Matrik: Jurnal Manajemen, Teknik Informatika, dan Rekayasa Komputer
Vol. 22, No. 2, Maret 2023, pp. 325∼338
ISSN: 2476-9843, accredited by Kemenristekdikti, Decree No: 200/M/KPT/2020
DOI: 10.30812/matrik.v22i2.2473

Essay Auto-scoring using N-Gram and Jaro Winkler Based Indonesian Typos

Herlina Jayadianti[1], Budi Santosa[1], Judanti Cahyaning[1], Shoffan Saifullah[1,2], Rafal Drezewski[2]
[1]Universitas Pembangunan Nasional Veteran Yogyakarta, Yogyakarta, Indonesia
[2]AGH University of Science and Technology,Cracow, Poland

Article Info

Article history:

Received October 30, 2022
Revised January 25, 2023
Accepted February 28, 2023

Keywords:

Automation
Spelling error detection and correction
N-Gram
Jaro Winkler

ABSTRACT

Writing errors on e-essay exams reduce scores. Thus, detecting and correcting errors automatically in writing answers is necessary. The implementation of Levenshtein Distance and N-Gram can detect writing errors. However, this process needed a long time because of the distance method used. Therefore, this research aims to hybrid Jaro Winker and N-Gram methods to detect and correct writing errors automatically. This process required preprocessing and finding the best word recommendations by the Jaro Winkler method, which refers to Kamus Besar Bahasa Indonesia (KBBI). The N-Gram method refers to the corpus. The final scoring used the Vector Space Model (VSM) method based on the similarity of words between the answer keys and the respondents answers. Datasets used 115 answers from 23 respondents with some writing errors. The results of Jaro Winkler and N-Gram methods are good in detecting and correcting Indonesian words with the accuracy of detection averages of 83.64% (minimum of 57.14% and maximum of 100.00%). In contrast, the error correction accuracy averages 78.44% (minimum of 40.00% and maximum of 100.00%). However, Natural Language Processing (NLP) needs to improve these results for word recommendations.

Corresponding Author:

Budi Santosa, +62 815-6027-294,
Faculty of Industrial Engineering, Department of Informatics,
Universitas Pembangunan Nasional Veteran Yogyakarta, Yogyakarta, Indonesia,
Email: dissan@upnyk.ac.id

How to Cite:H. Jayadianti, B. Santosa, J. Cahyaning, S. Saifullah, and R. Drezewski, "Essay auto-scoring using N-Gram and Jaro Winkler based Indonesian Typos", *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 22, no. 2, pp. 325-338, Mar. 2023.

1. INTRODUCTION

Exams are one way to assess the level of student understanding of the material provided [1]. One type of examination is an essay exam, which can be conducted online. The essay exam does not give answer choices but requires writing down the answer in the form of a sentence description [2]. If the assessment is done manually, it takes a long time, allowing non-objective reviews to occur, and the quality decreases [3]. Thus, it can be overcome by automatically evaluating essays using a computer [4, 5].

Students require accuracy in writing answers [6] and no writing errors because that can reduce the assessment exam if using an automatic essay scoring system [7, 8]. However, several writing errors include a lack of concentration in typing, adjacent keyboard positions, and a lack of understanding of correct words [9–12]. In addition, there are four writing errors: insertion, deletion, adding, and changing the position of letters in a sentence. Therefore, writing mistakes in answering essay exams must be overcome by detecting and correcting writing errors [13].

Research conducted to overcome writing errors using bigrams and trigrams for writing errors [14, 15] in English obtained good results with 71%-79% precision and 81%-88% recall. In addition, this research can also detect and correct more than one word in a sentence. However, many bigrams and trigrams have not been discovered due to lacking an n-gram corpus [16]. Another research uses bigrams and trigrams in Indonesian [17, 18]. This research added the additive smoothing method to the n-gram probability calculation [19, 20]. This study resulted in a reasonably low accuracy of 11% due to the lack of quantity of n-gram corpus. An increase in the occurrence of 0 of the n-gram calculation probability is due to the additive smoothing method. Another research compared several techniques, such as the hamming distance, Levenshtein distance, Damerau Levenshtein distance, and Jaro Winkler distance [21], where the test used an evaluation relevance judgment. The results of this study indicate that the Jaro Winkler method obtains the highest Mean Average Precision (MAP) value of 0.87 [22] compared to other forms [23, 24].

Further research uses the Jaro Winkler method, implemented in essay scoring. This method can detect three types of writing errors: excess letters, lack of letters, and misplacement of letters in a sentence. This research obtained an accuracy of 57%-73% [25]. Based on the methods from previous research, we proposed a combined method of n-gram and Jaro-Winkler methods to detect and correct writing errors in the essay scoring system. The use of n-grams in this research is due to previous research obtaining reasonably good accuracy and having the ability to detect more than one word in a sentence. In addition, the Jaro Winkler can also run faster processes than the Levenshtein distance algorithm [21, 26]. Therefore, this research aims to hybrid Jaro Winker and N-Gram methods to detect and correct writing errors automatically.

2. RESEARCH METHOD

This research method consists of data collection and data processing. Data processing consists of several stages: data preprocessing, making an n-gram corpus, writing error detection, and correction and scoring.

2.1. Data Collection

Data collection is one of the stages used to find the information needed and used as a basis or support in research. This research uses three types of data from several sources such as:

a. Indonesian Dictionary
   This dictionary is sourced from Kamus Besar Bahasa Indonesia (KBBI) versions 3 and 4, published by Balai Pustaka. This dictionary is used as a reference in detecting and correcting writing errors previously carried out by case folding and filtering processes by removing hyphens. The total words used are 43,562 words.
b. N-Gram Corpus
   The n-gram corpus results from processing Indonesian articles using the n-gram method as a reference in the detection and correction process [27]. In addition, the n-gram word corpus is also obtained from the n-gram processing process in the Kamus Besar Bahasa Indonesia (KBBI). The total data used are 52828 unigrams, 110412 bigrams, and 69787 trigrams.
c. Test Data
   Test data is sourced from a questionnaire that answers questions about artificial intelligence, machine learning, and artificial neural networks that have been filled in by several respondents who are then given an error scenario in the testing process. The test data used was 115 answers from 23 respondents.

2.2. Data Processing

Data processing is a process that starts from raw data processing and finishes when final results are obtained [28]. The stages of data processing are shown in Figure 1.

a. Data Preprocessing
Data preprocessing is the initial stage of processing data so that the data is ready to be processed. Some of the preprocessing data stages are as follows, and the sample result of this process can be seen in Table 1.

1. Case Folding
Case folding is a method for converting all uppercase characters to lowercase [29]. Only letters a-z are accepted.

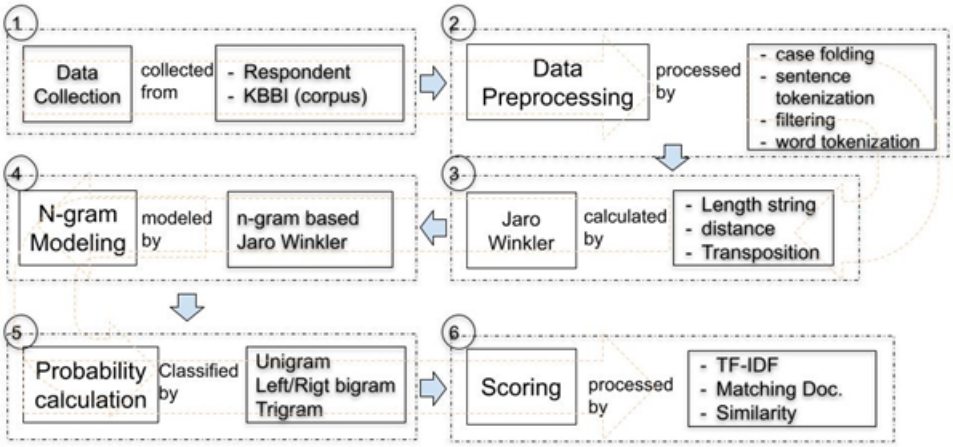

Figure 1. Block Diagrams In Data Processing (Scoring)

2. Sentence Tokenization
Tokenization is a process of separating or cutting the constituent text [30]. This tokenization requires a delimiter or separator, such as spaces, from one word to another. Meanwhile, the tokenization of this sentence separates the sentences that make up the paragraph with a dot delimiter or separator.
3. Filtering
Filtering is the process of retrieving important words or unimportant words [31]. However, this research conducted filtering to remove punctuation. The punctuation marks are characters other than letters and spaces.
4. Word Tokenization Like sentence tokenization, word tokenization is used to cut words in a sentence with a space delimiter or separator. The result of this process is a word token where an underscore ("_") is given at the beginning and end of the sentence.

Table 1. The Example of Data Preprocessing

Before preprocessing	**After Preprocessing**
Data processing is the processing of raw data into information	_ data processing is the processing raw data into information _

b. N-Gram Corpus N-Gram corpus is the result of processing Indonesian articles based on Kamus Besar Bahasa Indonesia using the N-Gram method to reference writing errors' detection and correction process. Several stages are as follows:

1. Data Preprocessing
   Data preprocessing is the initial process that needs to be done to prepare the data for the next process that has been described previously.
2. N-Gram Modelling
   N-gram is an algorithm that pays attention to the adjacent n-letter sequence by dividing a sentence or word into smaller forms and then calculating the probability [32]. There are several types of n-grams that can be used, namely unigram with a value of n=1, bigram with a value of n=2, trigram with a value of n=3, and quadram with a value of n=4 [33]. However, in this study, only three types of n-grams were used: unigram, bigram, and trigram. The n-gram modeling is presented in Figure 2.

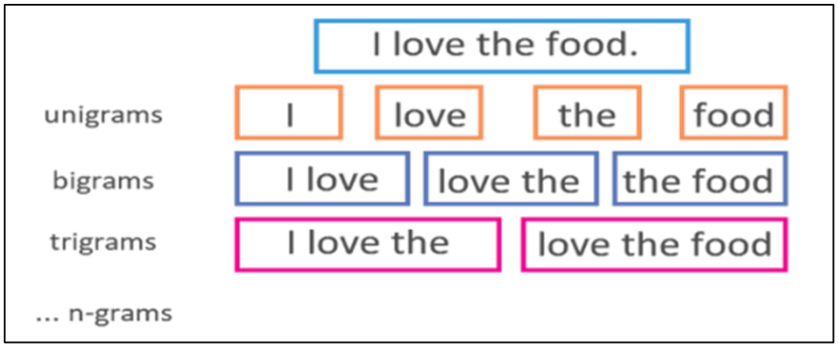

Figure 2. The Illustration of N-Gram Modelling [34]

   Based on Fig. 1, the sentence I love the food is modeled into three types of n-grams. Unigram with a value of n = 1, which means it only consists of 1 word. Bigram with a value of n = 2, which means it consists of 2 words taken from the word at position n with the next word. The trigram with a value of n=3 consists of 3 words taken from the word at position n with the next two words.
3. Calculate Word Occurrence
   This process is carried out by counting the occurrences of words in each type of n-gram in the processed text and entering the processed results into the database. An example of the results of making an n-gram corpus is in Table 2.

Table 2. The Occurrence Number of Unigram

Unigram	Occurrences Number	Bigram	Occurrences Number	Trigram	Occurrences Number
_	4	_ data	1	_ data processing	1
data	4	data processing	1	data processing is	1
processing	2	processing is	1	processing is the	1
...	...	...	...	...	...

c. Writing Error Detection and Correction
   The main process in this research is writing error detection and correction using n-gram and Jaro Winkler methods. This process needs references to Kamus Besar Bahasa Indonesia (KBBI) and N-Gram corpus. Several processes are applied as follows:
   1. Data Preprocessing
      Data preprocessing is the initial process that needs to be done to prepare the data for the next process that has been described previously.
   2. Jaro Winkler
      Jaro Winkler is an algorithm used to measure the similarity between two strings being compared where this algorithm is included in the Jaro distance variant [34, 35]. Two strings are said to be the same if the Jaro Winkler value is 1. On the other hand, if the Jaro Winkler value is 0, then there is no similarity between the two strings [36]. According to Yulianingsih [37], there are three steps to calculate the similarity between 2 strings using Jaro Winkler; for example, take the word procesing and then compare it with ”processing.”

- Calculate the length of the string/character
  The word procesing has a character length of 9, and processing has a character length of 10. This step is applied to each word to be compared.
- Calculate common string in 2 strings
  This step needs to calculate theoretical distance as an index reference in the process of searching for the same character and the formula as in (1).

$$theoreticaldistance = (\frac{max(\mid S_1 \mid, \mid S_2 \mid)}{2}) - 1 \tag{1}$$

Based on the example, the words procesing and processing have a theoretical distance of 4, meaning that the match starts from the four leading indexes to the four backward indexes. The process of matching two words is shown in Table 3.

Table 3. Illustration Of Character Similarity Search

S1	**p**	**r**	**o**	**c**	**e**	**s**	**i**	**n**	**g**	
Index	0	1	2	3	4	5	6	7	8	
S2	p	r	o	c	e	s	s	i	n	g
Index	0	1	2	3	4	5	6	7	8	9

Based on Figure. 2, the words procesing and processing have the same nine characters.

- Calculate transposition between 2 strings
  This step of the process is the same as in the process of calculating the same character that can be seen in Fig. 2. The words procesing and processing do not have transposition characters because there are no two same characters in different positions.
  After the basic steps are done, then Jaro distance and Jaro Winkler are calculated. The Jaro Distance can be seen in formula (2) and Jaro Winkler in formula (2).

$$d_j = \frac{1}{3}(\frac{m}{\mid S_1 \mid} + \frac{m}{\mid S_2 \mid} + \frac{m - \frac{t}{2}}{m}) \tag{2}$$

$$d_w = d_j + (lp(1 - d_j)) \tag{3}$$

3. N-Gram Modeling This step is the same as in the process of N-Gram modeling to make an N-Gram corpus. The types of n-grams used are unigram, left bigram, right bigram, and trigram. Words that are assembled with n-grams are words from the recommendation of Jaro Winkler with the text tokens being tested. The n-gram modeling is as Table 4.

Table 4. The Occurrence Number of Unigram

C_j^i		W^{i-1}		W^{i+1}		$W^{i-1}C_j^i$ **(left biagram)**	$C_j^i W^{i+1}$ **(right biagram)**	$W^{i-1}C_j^i W^{i+1}$ **(trigram)**
C_j^0	_							
C_i^i	date	W^0	_	W^2	procesing	_ date	date procesing	_ date procesing
C_2^1	data					_ data	data procesing	_ data procesing
C_1^2	process	W^1	data	W^3	is	data process	process is	data process is
	...	...	...	...	...	...	...	...

C_j^i is the result of word recommendations obtained from Jaro Winkler calculations where i is the word token index, and j is the word recommendation order index from Jaro Winkler. W^{i-1} is the word token at position $n-1$. While W^{i+1} is the word token at position $n+1$. The left bigram is obtained from the combination of W^{i-1} with C_j^i, the right bigram is obtained from the combination of C_j^i with W^{i+1} and the trigram is obtained from W^{i-1} combination with C_j^i and W^{i+1}.

4. Calculate a Probability
   This step needs to count every word occurrence from the n-gram modeling on the n-gram corpus that has been made. If the word is found in n-grams, then the occurrences contained in the corpus are taken. Meanwhile, if no event is found, it will be assigned a value of 0. The example can be seen in Table 5.

Table 5. The Occurrence Number of These Modeling

Occurrence Number of Unigram					
	C_j^i			count (C_j^i)	
C_1^0		-		count (C_1^0)	4
C_1^1		date		count (C_1^1)	0
C_1^2		data		count (C_2^1)	4
C_2^1		process		count (C_1^2)	0
...		...		...	...
Occurrence Number of Left Bigram					
	C_j^i		$W^{i-1}C_j^i$	count $(W^{i-1}C_j^i)$	
C_1^1	date	$W^0C_1^1$	- date	count $(W^0C_1^1)$	0
C_2^1	data	$W^0C_2^1$	- data	count $(W^0C_2^1)$	1
C_1^2	process	$W^0C_1^2$	data process	count $(W^0C_1^2)$	0
...	...	...	...	...	...
Occurrence Number of Right Bigram					
	C_j^i		$C_j^iW^{i+1}$	count $(C_j^iW^i+1)$	
C_1^1	date	$C_1^1W^2$	date procesing	count $(C_1^1W^2)$	0
C_2^1	data	$C_2^1W^2$	data procesing	count $(C_2^1W^2)$	0
C_1^2	process	$C_j^2W^3$	process is	count $(C_1^2W^3)$	0
...	...	...	...	...	...
Occurrence Number of Trigram					
	C_j^i		$W^{i-1}C_j^iW^{i+1}$	count $(W^{i-1}C_j^iW^i+1)$	
C_1^1	date	$W^0C_1^1W^2$	- date procesing	count $(W^0C_1^1W^2)$	0
C_2^1	data	$W^0C_2^1W^2$	- data procesing	count $(W^0C_2^1W^2)$	0
C_1^2	process	$W^0C_1^2W^3$	data process is	count $(W^0C_1^2W^3)$	0
...	...	...	...	...	...

The number of occurrences that have been obtained in each type of n-gram is followed by the calculation of the total occurrence. $\Sigma_{r=1}^{k_i}\ count(C_r^i)$ is the total occurrence of the unigram, $\Sigma_{r=1}^{k_i}\ count(W^{i-1}C_r^i$ is the total occurrence of the left bigram, $\Sigma_{r=1}^{k_i}\ count(C_r^iW^{i+1})$ is the total occurrence of the right bigram, and $\Sigma_{r=1}^{k_i}\ count(W^{i-1}C_r^iW^{i+1})$ is the total number of occurrences of the trigram. k is the total words calculated by Jaro Winkler, which have been modeled with n-grams in each word token. r is a word order calculated by Jaro Winkler, which has been modeled with n-grams in each word token. The total occurrence of n-grams can be seen in Table 6.

Table 6. The Total Occurrence Number of N-Gram

W^i	**Token**	$\Sigma_{r=1}^{ki} count(C_r^i)$ **(Unigram)**	$\Sigma_{r=1}^{ki} count(W^{i-1}C_r^i)$ **(Left Bigram)**	$\Sigma_{r=1}^{ki} count(C_r^iW^{i+1})$ **(Right Bigram)**	$\Sigma_{r=1}^{ki} count(W^{i-1}C_r^IW^{i+1})$ **(Trigram)**
W^1	data	$count(C_1^1) + count(C_2^1)$ $= 1+0=1$	$count(W^0C_1^1)$ $+count(W^0C_2^1)$ $= 1+0=1$	$count(C_1^1W^2)$ $+count(C_2^1W^2)$ $= 1+0=1$	$count(W^0C_1^1W^2)$ $+count(W^0C_2^1W^2)$ $1+0=1$
W^2	procesing	$count(C_1^2)=4$	$count(W^1C_1^2)=1$	$count(C_1^2W^3)=1$	$count(W^1C_1^2W^3)=1$
...	...	...	...	...	...

If all occurrences have been totaled, the process can be continued by calculating probabilities. Probability calculations can use Markov Chain assumptions where the probability of the next word depends on the previous word [16]. From these assumptions, the probability calculation uses the Maximum Likelihood Estimation (MLE) by dividing the occurrence by the total occurrences so that the results obtained are between 0 and 1 [38]. The formulas of Maximum Likelihood Estimation (MLE) can be seen in formula (4) for Unigram, formula (5) for Left Bigram, formula (6) for Right Bigram, and formula (7) for Trigram.

Unigram

$$P_0(C_j^i) = \frac{\text{count } (C_j^i)}{\Sigma_{r=1}^{ki} \text{count } (C_j^i)} \tag{4}$$

Left Bigram

$$P_1(C_j^i \mid W^{i-1}) = \frac{\text{count } (W^{i-1}C_j^i)}{\Sigma_{r=1}^{ki} \text{ count } (W^{i-1}C_r^i)} \tag{5}$$

Right Biagram

$$P_2(C_j^i \mid W^{i+1}) = \frac{\text{count } (W^{i+1}C_j^i)}{\Sigma_{r=1}^{ki} \text{ count } (W^{i+1}C_r^i)} \tag{6}$$

Trigram

$$P_3(C_j^i \mid W^{i-1}, W^{i+1}) = \frac{\text{count } (W^{i-1}C_j^i W^{i+1})}{\Sigma_{r=1}^{ki} \text{ count } (W^{i-1}C_r^i W^{i+1})} \tag{7}$$

Probability calculations using MLE need to be continued with probability calculations using the smoothing Jelinek Mercer method, which is used to overcome the 0 probability calculation results [32]. The Jelinek Mercer method uses an interpolation method associated with the n-gram levels and calculated by the formula (8, 9, 10) [25].

Left Bigram

$$P_1'(W^{i-1}) = \lambda P_1(W^{i-1}) + (1-\lambda)P_0(C_j^i) \tag{8}$$

Right Bigram

$$P_2'(W^{i+1}) = \lambda P_2(W^{i+1}) + (1-\lambda)P_0(C_j^i) \tag{9}$$

Trigram

$$P_3'(W^{i-1}, W^{i+1}) = \lambda P_3(W^{i-1}, W^{i+1}) + (1-\lambda)(P_1(W^{i+1})) + \frac{P_2(W^{i+1})}{2} \tag{10}$$

The example of the results of the probability calculation using Jelinek Mercer is presented in Table 7.

Table 7. The Probability Calculation Using Jelinek Mercer

The result of Jaro Winkler (C_j^i)		**Left bigram, right bigram, and trigram**	**Jelinek Mercer Probability**
C_1^1	date	_ date	$P_1'(W^{i-1}) = \lambda P_1(W^{i-1}) + (1-\lambda)P_0(C_j^i) = 0$
		date procesing	$P_2'(W^{i+1}) = \lambda P_2(W^{i-1}) + (1-\lambda)P_0(C_j^i) = 0$
		_ date procesing	$P_3'(W^{i-1}, W^{i+1})$
			$= \lambda P_3(W^{i-1}, W^{i+1}) + (1-\lambda)((P_1(W^{i-1}) + \frac{P_2(W^{i+1})}{2} = 0$
…	…	…	…

5. Calculate a Score The calculation of the score is one of the final stages in the process of detecting and correcting writing errors. This score is computed by combining the results of bigrams and trigrams because high-order n-grams are more sensitive to a context, and low-order n-grams are less sensitive in recognizing a context [16]. This score is calculated using the weighted combination score formula (11).

$$Score(C_j^i) = \lambda_1 P_1(W^{i-1}) + \lambda_2 P_2(W^{i+1}) + \lambda_3 P_3(W^{i-1}, W^{i+1}) \tag{11}$$

The calculation of the score can be seen in Table 8.

Table 8. The Calculation of Scores

The Result of Jaro Winkler (C_j^i)		**score** (C_j^i)
C_1^1	date	$Score(C_j^i) = \lambda_1 P_1(W^{i-1}) + \lambda_2 P_2(W^{i+1}) + \lambda_3 P_3(W^{i-1}, W^{i+1}) = 0,25(0) + 0,25(0) + 0,5(0) = 0$
C_2^1	data	$Score(C_j^i) = \lambda_1 P_1(W^{i-1}) + \lambda_2 P_2(W^{i+1}) + \lambda_3 P_3(W^{i-1}, W^{i+1}) = 0,25(1) + 0,25(0,9) + 0,5(0,45) = 0,7$
C_1^2	process	$Score(C_j^i) = \lambda_1 P_1(W^{i-1}) + \lambda_2 P_2(W^{i+1}) + \lambda_3 P_3(W^{i-1}, W^{i+1}) = 0,25(0) + 0,25(1) + 0,5(0) = 0,25$

The score that has been obtained is then ranked from the largest to the smallest value. The greatest value is the recommendation of the right word.

d. Scoring

The scoring process is used to obtain the value of the respondents answers who have been given corrections for writing errors. This stage uses the Vector Space Model (VSM) method. This method is used to find the similarity of a document with a query that is represented in a vector form [39]. This method uses the calculation of Term Frequency (TF) and Inverse Document Frequency (IDF). After the TF-IDF is found, then the matching document uses cosine similarity.

1. Term Frequency (TF)
   Term Frequency (TF) is a way to get the weight value of the appearance of the term in a document. If the term frequently appears in the document, the TF value will be higher than other terms [40]. The TF is computed using the following (12).

$$tf(t,d) = \frac{f_{t,d}}{\Sigma_{t' \in d} f_{t',d}} \tag{12}$$

2. Inverse Document Frequency (IDF)
   Inverse Document Frequency (IDF) is used to count the occurrence of a term in another document [41]. The IDF is computed using the (13)

$$idfi = log\ log \frac{N}{dfi} \tag{13}$$

3. Calculate a Weight of TF IDF
   This weight calculation is done by multiplying the TF value by the IDF, which can be seen in (14).

$$W_{ij} = tfij.log\ log \frac{N}{dfi} \tag{14}$$

   If the value of $N = dfi$, the weight or W_{ij} will be obtained based on the tfij calculation [17]. Thus, it is necessary to add a value of 1 to the Inverse Document Frequency (IDF), which can be seen in (15).

$$W_{ij} = tfij. \left(log\ log \frac{N}{dfi} + 1 \right) \tag{15}$$

4. Matching Documents
   Matching documents is a way to calculate the level of similarity between documents [40]. Documents for which the similarity is calculated in this research are the answers given by the respondents and the answer keys to the questions. Calculations on this matching document use the cosine similarity method that can be calculated by (16).

$$Sim(q,d) = \cos\cos\theta = \frac{q.d}{|q||d|} = \frac{\Sigma_{k=1}^{t} W_{qk}.W_{dk}}{\sqrt{\Sigma_{k=1}^{t}(W_{qk})^2}\sqrt{\Sigma_{k=1}^{t}(W_{dk})^2}} \tag{16}$$

5. Similarity Value Conversion
   This step is the process of converting the similarity values from the document matching process into general test scores (Human Rate Value). The similarity conversion value range can be seen in Table 9 [16].

Table 9. The Similarity Conversion Value Range

Similarity Calculation Value	Human Rates Value
0.01-0.10	10
0.11-0.20	20
0.21-0.30	30
0.31-0.40	40
0.41-0.50	50
0.51-0.60	60
0.61-0.70	70
0.71-0.80	80
0.81-0.90	90
0.91-1.00	100

3. RESULT AND ANALYSIS

This research was conducted using test data from the results of the questionnaire, which amounted to 115 answers from 23 respondents.3.1.Writing Errors Detection and Correction Test In this test, each respondents answer is given a writing error scenario which consists of a letter overload, letter shortage, letter transposition, and letter replacement scenario, then continued with testing using the system and recorded in the table by giving a value of 1 which means true and 0 means false. This value is then used for accurate calculations. The calculation of accuracy is applied to every answer given to the respondent. If it has been applied in all answers, then the average, minimum, and maximum accuracy will be obtained. The accuracy calculation is as follows:

a. Accuracy of Detection Writing Error
The calculation of writing error detection accuracy is used to determine how accurately the system can detect writing errors by first comparing the results of manual detection with the modeling and then calculating the accuracy (Show Table 10).

Table 10. The Example of Accuracy Detection Calculation

Respondent	Question	Error Scenario	Manual		Correction	System		Check		Accuracy
			F	T		F	T	T	F	
Respondent 1	1	data	0	1	data	0	1	1	0	100,00%
		procesing	1	0	processing	1	0	1	0	
		is	0	1	is	0	1	1	0	
		the	0	1	the	0	1	1	0	
		processing	0	1	processing	0	1	1	0	
		raw	0	1	raw	0	1	1	0	
		data	0	1	data	0	1	1	0	
		into	0	1	into	0	1	1	0	
		information	0	1	information	0	1	1	0	
	Total							9	0	

Based on Table 10, there are scenarios of writing errors in the word processing (procesing). The error scenario of the word procesing has a value of 1 in the false column, which means that the word is included in the category of writing errors. During testing using the system, the word procesing also has a value of 1 in the false column, marked by a word recommendation generated by the system. If the results of manual detection and the detection performed by the system have the same value, then a value of 1 is given in the true check column. If the results are not the same, then a value of 1 is given in the false check column. For example, suppose every word in the sentence has been compared. In that case, the comparison results are added up to calculate the accuracy of writing error detectionthe formula to calculate the accuracy [42] using equation (17).

$$\text{Detection Accuracy} = \frac{\text{true detection}}{\text{true detection + false detection}} \times 100\% \tag{17}$$

b. Accuracy of Correction Writing Error
The calculation of writing error correction accuracy is almost the same as the calculation of writing error detection accuracy. This calculation determines how accurately the system can provide correct recommendations by comparing the results of manual recommendations with the systems recommendations. The accuracy is calculated based on the results of these comparisons (as shown in Table 11).

Table 11. The Example of Accuracy Correlation Calculation

Respondent	Question	Error Scenario	Manual Correction	Manual		System Correction	System		Accuracy
				T	F		T	F	
Respondent 1	1	data	data	1	0	data	1	0	100,00%
		procesing	processing	1	0	processing	1	0	
		is	is	1	0	is	1	0	
		the	the	1	0	the	1	0	
		processing	processing	1	0	processing	1	0	
		raw	raw	1	0	raw	1	0	
		data	data	1	0	data	1	0	
		into	into	1	0	into	1	0	
		information	information	1	0	information	1	0	
	Total						9	0	

Based on Table 11, the word procesing is included in the writing error where the actual writing is processing. If the word is tested using the system, the correct recommendation result is "processing." The correct recommendation is given a value of 1 in the true column, and the incorrect recommendation is given a value of 1 in the false column. The calculation is continued by adding up each column of the systems true and false recommendations, and the accuracy is calculated using the formula (18).

$$\text{Detection Accuracy} = \frac{\text{true correction}}{\text{true correction + false correction}} \times 100\% \tag{18}$$

3.1. Easy Scoring Test with and without Writing Errors

Essay scoring in this research was carried out by comparing the similarity of words between respondents answers and answer keys. The essay scoring test was carried out with two conditions (with and without writing errors). The result of this test is that the essay scoring is strongly influenced by the accuracy of the word recommendations generated by the system. If the recommendations are incorrect, the resulting scoring value will decrease, increase or remain the same.

3.2. Discussion of Results

Based on the research results, the following conclusions can be drawn:

1. Writing Errors Detection and Correction Test
   Based on the test results, the average, minimum, and maximum accuracy was obtained in detecting and correcting writing errors. Based on our model testing, the average accuracy of writing error detection is 83.64%, the worst accuracy is 57.14%, and the best accuracy is 100.00% (Table 10). While the writing error correction results obtained an average accuracy of 78.44%, the worst accuracy was 40.00%, and the best accuracy was 100.00% (Table 11). Testing of Jaro Winkler and N-Gram methods in this research succeeded in detecting and correcting writing errors with four types of writing errors, namely letter overload, letter transposition, letter replacement, and letter deficiency. However, incorrect words are still detected in the writing error detection test. There are words that should not be included in the category of misspelling, but the system has detected a typo by being marked with an inappropriate word recommendation. This happens because there are words that are not in the Kamus Besar Bahasa Indonesia (KBBI), so when the words are checked using Jaro Winkler, they have been given word recommendations that are not appropriate, which can also affect the processing stage with N-Grams. In addition, it is also caused by the influence of the quantity or frequency of occurrence of words in the N-Gram. When processing the text using Jaro Winkler, the appropriate word recommendation has been given, but when processed with N-Gram, the same word is not found in the N-Gram corpus, or the frequency of occurrence of the word is less than that of other words. Thus, it can be said that the process of detecting and correcting writing errors using Jaro Winkler and N-Grams depends on the availability or completeness of words in the Kamus Besar Bahasa Indonesia (KBBI) and the N-Gram corpus to get good detection and correction accuracy.
2. Comparison of essay assessment test results with writing errors and without writing errors
   The results of the comparison between essay scoring with writing errors and without writing errors are influenced by whether the correct or not correct recommendation word was given in the previous process. It is because the essay scoring in this

research was only based on the similarity of words contained in the respondents answers with the answer key (the word's meaning was not considered). The correctness of recommendations and the correctness of the choice of words in the answer and answer key can affect the scoring. As a result, it is still difficult to achieve a scoring of an essay from a system similar to that of an expert (teacher). Thus, the next research will be increased by the Latent Semantic Analysis method or NLP [43, 44].

4. CONCLUSION

The method proposed in this paper detects writing errors with an average accuracy of 83.64%, minimum accuracy of 57.14%, and maximum accuracy of 100.00%. While the writing error correction test obtained an average accuracy of 78.44%, a minimum accuracy of 40.00%, and the maximum accuracy of 100.00%. The value of the accuracy of detection and correction of writing errors in this research is very dependent on the number of completeness or availability of words in the word dictionary used, namely the Kamus Besar Bahasa Indonesia (KBBI) and the n-gram corpus. The results of the application of detection and correction of writing errors in the essay scoring system are still not good because of the influence of several word recommendations in the previous process, which are still not correct. In addition, the accuracy of word choice in answers and answer keys can affect the value of essay assessment, which is only based on word similarity. Therefore, further research is recommended to complete the word corpus n-gram to overcome the absence of the desired word or the lack of frequency of occurrence of the word corpus n-gram. In addition, methods to overcome essay assessments based on word meanings, such as the Latent Semantic Analysis method or NLP.

REFERENCES

[1] G. Giray, "An Assessment of Student Satisfaction with E-Learning: An Empirical Study with Computer and Software Engineering Undergraduate Students in Turkey Under Pandemic Conditions," *Education and Information Technologies*, vol. 26, no. 6, pp. 6651–6673, nov 2021.

[2] D. Ramesh and S. K. Sanampudi, "An Automated Essay Scoring Systems: A Systematic Literature Review," *Artificial Intelligence Review*, vol. 55, no. 3, pp. 2495–2527, mar 2022.

[3] R. Fitri and A. N. Asyikin, "Aplikasi Penilaian Ujian Essay Otomatis Menggunakan Metode Cosine Similarity," *Jurnal Poros Teknik*, vol. 7, no. 2, pp. 88–94, 2015.

[4] M. A. Hussein, H. Hassan, and M. Nassef, "Automated Language Essay Scoring Systems: A Literature Review," *PeerJ Computer Science*, vol. 5, no. August, pp. 1–28, aug 2019.

[5] N. Süzen, A. N. Gorban, J. Levesley, and E. M. Mirkes, "Automatic Short Answer Grading and Feedback using Text Mining Methods," in *Procedia Computer Science*, 2020, pp. 726–743.

[6] E. Hartati and M. Mardiana, "Evaluasi Penerapan Computer Based Test (CBT) sebagai Upaya Perbaikan Sistem pada Ujian Nasional untuk Sekolah Terpencil di Sumatera Selatan," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 18, no. 1, pp. 58–64, nov 2018.

[7] S. Link, M. Mehrzad, and M. Rahimi, "Impact of Automated Writing Evaluation on Teacher Feedback, Student Revision, and Writing Improvement," *Computer Assisted Language Learning*, vol. 35, no. 4, pp. 605–634, may 2022.

[8] M. Zhu, O. L. Liu, and H.-S. Lee, "The Effect of Automated Feedback on Revision Behavior and Learning Gains in Formative Assessment of Scientific Argument Writing," *Computers & Education*, vol. 143, no. January, pp. 1–43, jan 2020.

[9] E. Lindgren, A. Westum, H. Outakoski, and K. P. Sullivan, "Revising at the Leading Edge: Shaping Ideas or Clearing up Noise," in *Observing Writing*. BRILL, jan 2019, pp. 346–365.

[10] S. J. Putra, T. Mantoro, and M. N. Gunawan, "Text Mining for Indonesian translation of the Quran: A Systematic Review," in *2017 International Conference on Computing, Engineering, and Design (ICCED)*. IEEE, nov 2017, pp. 1–5.

[11] I. Ganguli, R. S. Bhowmick, and J. Sil, "Deep Insights of Erroneous BengaliEnglish Code-Mixed Bilingual Language," *IETE Journal of Research*, pp. 1–12, jun 2021.

[12] D. Deksne, "Bidirectional LSTM Tagger for Latvian Grammatical Error Detection," in *Text, Speech, and Dialogue. TSD 2019. Lecture Notes in Computer Science*, 2019, vol. 11697, pp. 58–68.

[13] W. Wei and Y. K. Cao, "Written Corrective Feedback Strategies Employed by University English Lecturers: A Teacher Cognition Perspective," *SAGE Open*, vol. 10, no. 3, pp. 1–12, jul 2020.

[14] J. L. Hernández, F. M. Molina, and Á. Almela, "Analysis of Context-Dependent Errors in the Medical Domain in Spanish: A Corpus-Based Study," *SAGE Open*, vol. 13, no. 1, pp. 1–11, jan 2023.

[15] J. Zhang, C. Wang, A. Muthu, and V. Varatharaju, "Computer Multimedia Assisted Language and Literature Teaching using Heuristic Hidden Markov Model and Statistical Language Model," *Computers & Electrical Engineering*, vol. 98, no. March, p. 107715, mar 2022.

[16] P. Samanta and B. B. Chaudhuri, "A Simple Real-Word Error Detection and Correction using Local Word Bigram and Trigram," in *Proceedings of the 25th Conference on Computational Linguistics and Speech Processing (ROCLING 2013)*. Kaohsiung: The Association for Computational Linguistics and Chinese Language Processing (ACLCLP), oct 2013, pp. 211–220.

[17] D. Sudigyo, A. A. Hidayat, R. Nirwantono, R. Rahutomo, J. P. Trinugroho, and B. Pardamean, "Literature study of stunting supplementation in Indonesian utilizing text mining approach," in *Procedia Computer Science*, 2023, pp. 722–729.

[18] A. Musyafa, Y. Gao, A. Solyman, C. Wu, and S. Khan, "Automatic Correction of Indonesian Grammatical Errors Based on Transformer," *Applied Sciences*, vol. 12, no. 20, pp. 1–17, oct 2022.

[19] A. Bannayeva and M. Aslanov, "Development of the N-gram Model for Azerbaijani Language," in *2020 IEEE 14th International Conference on Application of Information and Communication Technologies (AICT)*. IEEE, oct 2020, pp. 1–5.

[20] C. Lai, "Fast Retrieval Algorithm of English Sentences Based on Artificial Intelligence Machine Translation," in *In: Atiquzzaman, M., Yen, N., Xu, Z. (eds) 2021 International Conference on Big Data Analytics for Cyber-Physical System in Smart City. BDCPS 2021. Lecture Notes on Data Engineering and Communications Technologies*, 2022, pp. 1057–1065.

[21] F. Friendly, "JaroWinkler Distance Improvement For Approximate String Search Using Indexing Data For Multiuser Application," *Journal of Physics: Conference Series*, vol. 1361, no. 1, pp. 1–7, nov 2019.

[22] Y. Rochmawati and R. Kusumaningrum, "Studi Perbandingan Algoritma Pencarian String dalam Metode Approximate String Matching untuk Identifikasi Kesalahan Pengetikan Teks," *Jurnal Buana Informatika*, vol. 7, no. 2, pp. 125–134, jan 2016.

[23] P. Pitchandi and M. Balakrishnan, "Document Clustering Analysis with Aid of Adaptive Jaro Winkler with Jellyfish Search Clustering Algorithm," *Advances in Engineering Software*, vol. 175, no. January, p. 103322, jan 2023.

[24] D. A. Anggoro and I. Nurfadilah, "Active Verb Spell Checking Mem- + P in Indonesian Language Using the Jaro-Winkler Distance Algorithm," *Iraqi Journal of Science*, vol. 63, no. 4, pp. 1811–1822, apr 2022.

[25] F. Shole, "Perbandingan Metode Smoothing untuk Deteksi dan Koreksi Kesalahan Kata Dalam Teks Berbahasa Indonesia," *Unikom Repository, Diploma thesis, Universitas Komputer Indonesia.*, vol. 63, no. 4, pp. 1811–1822, 2018.

[26] I. Ahamed, M. Jahan, Z. Tasnim, T. Karim, S. M. S. Reza, and D. A. Hossain, "Spell Corrector for Bangla Language using Norvig's Algorithm and Jaro-Winkler Distance," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 4, pp. 1997–2005, aug 2021.

[27] A. M. Fanani and S. Suyanto, "Syllabification Model of Indonesian Language Named-Entity using Syntactic n-Gram," in *Procedia Computer Science*, 2021, pp. 721–727.

[28] H. Jayadianti, W. Kaswidjanti, A. T. Utomo, S. Saifullah, F. A. Dwiyanto, and R. Drezewski, "Sentiment Analysis of Indonesian reviews using Fine-Tuning IndoBERT and R-CNN," *ILKOM Jurnal Ilmiah*, vol. 14, no. 3, pp. 348–354, 2022.

[29] P. S. Br Ginting, B. Irawan, and C. Setianingsih, "Hate Speech Detection on Twitter using Multinomial Logistic Regression Classification Method," in *2019 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)*. IEEE, nov 2019, pp. 105–111.

[30] Y. Fauziah, S. Saifullah, and A. S. Aribowo, "Design Text Mining for Anxiety Detection using Machine Learning Based-on Social Media Data during COVID-19 Pandemic," in *Proceeding of LPPM UPN Veteran Yogyakarta Conference Series 2020Engineering and Science Series*, 2020, pp. 253–261.

[31] S. Saifullah, Y. Fauziyah, and A. S. Aribowo, "Comparison of Machine Learning for Sentiment Analysis in Detecting Anxiety Based on Social Media Data," *Jurnal Informatika*, vol. 15, no. 1, pp. 45–55, feb 2021.

[32] V. C. M., R. Rudy, and D. S. Naga, "Fast and Accurate Spelling Correction Using Trie and Damerau-levenshtein Distance Bigram," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 16, no. 2, pp. 827–833, apr 2018.

[33] A. Indriani, M. Muhammad, S. Suprianto, and H. Hadriansa, "Implementasi Jaccard Index dan N-Gram pada Rekayasa Aplikasi Koreksi Kata Berbahasa Indonesia," *Sebatik*, vol. 22, no. 2, pp. 95–101, dec 2018.

[34] A. A. P. Ratna, R. Sanjaya, T. Wirianata, and P. Dewi Purnamasari, "Word Level Auto-Correction for Latent Semantic Analysis Based Essay Grading System," in *2017 15th International Conference on Quality in Research (QiR) : International Symposium on Electrical and Computer Engineering*. IEEE, jul 2017, pp. 235–240.

[35] I. E. Agbehadji, H. Yang, S. Fong, and R. Millham, "The Comparative Analysis of Smith-Waterman Algorithm with Jaro-Winkler Algorithm for the Detection of Duplicate Health Related Records," in *2018 International Conference on Advances in Big Data, Computing and Data Communication Systems (icABCD)*. IEEE, aug 2018, pp. 1–10.

[36] T. Tinaliah and T. Elizabeth, "Perbandingan Hasil Deteksi Plagiarisme Dokumen dengan Metode Jaro-Winkler Distance dan Metode Latent Semantic Analysis," *Jurnal Teknologi dan Sistem Komputer*, vol. 6, no. 1, pp. 7–12, jan 2018.

[37] Y. Yulianingsih, "Implementasi Algoritma Jaro-Winkler dan Levenstein Distance dalam Pencarian Data pada Database," *STRING (Satuan Tulisan Riset dan Inovasi Teknologi)*, vol. 2, no. 1, pp. 18–27, aug 2017.

[38] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd ed., dec 2021.

[39] C. Slamet, A. R. Atmadja, D. S. Maylawati, R. S. Lestari, W. Darmalaksana, and M. A. Ramdhani, "Automated Text Summarization for Indonesian Article Using Vector Space Model," in *IOP Conference Series: Materials Science and Engineering*, jan 2018, pp. 1–6.

[40] M. E. Sulistyo, R. Saptono, and A. Asshidiq, "Penilaian Ujian Bertipe Essay Menggunakan Metode Text Similarity," *Telematika*, vol. 12, no. 2, pp. 146–158, jul 2015.

[41] S. Saifullah, N. H. Cahyana, Y. Fauziah, A. S. Aribowo, F. A. Dwiyanto, and R. Drezewski, "Text Annotation Automation for Hate Speech Detection using SVM-classifier Based on Feature Extraction," in *International Conference on Advanced Research in Engineering and Technology*, 2022.

[42] T. Tundo and S. Saifullah, "Fuzzy Inference System Mamdani dalam Prediksi Produksi Kain Tenun Menggunakan Rule Berdasarkan Random Tree," *Jurnal Teknologi Informasi dan Ilmu Komputer*, vol. 9, no. 3, pp. 443–451, jun 2022.

[43] M. R. Pratama and M. Yunus, "Sistem Deteksi Struktur Kalimat Bahasa Arab Menggunakan Algoritma Light Stemming," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 19, no. 1, pp. 109–118, nov 2019.

[44] N. H. Cahyana, S. Saifullah, Y. Fauziah, A. S. Aribowo, and R. Drezewski, "Semi-Supervised Text Annotation for Hate Speech Detection using K-Nearest Neighbors and Term Frequency-Inverse Document Frequency," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 10, pp. 147–151, 2022.

[This page intentionally left blank.]