosine\ similarity = \frac{6}{3*3} = \frac{6}{9} = 0.67$$

The calculated result is then expressed as a percentage to determine the level of similarity between the answer key text and the student's answer.

$$cosine\ similarity * 100 = 0.67 * 100 = 67\ \%$$

After the manual calculations were completed, the same process was carried out using Python programming to enable a comparison between the manual calculation results and the program-generated results, as presented in Tabel 4 below.

Table 4. Cosine similarity calculation results

Component	Expression	Value
Answer Key Vector	A = (1,1,1,1,1,1,1,1,1,0,0,0,0)	—
Student Answer Vector	*B = (1,0,0,1,1,1,1,0,1,1,1,1,1)*	—
Dot Product	$\Sigma A_i B_i$	6
Norm of A	$\|\|A\|\| = \sqrt{(\Sigma A_i^2)}$	*3*
Norm of B	$\|\|B\|\| = \sqrt{(\Sigma B_i^2)}$	3
Cosine Similarity	*(A · B) / (||A|| × ||B||)*	*0.67*
Similarity Percentage	0.67 × 100	67.00%

The results of the Cosine Similarity calculation using the Python programming language are shown in Tabel 4, indicating a similarity value of 67%, which is the same as the result obtained from the manual calculation carried out previously.

Jaccard Similarity

In the Jaccard Similarity calculation process, the first step is to separate each sentence into a set of words, ignoring any repeated words, as follows:
Answer key: Tujuannya adalah membuat interaksi lebih mudah, efisien, dan menyenangkan.
Student Answer: Tujuannya yaitu melakukan interaksi menjadi lebih mudah, efisien serta menyenangkan

Table 5. Word separation

Answer key	Student Answer
Tujuannya	Tujuannya
adalah	yaitu
membuat	melakukan
interaksi	interaksi
lebih	menjadi
mudah	Lebih
efisien	mudah
dan	efisien
menyenangkan	serta
	menyenangkan

After separating the words, the second step is to calculate the intersection and union of the sets. The intersection consists of the words that appear in both sentences (the answer key and the student's answer), while the union consists of all unique words that appear in both sentences. These results can be seen in Table 6 below.

Table 6. Intersection and union

Answer key	Student Answer
Tujuannya	Tujuannya
interaksi	Adalah
lebih	Membuat
mudah	interaksi
efisien	Lebih
menyenangkan	mudah
	efisien
	Dan
	Menyenangkan
	Yaitu
	Melakukan
	menjadi
	serta

Thus, the number of intersection words is 6, while the total number of unique words in the union is 13. After obtaining the number of intersection words and the number of union words, the third step is to calculate the Jaccard Similarity. The formula for Jaccard Similarity is as follows:

$$Jaccard\ Similarity = \frac{Number\ of\ words\ in\ the\ segment}{Total\ number\ of\ compound\ words} \quad (2)$$

$$Jaccard\ similarity = \frac{6}{13} \approx 0.4615$$

The result is then converted into a percentage to obtain the similarity percentage between the answer key text and the student's answer.

$$Jaccard\ similarity * 100 = 0.4615 * 100 = 46.15\ \%$$

After completing the manual calculations, the same calculations were then performed using Python programming in order to compare the manual results with the program-generated results, as shown in Table 7 below.

Table 7. Jaccard similarity calculation results

Component	Expression	Value
Answer Key Set	A = {Tujuannya, adalah, membuat, interaksi, lebih, mudah, efisien, dan, menyenangkan}	9 words
Student Answer Set	B = {Tujuannya, yaitu, melakukan, interaksi, menjadi, lebih, mudah, efisien, serta, menyenangkan}	10 words
Intersection	|A ∩ B|	6
Union	|A ∪ B|	13
Jaccard Similarity	|A ∩ B| / |A ∪ B|	6/13 ≈ 0.4615
Similarity Percentage	0.4615 × 100	46.15%

The results of the Jaccard Similarity calculation using the Python programming language are shown in Table 7, indicating a similarity value of 46.15%, which is the same as the result obtained from the manual calculation carried out previously.

Dice's Coefficient

In the Dice's coefficient calculation process the first step is Tokenization and Compilation of Word Sets which are,
Answer key: Tujuannya adalah membuat interaksi lebih mudah, efisien, dan menyenangkan.
Student Answer: Tujuannya yaitu melakukan interaksi menjadi lebih mudah, efisien serta menyenangkan

Table 8. Tokenization and word set compilation

Answer key	Student Answer
Tujuannya	Tujuannya
adalah	yaitu
membuat	melakukan
interaksi	interaksi
lebih	menjadi
mudah	Lebih
efisien	mudah
dan	efisien
menyenangkan	serta
	menyenangkan

After separating the words, the second step is to calculate the intersection and union of the sets. The intersection refers to the words that appear in both sentences (the answer key and the student's answer), while the union consists of all unique words found in both sentences. These results can be seen in Table 9 below.

Table 9. Intersection and union

Answer key	Student Answer
Tujuannya	Tujuannya
interaksi	Adalah
lebih	Membuat
mudah	interaksi
efisien	Lebih
menyenangkan	mudah
	efisien
	Dan
	Menyenangkan
	Yaitu
	Melakukan
	menjadi
	serta

As a result, the number of words in the intersection is 6, and the total number of unique words in the union is 13.

After getting the number of Intersection words, the third step is to calculate Dice's coefficient.
The Dice's Coefficient formula is as follows:
|A∩B| =6 (number of words in the intersection)
|A| =9 (number of words in Answer key)
|B|=10 (number of words in Answer key)

$$Dice's\ Coeficient = \frac{2 \cdot |\ A \cap\ B\ |}{\cdot |\ A| + |\ B\ |} \tag{3}$$

$$Dice's\ Coeficient = \frac{2 * 6}{9 + 10} = \frac{12}{19} \approx 0.6316$$

The result is then converted into a percentage to obtain the similarity percentage between the answer key text and the student's answer.

$$Dice's\ similarity * 100 = 0.6316 * 100 = 63.16\ \%$$

After completing the manual calculations, the same calculations were then performed using Python programming in order to compare the manual results with the program-generated results, as shown in Table 10 below.

Table 10. Dice's coefficient calculation results

Component	Expression	Value
Answer Key Set	A = {Tujuannya, adalah, membuat, interaksi, lebih, mudah, efisien, dan, menyenangkan}	9 words
Student Answer Set	B = {Tujuannya, yaitu, melakukan, interaksi, menjadi, lebih, mudah, efisien, serta, menyenangkan}	10 words
Intersection	|A ∩ B|	6
Total Size	|A| + |B|	19
Dice's Coefficient	2 × |A ∩ B| / (|A| + |B|)	12/19 ≈ 0.6316
Similarity Percentage	0.6316 × 100	63.16%

The results of the Dice's Coefficient calculation using the Python programming language are shown in Figure 4, indicating a similarity value of 63.16%, which is the same as the result obtained from the manual calculation carried out previously.

Matching Coefficient

In the Matching Coefficient calculation process, the first step is tokenization and the construction of word sets, as follows:
Answer key: Tujuannya adalah membuat interaksi lebih mudah, efisien, dan menyenangkan.
Student Answer: Tujuannya yaitu melakukan interaksi menjadi lebih mudah, efisien serta menyenangkan

Table 11. Tokenization and word set compilation

Answer key	Student Answer
Tujuannya	Tujuannya
adalah	yaitu
membuat	melakukan
interaksi	interaksi
lebih	menjadi
mudah	Lebih
efisien	mudah
dan	efisien
menyenangkan	serta
	menyenangkan

After performing tokenization and constructing the word sets, the next step is to calculate the matching set. The matching set consists of words that appear in both sentences. In the Matching Coefficient method, only words that are exactly the same are taken into account, without considering their order or frequency of occurrence. The third step is then to calculate the Matching Coefficient. The formula for the Matching Coefficient is as follows:

$$Matching\ \ Coeficient = \frac{number\ of\ matching\ words}{The\ number\ of\ words\ in\ the\ combined\ set} \quad (4)$$

$$Matching\ \ Coeficient = \frac{6}{13} \approx 0.46$$

The result is then converted into a percentage to obtain the similarity percentage between the answer key text and the student's answer.

$$Matching\ \ coeficient * 100 = 0.46 * 100 = 46\ \%$$

After completing the manual calculations, the same calculations were then performed using Python programming in order to compare the manual results with the program-generated results, as shown in Table 12 below.

Table 12. Matching coefficient calculation results

Component	Expression	Value
Answer Key Set	A = {Tujuannya, adalah, membuat, interaksi, lebih, mudah, efisien, dan, menyenangkan}	9 words
Student Answer Set	B = {Tujuannya, yaitu, melakukan, interaksi, menjadi, lebih, mudah, efisien, serta, menyenangkan}	10 words
Matching Elements	|A ∩ B|	6
Union	|A ∪ B|	13
Matching Coefficient	|A ∩ B| / |A ∪ B|	6/13 ≈ 0.4615
Similarity Percentage	0.4615 × 100	46.15%

The results of the Matching Coefficient calculation using the Python programming language are shown in Figure 5, indicating a similarity value of 46.15%, which is the same as the result obtained from the manual calculation carried out previously.

Overlap Coefficient

In the Matching Coefficient calculation process, the first step is tokenization and the construction of word sets, as follows:
Answer key: Tujuannya adalah membuat interaksi lebih mudah, efisien, dan menyenangkan.
Student Answer: Tujuannya yaitu melakukan interaksi menjadi lebih mudah, efisien serta menyenangkan

Table 13. Tokenization and word set compilation

Answer key	Student Answer
Tujuannya	Tujuannya
adalah	*yaitu*
membuat	melakukan
interaksi	*interaksi*
lebih	menjadi
mudah	*Lebih*
efisien	mudah
dan	*efisien*
menyenangkan	serta
	menyenangkan

After tokenization and word set construction, the next step is to calculate the set intersection (overlap set). The overlap set consists of the words that appear in both sentences. The overlap set is presented in Table 10 below.

Table 14. Iris set (overlap set)

Iris Set (Overlap Set)
Tujuannya
interaksi
lebih
mudah
efisien
menyenangkan

Thus, the number of words in the intersection (overlap set) is 6. The third step is then to calculate the Overlap Coefficient. The formula for the Overlap Coefficient is as follows:

$$Overlap\ \ Coeficient = \frac{The\ number\ of\ words\ with\ overlaps}{\min\ (number\ of\ words\ in\ the\ answer\ key, number\ of\ words\ in\ the\ answer)} \quad (5)$$

The answer key sentence consists of 9 words, while the student's answer sentence consists of 10 words. The number of words in the intersection (overlap set) is 6.

$$Overlap\ Coeficient = \frac{6}{\min(9, 10)} = \frac{6}{9} \approx 0.6667$$

The calculated result is then expressed as a percentage to determine the level of similarity between the answer key text and the student's answer.

$$Overlap\ similarity * 100 = 0.6667 * 100 = 66.67\ \%$$

After completing the manual calculations, the same calculations were then performed using Python programming in order to compare the manual results with the program-generated results, as shown in Table 15 below.

Table 15. Overlap coefficient calculation results

Component	Expression	Value
Answer Key Set	A = {Tujuannya, adalah, membuat, interaksi, lebih, mudah, efisien, dan, menyenangkan}	9 words
Student Answer Set	B = {Tujuannya, yaitu, melakukan, interaksi, menjadi, lebih, mudah, efisien, serta, menyenangkan}	10 words
Intersection	|A ∩ B|	6
Minimum Size	min(|A|, |B|)	9
Overlap Coefficient	|A ∩ B| / min(|A|, |B|)	6/9 ≈ 0.6667
Similarity Percentage	0.6667 × 100	66.67%

The results of the Overlap Coefficient calculation using the Python programming language are shown in Figure 6, indicating a similarity value of 66.67%, which is the same as the result obtained from the manual calculation carried out previously. In this study, the researchers compared five similarity methods to evaluate their effectiveness in Automatic Short Answer Grading (ASAG). Based on the results obtained from both the manual calculations and the Python-based calculations, Cosine Similarity demonstrated the best performance in measuring the similarity between answer keys and student answers. This is due to its ability to account for sentence structure similarity and the use of relevant keywords, despite variations in word choice and the presence of synonyms.

The Jaccard Similarity method produced fairly good results, but it was less effective in handling word variations or the use of synonyms in student answers. This is because Jaccard only considers the words that appear in both sets, without taking into account word frequency or order. Dice's Coefficient and Overlap Coefficient produced results that were quite similar to those of Jaccard, with a slight improvement in accuracy, especially for longer and more complex answers. Both methods measure similarity in a comparable way, but Dice's Coefficient yielded better results in scoring answers with more varied lengths.

On the other hand, the Matching Coefficient performed worse than the other methods and showed the lowest level of accuracy. This may be due to its inability to account for variations in word order and word frequency, which are important factors in the context of automatic answer assessment.

which is the same as the result obtained from the manual calculation carried out previously.

Table 16. Summary of the similarity scores

Method	Score	Similarity Percentage
Cosine Similarity	0.6700	67.00%
Overlap Coefficient	*0.6667*	*66.67%*
Dice's Coefficient	0.6316	63.16%
Jaccard Similarity	*0.4615*	*46.15%*
Matching Coefficient	0.4615	46.15%

As shown in Table 16, Cosine Similarity achieved the highest similarity score (67.00%), followed by Overlap Coefficient (66.67%) and Dice's Coefficient (63.16%). Jaccard Similarity and Matching Coefficient produced the lowest scores at 46.15%. These results indicate that vector-based methods such as Cosine Similarity are more sensitive to structural variation in short answers compared to set-based methods.

Overall, the results of this study indicate that more advanced methods, such as Cosine Similarity, are more suitable for Automatic Short Answer Grading applications than simpler methods such as Jaccard Similarity or Matching Coefficient. However, the selection of the most appropriate similarity method also depends greatly on the type of data used, the length of the answers, and the expected level of similarity between the student's response and the answer key.

Table 17. Comparison of related studies on textual similarity methods for ASAG

Study	Methods Compared	Evaluation Metric	Main Findings	Result
Wahyuningsih et al. [20]	Cosine, Jaccard, Dice	Correlation	Cosine and Dice Similarity produced the best performance in short answer grading	Similarity = N/A, Correlation = 0.76
Lubis et al. [21]	Word Embedding Similarity	Correlation, MAE	Semantic similarity improved grading accuracy	Similarity = N/A, Correlation = 0.70, MAE = 0.70
Goenka et al. [24]	Explainable scoring methods	N/A	Method selection affects grading reliability	Similarity = N/A, Correlation = N/A
This Study	Cosine, Jaccard, Dice, Overlap, Matching	Similarity Score	Cosine Similarity achieved the highest similarity score	Similarity = 67.00%, Correlation = N/A

Although lexical similarity methods provide efficient and interpretable results, they may be limited in capturing deeper semantic relationships compared to modern embedding-based approaches.

4. CONCLUSION

This article discusses the application of text mining techniques in Automatic Short Answer Grading (ASAG) by comparing five similarity methods, namely Cosine Similarity, Jaccard Similarity, Dice's Coefficient, Overlap Coefficient, and Matching Coefficient. The results of the study, based on both manual calculations and Python-based verification, show that Cosine Similarity is the most effective method, achieving the highest similarity score of 67.00%. Overlap Coefficient produced a comparable result at 66.67%, followed by Dice's Coefficient at 63.16%, while Jaccard Similarity and Matching Coefficient produced the lowest scores at 46.15%. Overall, this study confirms that vector-based methods such as Cosine Similarity are more suitable for ASAG applications than simpler set-based methods. The selection of an appropriate method should take into account the type of data, the length of the answers, and the expected level of similarity.

Despite these findings, this study has several limitations. The dataset used is relatively small, consisting of only five definition-based questions answered by 25 students from a single course, which may limit the generalizability of the findings. Additionally, the methods evaluated in this study are purely lexical and do not account for semantic relationships between words, which may result in lower scores for answers that use synonyms or paraphrased expressions.

Future work should consider expanding the dataset to include a larger number of questions, students, and course subjects to improve the robustness of the evaluation. Furthermore, the integration of semantic similarity techniques, such as word embedding-based approaches or pre-trained language models, could enhance the accuracy of ASAG systems by capturing deeper linguistic relationships beyond exact word matching. A hybrid approach combining lexical and semantic similarity methods may represent a promising direction for developing more accurate and adaptive ASAG systems

REFERENCES

[1] S. Eybers and H. Kahts, "In Search of Insight from Unstructured Text Data: Towards an Identification of Text Mining Techniques BT - Digital Science," T. Antipova, Ed., Cham: Springer International Publishing, 2022, pp. 591–603.

[2] B. Goswami, N. Bhavsar, S. A. Alzobidy, B. Lavanya, R. U. Kumar, and K. Rajapandian, *Sentiment Analysis Using Natural Language Processing*. 2025. doi: 10.1002/9781394272464.ch20.

[3] F. G. Pedrosa and J. S. dos Reis, "Quantitative analysis of qualitative data: using text mining techniques for the music therapy clinic," 2023, doi: 10.56238/homeiisevenhealth-052.

[4] A. Hermawan, I. Jowensen, J. Junaedi, and Edy, "Implementasi Text-Mining untuk Analisis Sentimen pada Twitter dengan Algoritma Support Vector Machine," *JST (Jurnal Sains dan Teknol.*, vol. 12, no. 1, pp. 129–137, 2023, doi: 10.23887/jstundiksha.v12i1.52358.

[5] M. Kaya and I. Cicekli, "A Hybrid Approach for Automated Short Answer Grading," *IEEE Access*, vol. 12, no. June, pp. 96332–96341, 2024, doi: 10.1109/ACCESS.2024.3420890.

[6] D. R. Thomas, S. Gupta, and K. R. Koedinger, "Comparative Analysis of Learnersourced Human-Graded and AI-Generated Responses for Autograding Online Tutor Lessons," *Commun. Comput. Inf. Sci.*, vol. 1831 CCIS, pp. 714 – 719, 2023, doi: 10.1007/978-3-031-36336-8_110.

[7] P. Akhilesh, K. Amal Krishna, S. K. Bharadwaj, and M. Venugopalan, "Automated Short Answer Grading With Word Embedding-Based Semantic Similarity Using PySpark," *RAICS - IEEE Recent Adv. Intell. Comput. Syst.*, no. 2024, 2024, doi: 10.1109/RAICS61201.2024.10690067.

[8] U. Padó, "Assessing the Practical Benefit of Automated Short-Answer Graders," *Lect. Notes Comput. Sci.*, vol. 13356 LNCS, pp. 555 – 559, 2022, doi: 10.1007/978-3-031-11647-6_114.

[9] N. S. Lagutina and K. V Lagutina, "A Survey of Models for Automatic Assessment of Similarity of Student's Answer to the Reference Answer," *Autom. Control Comput. Sci.*, vol. 59, no. 7, pp. 1152 – 1169, 2025, doi: 10.3103/S0146411625700427.

[10] A. R. Borah, R. S. S. Dev, B. M. Suprathik, A. R. Harshini, Y. Boggula, and V. Charitha, "Automated Models in Educational Assessment: A Comprehensive Survey," in *Proceedings of the 9th International Conference on Communication and Electronics Systems, ICCES 2024*, 2024, pp. 1029 – 1034. doi: 10.1109/ICCES63552.2024.10859689.

[11] A. S. Bhatia and L. Yadav, "Design and Implementation of Deep Learning Algorithms for Automatic Grading of Student's Essay," in *International Conference on Engineering, Technology and Management, ICETM 2025*, 2025. doi: 10.1109/ICETM63734.2025.11051991.

[12] J. Schneider, R. Richner, and M. Riser, "Towards Trustworthy AutoGrading of Short, Multi-lingual, Multi-type Answers," *Int.*

J. Artif. Intell. Educ., vol. 33, no. 1, pp. 88–118, 2023, doi: 10.1007/s40593-022-00289-z.

[13] A. Melnyk *et al.*, "AI-Powered Knowledge Assessment: Application of GPT Models in Educational Testing," in *Proceedings - International Conference on Advanced Computer Information Technologies, ACIT*, 2025, pp. 997 – 1001. doi: 10.1109/ACIT65614.2025.11185592.

[14] P. Isaias, P. Miranda, and S. Pifano, "E-Assessment Systems: An Evaluation Framework from the Perspective of Higher Education Experts," in *25th International Symposium on Computers in Education, SIIE 2023*, 2023. doi: 10.1109/SIIE59826.2023.10423677.

[15] M. Kaya and I. Cicekli, "A Hybrid Approach for Automated Short Answer Grading," *IEEE Access*, vol. 12, pp. 96332 – 96341, 2024, doi: 10.1109/ACCESS.2024.3420890.

[16] C. Zhao, M. Silva, and S. Poulsen, "Language Models are Few-Shot Graders," *Lect. Notes Comput. Sci.*, vol. 15880 LNAI, pp. 3 – 16, 2025, doi: 10.1007/978-3-031-98459-4_1.

[17] S. Bonthu, S. R. Sree, and M. H. M. Krishna Prasad, "Framework for automation of short answer grading based on domain-specific pre-training," *Eng. Appl. Artif. Intell.*, vol. 137, 2024, doi: 10.1016/j.engappai.2024.109163.

[18] A. Alqurashi, B. Alharbi, and S. Sabbeh, "An Automatic Grading System for Arabic Language Short-Answer Questions Using Deep Learning," *Eng. Technol. Appl. Sci. Res.*, vol. 15, no. 5, pp. 26665 – 26675, 2025, doi: 10.48084/etasr.10917.

[19] K. Islam, P. Ahmadi, and S. Yousaf, "Assessment formats and student learning performance: What is the relation?," *2017 Res. Eng. Educ. Symp. REES 2017*, pp. 1–8, 2017.

[20] T. Wahyuningsih, Henderi, and Winarno, "Text Mining an Automatic Short Answer Grading (ASAG), Comparison of Three Methods of Cosine Similarity, Jaccard Similarity and Dice's Coefficient," *J. Appl. Data Sci.*, vol. 2, no. 2, pp. 45–54, 2021, doi: 10.47738/jads.v2i2.31.

[21] F. F. Lubis *et al.*, "Automated Short-Answer Grading using Semantic Similarity based on Word Embedding," *Int. J. Technol.*, vol. 12, no. 3, pp. 571–581, 2021, doi: 10.14716/ijtech.v12i3.4651.

[22] M. Putnikovic and J. Jovanovic, "Embeddings for Automatic Short Answer Grading: A Scoping Review," *IEEE Trans. Learn. Technol.*, vol. 16, no. 2, pp. 219–231, 2023, doi: 10.1109/TLT.2023.3253071.

[23] A. Ayaan and K. W. Ng, "Automated grading using natural language processing and semantic analysis," *MethodsX*, vol. 14, no. October 2024, p. 103395, 2025, doi: 10.1016/j.mex.2025.103395.

[24] P. Goenka, M. Piplani, R. Sawhney, P. Mathur, and R. R. Shah, "ESAS: Towards Practical and Explainable Short Answer Scoring," *AAAI 2020 - 34th AAAI Conf. Artif. Intell.*, pp. 13797–13798, 2020.

[25] M. Srivastava, A. K. Srivastava, R. Garg, and P. K. Mishra, "Performance Evaluation of the MapReduce-based Parallel Data Preprocessing Algorithm in Web Usage Mining with Robot Detection Approaches," *IETE Tech. Rev. (Institution Electron. Telecommun. Eng. India)*, vol. 39, no. 4, pp. 865 – 879, 2022, doi: 10.1080/02564602.2021.1918584.

[26] R. Rani and D. K. Lobiyal, "Performance evaluation of text-mining models with Hindi stopwords lists," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 6, pp. 2771 – 2786, 2022, doi: 10.1016/j.jksuci.2020.03.003.

[27] V. P. Kadam, K. B. Khandale, and N. M. C, "Text Analysis and Classification for Preprocessing Phase of Automatic Text Summarization Systems," *Commun. Comput. Inf. Sci.*, vol. 1572 CCIS, pp. 382 – 396, 2022, doi: 10.1007/978-3-031-05767-0_30.

[28] H. Häntze, M. Buser, A. Hering, L. C. Adams, and K. K. Bressem, "Sex-Based Bias Inherent in the Dice Similarity Coefficient: A Model Independent Analysis for Multiple Anatomical Structures," *Lect. Notes Comput. Sci.*, vol. 15976 LNCS, pp. 125 – 134, 2026, doi: 10.1007/978-3-032-05870-6_13.

[29] M. Kryszkiewicz, "A New Approach to Deriving Jaccard Similarity and Jaccard Distance Properties with and without Considering Feature Weights," *Commun. Comput. Inf. Sci.*, vol. 2145 CCIS, pp. 341 – 349, 2024, doi: 10.1007/978-981-97-5934-7_29.

[30] H. Steck, L. Gatos, U. States, and H. Steck, "Is Cosine-Similarity of Embeddings Really About Similarity ?," vol. 2024, no. May 2024, 2026, doi: 10.1145/3589335.3651526.

.