ISSN: 0067-2904

Automatic Short Answer Grading System Based on Semantic Networks and Support Vector Machine

Naamah Hussein Hameed*, Ahmed T. Sadiq
Department of computer science, University of Technology, Baghdad, Iraq

Received: 5/11/2022 Accepted: 18/1/2023 Published: 30\11\2023

ABSTRACT

In education, exams are used to asses students' acquired knowledge; however, the manual assessment of exams consumes a lot of teachers' time and effort. In addition, educational institutions recently leaned toward distance education and e-learning due the Coronavirus pandemic. Thus, they needed to conduct exams electronically, which requires an automated assessment system. Although it is easy to develop an automated assessment system for objective questions. However, subjective questions require answers comprised of free text and are harder to automatically assess since grading them needs to semantically compare the students' answers with the correct ones. In this paper, we present an automatic short answer grading method by measuring the semantic similarity between the students answer and the correct answer. A semantic network was built to represent the relationship between the words of the two texts to calculate semantic similarity. Representing a text as a semantic network is the best knowledge representation that comes closest to human comprehension of the text, where the semantic network reflects the sentence's semantic, syntactical, and structural knowledge. Several features were extracted from the semantic network and used as input to train the support vector machine (SVM) model to predict the degree of the targeted semantic similarity. The proposed method was tested on the Mohler dataset that is publicly available online. The obtained results were evaluated and reported in terms of Pearson correlation and root mean squared error (RMSE) where it achieved 0.63 and 0.83 respectively. The proposed method outperformed all previous methods on the used dataset.

Keywords: Automatic Short Answer Grading, E-learning, Semantic Similarity, Semantic Network, Support Vector Machine.

نظام آلي لتقييم الإجابات القصيرة استنادًا إلى الشبكة الدلالية وآلة المتجه الداعم

نعمة حسين حميد*، احمد طارق صادق
قسم علوم الحاسوب، الجامعة التكنولوجية، بغداد، العراق

الخلاصة

في نظام التعليم الخاص بالطلبة يتم تقييم الطلبة بشكل حاسم من خلال الختبارات، ولكن التقييم اليدوي للامتحانات يستهلك الكثير من وقت المعلمين وجهدهم. في الآونة الأخيرة، اتجهت المؤسسات التعليمية نحو التعليم عن بعد والتعلم الإلكتروني خصوصا مع انتشار جائحة فيروس كورونا. وبالتالي الحاجة إلى إجراء الاختبارات إلكترونيًا، الأمر الذي يتطلب نظام تقييم آلي. على الرغم من أنه من السهل تطوير نظام تقييم آلي للأسئلة الموضوعية، إلا أن الأسئلة المقالية تتطلب إجابات تتكون من نص حر ويصعب تقييمها تلقائيًا نظرًا لأن تقييمها

*Email: aa23329@gmail.com

يحتاج إلى مقارنة لغوية لإجابات الطلاب بالإجابة الصحيحة. في هذا البحث، نقدم طريقة آلية لتقييم الإجابات القصيرة عن طريق قياس التشابه الدلالي (semantic similarity) بين إجابات الطلاب والإجابة الصحيحة. تم بناء شبكة دلالية (semantic network) لتمثيل العلاقة بين كلمات النصين لحساب التشابه الدلالي. يعتبر تمثيل النص كشبكة دلالية أفضل تمثيل معرفي يقترب من الفهم البشري للنصوص، حيث تعكس الشبكة الدلالية المعرفة الدلالية والنحوية والهيكلية للجملة. تم استخراج العديد من الميزات من الشبكة الدلالية واستخدامها كمدخل لتدريب نموذج آلة متجه الدعم (SVM) للتنبؤ بدرجة التشابه الدلالي بين نصين. تم اختبار الطريقة المقترحة على مجموعة بيانات Mohler المتاحة مجانا على الإنترنت. تم تقييم النتائج التي تم الحصول عليها من حيث معامل ارتباط بيرسون (Pearson correlation) والخطأ التربيعي لمتوسط الجذر (RMSE). حيث حققت نتائج 0.63 و0.83 على التوالي، متفوقة على جميع الطرق السابقة مع نفس مجموعة البيانات.

INTRODUCTION

Student assessment exams are a routine process in the education field, and the manual assessment of these exams costs considerable time and effort for assessors. The advent of the global Coronavirus pandemic and the collective institutional shift toward distance education highlighted the necessity for an automated electronic exams assessment system. The exam's questions are often classified into objective questions; for example, (multiple-choice, true/false, fill-in-the-blanks) and subjective questions that require short or long answers [1][2].

Building an automated evaluation system for objective questions can be easily done, while the same cannot be said for subjective questions. Automating such a system requires textual analysis and comprehending the answers' semantic meaning. For instance, a student may answer questions such as "what is the definition of a computer?" as "a programmable machine that reacts to specific commands and executes a set of instructions known as a program" whereas another student would define it as "a programmable electronic device that performs arithmetic operations and logical functions". Both answers can be considered correct despite their different description.

The automatic short answer grading system provides an immediate grade to a student's answer by comparing it to one or more optimal answers. This differs from the related tasks of paraphrase detection [3] and textual entailment that are typical requirements in student answer grading to deliver a score based on a certain scale rather than a simple "yes/no" decision [4].

Evaluating a student's answer score can be deduced by measuring the level of semantic similarity between the student's and the reference answer. In this paper, the authors employ comparing semantic similarity to create an automatic short answer grading method. The calculation of the semantic similarity is based on the semantic network and Support Vector Machine (SVM). The semantic network is a method of knowledge representation as the nodes symbolize words or concepts, whereas the edges symbolize the binary relationship that connects two nodes.

Furthermore, a semantic network is built to denote the relationship between the words of the compared texts. The words of both the student's and the reference's answers represent the nodes, whereas the edges represent the similarity level between them. Several features are extracted from the built semantic network to be used as input to train the SVM model for regression to predict the degree of similarity between both answers. The major contributions of this paper are as follows:

- Develop a fast and easy-to-implement model that reduces training and implementation time that relies on fewer features.
- Reduce the error rate in comparison to the traditional method of human evaluation (the difference between the score given by the system and that of the human evaluator).

The rest of the paper is structured as follows: Section 2 briefly reviews similar works, while Section 3 presents the theoretical background. Section 4 describes the proposed method. Sections 5 and 6 address the implementation details and the results obtained, respectively. Finally, Section 7 summarizes the study and concludes it.

RELATED WORK

Several methods have been introduced by researchers for evaluating students' short answers[5]. The early methods relied on matching patterns, requiring expert intervention to be extracted. Along with the evolution of natural language processing techniques, automated evaluation of students' answers by calculating semantic similarity has also gained popularity [6]. Mohler and Mihalcea [4] explored several techniques for automatic short answer grading (ASAG) tasks. They compared eight knowledge-based text similarity measures and two corpus-based measures (latent semantic analysis (LSA) and explicit semantic analysis (ESA)) that are trained on domain-specific and generic corpora. To improve their system performance, they integrated automatic feedback from the student answers where the top-ranked student's answered words were added to the right answer vocabulary. They created a dataset from computer science exams to evaluate their methods for each question, then a typical answer was provided. The achieved Pearson correlation between the system score and the human-annotated score was 0.6735 with LSA embedded in Wikipedia computer science-related articles.

Recently, the use of machine learning algorithms for ASAG systems has also become popular. Mohler et al.[7] Suggested a machine learning approach for short answer question grading. They combined several semantic similarity measures with a dependency graph alignment feature to introduce a features vector. The features vector is provided to the SVM model to predicate student answer scores. The single feature vector contained 30 features, eight features extracted from the dependency graph alignment, and 11 features from the eight knowledge-based measures in addition to the LSA, which was introduced in [4]. In addition, there is a term frequency–inverse document frequency (tf*idf) vector that features both with and without question reduction. To evaluate their methods, they created a dataset of computer science exams with answers submitted by a class of undergraduate students. Their approach achieved a 0.978 RMSE. Thus, it is concluded that the use of many features may lead to system efficiency degradation. Additionally, the obtained results were not encouraging.

In a similar approach presented by Sultan et al.[8], that suggested a short text similarity-based short answer grading method. They measured the semantic closeness between the learner's answer and the reference answer by training a supervised model. Multiple features were extracted as alignment features that measure the percentage of words in the two similar texts. The semantic vector similarity feature uses word embedding vectors to measure the semantic similarity of two texts and both previous features that were recomputed after question demoting. To distinguish between keywords and less important words, weight was given to each word in the learner's response and reference answer. The final feature was the ratio between the learner's and the model's answer regarding the number of words. They tested their method with the Mohler dataset, where 0.887 RMSE was achieved. This however,

relied on dependency parsing, which is computationally expensive and slow to execute.

In the same context, Saha et al. [9] combined sentence-level features with token-level ones. For sentence-level features, they utilized InferSent [10] to attain sentence embedding for the question, the model's answer, and the learner's answer. Word overlap, Question Natures, and Histogram of Partial Similarity on the part of speech tags were their token level features. The combined features were fed to a machine learning classification or regression model depending on the dataset's nature. They tested their approach on three datasets: Large Scale Industry Dataset with 0.6636 accuracy results, SemEval-2013 [11] with 0.6444 , and the Mohler dataset with 0.902 RMSE results.

Another approach was suggested by Condor [12] where they proposed BERT [13] (Bidirectional Encoder Representations from Transformers) based ASAG system as a tool to support human assessors. They utilized a lightweight version of the BERT model, called the "Bert-base" for the assessment task. They tested their model on the DT-Grade dataset, which consists of 1000 student responses. Each answer was evaluated as either "correct", "correct but incomplete", "contradictory", and "incorrect". " Correct" answers were considered correct while the rest of the categories were considered incorrect to convert it into a binary classification problem. Their method achieved an accuracy of 0.76.

Lubis et al. [14] proposed semantic similarity based on word embedding for the ASAG system. They trained the word2vec model on a full Wikipedia dump in Indonesia to obtain a word embedding vector. The students' and model answer were converted to sentence vectors by computing the average of their words vector, and the semantic similarity was then computed as cosine similarity between their vectors. They tested their method on a 224student response from the computer network engineering class dataset where it achieved a Pearson correlation of 0.7.

It is worth noting in the current literature that:

- The methods that rely on dependency parsers to extract the learners/reference answers are structural and semantical features, which could be slow and computationally complex, making them unsuitable for real-world systems.
- Relying on many features burdens the machine learning algorithm and makes it computationally slow and complex.
- The achieved results are still insufficient. Hence, the existing methods are not qualified to entirely replace the manual assessment.

In contrast, this paper proposes a method that relies on fewer features (only four) to develop a fast and computationally low-complex process to achieve better performance.

THEORETICAL BACKGROUND

The evaluation of short answers is done by calculating the semantic similarity between the student's answer and the reference answer provided by the teacher. It is important to realize similarities between words in text since words can be similar in two aspects: lexical and semantic. Words are lexically similar if their character sequences are similar, whereas semantical similarity occurs if they are referring to the same meaning, employed in the same way/context, or one word is a synonym of another. String-Based methods can be used to compute lexical similarity and use their algorithms to ensure that a string comparison metric is utilized to compare the similarity of distinct character sequences. Corpus-Based or Knowledge-Based algorithms can also be used to calculate semantic similarity because the algorithms use information acquired in a huge corpus to compute semantic similarity between

words. This differs from the previous Knowledge-Based algorithms that use the knowledge obtained from semantic networks to determine the semantic closeness of words [5],[6],[7].

3.1 Semantic network

The semantic network is a method of representing knowledge where the nodes symbolize objects or concepts, and the edges symbolize the binary relationship that connects two nodes. The network representation provides a pictorial representation of knowledge objects, their attributes, and the relationship between them [18]. Representing a text as a semantic network is the best representation of knowledge that comes closest to the human mind's understanding of texts. The semantic network reflects the semantic, syntactical, and structural knowledge of the sentence. The relationship between the nodes can be either "is a"," a kind of"," a part of," and so on. In the proposed method, instead of a descriptive relationship between words, the relationship between network nodes is a numeric value that represents the degree of similarity between them. This study uses WordNet [19] as a knowledge-based source and GloVe [20] as a corpus-based source to measure the similarity score between words. Two sources are used to avoid any of the words lacking in one of the sources. Figure 1 is an example of a semantic network.

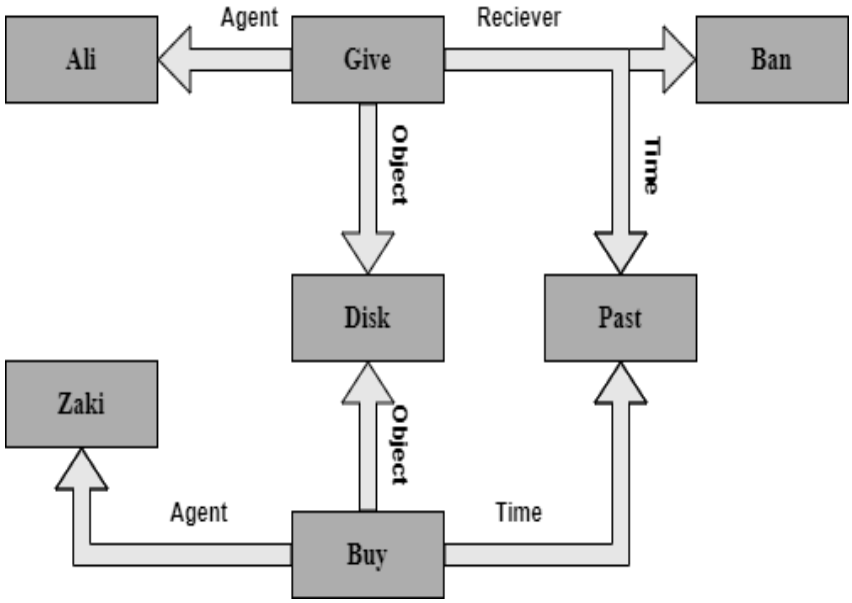

Figure 1: The semantic network of the sentence "Ali gave Ban a disk which Zakie has bought".

3.2 Word to Word Similarity

In this section, the sources used to calculate the semantic similarity between words are described.

3.2.1 WordNet

WordNet is an example of a semantic network where words or concepts are linked by synonymy or meronymy [21]. It is a lexical database with over 100,000 English words that are commonly used for knowledge-based semantic similarity approaches [22]. The lexicon is divided into nouns, verbs, adjectives, and adverbs. The words' clusters are grouped to form synsets or synonym sets. Here, all the terms have the same meaning, hence, are interchangeable in certain syntax. The definitions of the words and pointers to the other similar synsets are included in a synset's knowledge [23].

Synsets are also organized and structured in a tree hierarchy with many specialized bottom terms and a few general top phrases. Following trails of superordinate terms in "is a" or "is a sort of" (ISA) relations connects the lexical hierarchy. Each word rises the lexical tree until the two claiming paths meet to form a path between them. A subsumer is the synset at the intersection of the two claiming paths where a path connecting the two words is then identified through the subsumer. The path length is yielded by calculating synset links of the entire path between those words [24].

Counting the levels from the subsumer to the top of the lexical hierarchy yields the depth of the subsumer. If a word is polysemous (has numerous meanings), there may be multiple pathways between the two terms [25]. In WordNet, numerous approaches have been developed for identifying semantic similarities between words and concepts categorized as either Path-based, information content (IC)-based, gloss-based, feature-based, and hybrid measures. The proposed method uses Wu and Palmer's [24] measure to compute the similarity between words, which is expressed as

$$\text{Sim}(x1, x2) = 2k / (a1+a2+2k) \quad (1)$$

Where a1 and a2 denote the number of links from word1 (*x1)* and word2 (*x2)*, respectively, while the deepest common subsumer *x* and *k* denote the number of links from x to the root of the taxonomy [22].

3.2.2 GloVe

Word embedding is the process of representing words as a vector that maintains the basic linguistic link between words. These vectors are computed by a variety of methods, including neural networks, word co-occurrence matrices, and representations based on the context of the word [26]. Some of the commonly used pre-trained word embedding sources are GloVe [20], word2vec [19], fast text [27], and BERT[13].

GloVe was developed at Stanford University to employ a worldwide word co-occurrence matrix according to the underlying corpus. It calculates similarity since words that are similar to each other commonly occur together. A single run across the huge underlying corpus is used to populate the co-occurrence matrix with occurrence values. The GloVe model was trained on five corpora, a majority of which were Wikipedia dumps. Words were chosen within a given context window for constructing vectors because words further away are less similar to the contextual word. The GloVe loss function reduces the least square distance between the co-occurrence values in the context window and the world-wide co-occurrence values. The vectors were enhanced to generate contextual word vectors to discriminate words according to the context [22].

METHODOLOGY

The proposed method for the ASAG task is based on assessing the semantic similarity between the learner's and the reference answer by building a semantic network. This network represents the relationship between the elements of the two texts. Then it extracts multiple features from that network to use it as input for the SVM model for regression. Figure 2 shows the process of finding semantic similarities between the two texts. The method starts with text preprocessing (tokenization, cleaning and normalization, part of speech tagging, stop word removal and lemmatization), where the input text is converted into an analyzable and processable clean format. Next, the semantic network is built, which represents the binary relationship between the words of the two texts.

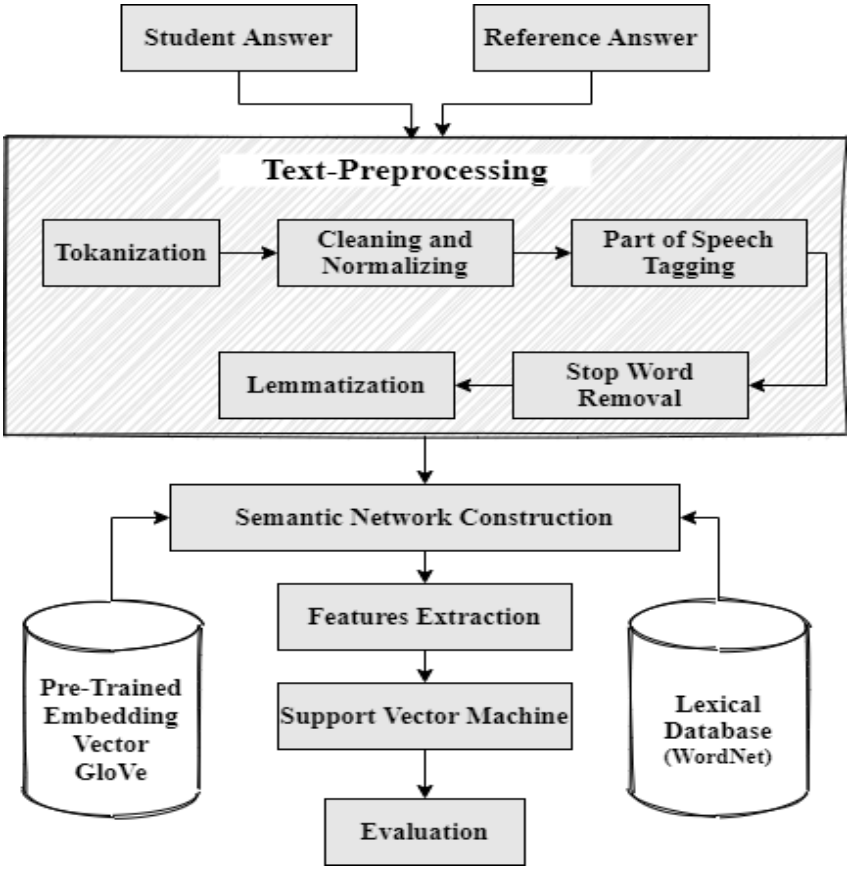

Figure 2: General framework method

The word-to-word semantic similarity is found through the lexical database (WordNet) and the pre-trained embedding vectors (GloVe). Then, a set of features are extracted from the built semantic network. Finally, the extracted features vector is fed to the SVM model to predict the score of semantic similarity between the learner's answer and the reference answer.

4.1 Preprocessing Step

Text preprocessing is a necessary step to convert the text into an analyzable and processable form.

- **Tokenization**: The procedure of dividing the text into smaller components known as "tokens" [28].
- **Cleaning and normalizing:** The original text often comes with some unwanted additions that do not affect the semantic meaning of the text. The cleaning process removes these additions such as duplicated whitespaces, special characters, HyperText Markup Language (HTML) tags, punctuation marks, Uniform Resource Locators links (URLs). After cleaning, the text is converted to lowercase [29].
- **Part of speech tagging:** The procedure of giving a part-of-speech tag to each word in the text based on its meaning and context. Tagging is a disambiguation task because ambiguous words have more than one possible part of speech, and the goal is to find the correct tag for the situation [30].
- **Stop word removal:** Stop words are a list of high-frequency words like pronouns (they, we, you), conjunctions (for, and, while). They have less impact on the semantic meaning of texts, hence deleting these is a suitable option in many natural language processing

(NLP) applications [31].

- **Lemmatization:** The procedure of returning a word to its root form. For example, 'run', 'ran', and 'running' are all conjugations of the verb 'run'. In semantic similarity measurement applications, lemmatization is preferred versus stemming (the process of reducing a word to its base or root form) as it avoids the stemming overgeneralization and considers the part of speech tag of the word; for instance, the word 'running' that has a noun part of speech tag remains the same while 'running' with verb part of speech tag reduced to 'run' [32].

Algorithm 1 show the preprocessing steps.

Algorithm 1: Text preprocessing

Input: raw text (reference answer, student answer)
Output: clean text
Begin:
Pm = punctuation marks list
Sc = special characters' list
Sw =stop word list
Tokens = divide the raw text into tokens
For each token in tokens:
    If token included in Pm or Sc
        Remove token
  Remove (white space, HTML tags, URL links)
    End for
For each token in tokens:
    get the PoS tag of token
    reduce token to its lemma
    remove token if it is included in Sw
    end for
Return Tokens
End

PoS stands for part of speech
Lemma refers to the root form of word
URL stands for Uniform Resource Locator
HTML stands for Hyper Text Markup Language

4.2 Building the Semantic network

After text processing, the network is built to represent the relationship between the two input texts. Each word of the two input texts represents a node in this network, while the score of semantic similarity between the words represents the edges. To build this network, the semantic similarity between each node from the first text and the nodes from the second text is calculated. The highest value that links the node with the nodes of the second text denotes the weight of this node.

The node-to-node similarity scores are identified in two cases:

1) **Case 1:** node A and node B have the same word; hence the relation (weight) assigned is 1.
2) **Case 2**: node A and node B are not the same, and the external sources are used to compute the similarity between the two nodes. Wu & Palmer similarity with WordNet and cosine similarity of Glove word embedding vector of two nodes are calculated. The relation between node A and node B is assigned to the highest score of those two

similarities. The entire process is shown in Algorithm 2.

Algorithm 2: semantic network construction

Input**:** two lists of words of preprocessed texts
Output: semantic network with nodes weights
Begin:
Node_list_1= get all words of preprocessed optimal answer
Node_list_2= get all words of preprocessed student answer
Semantic net= { }
Nodes_weight_1= { }
Nodes_weight_2= { }
For each node1 in Node_list_1:
Weight1 = [ ]
For each node2 in Node_list_2:
If node1=node2:
Add 1 to Weights1
Semantic net [node1, node2] =1
Else:
Add WordNet similarity (node1, node2) to Weights1
Add glove similarity (node1, node2) to Weight1
Semantic net [node1, node2] =max (WordNet similarity, glove similarity)
End for
Nodes_weight_1[node1] = max(Weight)
End for
For node2 in Node_list_2:
Weight2 = [ ]
For node1 in Node_list_1:
If node2=node1:
Add 1 to Weight2
Semantic net [node2, node1] =1
Else:
Add WordNet similarity (node2, node1) to Weights2
Add glove similarity (node2, node1) to Weight2
Semantic net [node2, node1] =max (WordNet similarity, glove similarity)
End for
Nodes_weight_2 [node2] = max (Weights2)
End for
Return (Nodes_weight_1, Nodes_weight_2)
End

4.3 Features extraction

With the semantic network constructed, both texts (reference and student's answer) have a list of nodes with their associated weights. From those lists, a set of features are extracted:
Let S1 be the sum of the reference answer nodes' weights.
Let S2 be the sum of the student answer nodes' weights.
Let N1 be the number of the reference answer nodes.
Let N2 be the number of the student answer nodes.

Reference Answer Semantic Weight (RASW): The first feature represents the degree of similarity of the reference's answer with the student's answer. It is calculated by summing the node's weight with words that appeared within the reference's answer (S1) divided by the number of their nodes (N1), it is calculated as:

$$RASW = S1 / N1 \quad (2)$$

Student Answer Semantic Weight (SASW): The second feature is calculated by summing the node's weight with words that appeared within the student's answer (S2) divided by the number of their nodes (N2), it is calculated as:

$$SASW = S2 / N2 \quad (3)$$

Word overlapping WO: Here, the word overlap is measured between the two texts by considering only the nodes with a weight of 1. It is calculated by the node's number with a weight of 1 (A) divided by the total number of nodes (N1+N2), it is expressed as:

$$WO = A / (N1+N2) \quad (4)$$

Where A denotes the number of nodes in the reference answer and the student's answer that have a weight value of 1.

Length ratio LR: In the last feature, the ratio in the length of the student's answer and the correct answer is considered. It is calculated as:

$$LR = 1 - (|N1-N2|) / (N1+N2) \quad (5)$$

Algorithm 3 illustrates the process of feature extraction.

Algorithm 3: features extraction

```
Input: two lists of Nodes weights
Output: features vector
Begin:
Nodes_weight_1 = get nodes weights of optimal answer
Nodes_weight_2 = get nodes weights of student answer
Features_vector = [ ]
Weight_sum1 = 0
Weight_sum2 = 0
N1 = 0
N2 = 0
T = 0
For each weight1 in Nodes_weight_1:
        Weight_sum1 = Weight_sum1 + weight1
        If weight1 = 1
                T = T+1
For each weight2 in Nodes_weight_2:
        Weight_sum2 = Weight_sum2 + weight2
        If weight2 = 1
                T = T+1
F1 = Weight_sum1 / N1
F2 = Weight_sum2 / N2
F3 = T / (N1+N2)
F4 = 1-absolute (N1-N2) / (N1+N2)
Features_vector = [F1,F2,F3,F4]
Return (Features_vector)
```

4.4 Support vector machine (SVM)

SVM is a powerful supervised machine learning algorithm used for both classification and regression tasks. For the former, SVM is referred to as support vector classification (SVC), while for the latter, it is referred to as support vector regression (SVR). SVM finds a hyperplane (in the case of two-dimensional space), or a line that divides the labeled data in class levels' space for the supplied sets of vectors in multi-dimensional space. This hyperplane is also known as the greatest margin hyperplane (GMH). With the use of a slack variable, SVM can cope with both linearly and non-linearly separable data. However, for nonlinearly separable data, the Kernel SVM model is often favored. The optimum plane in space to separate the vectors is the greatest margin hyperplane. The sets of vectors from each class closest to the maximum margin hyperplane are called "support vectors". Each class has at least one support vector, despite having more than one. SVM can find GMH even when only the support vectors are known and can handle datasets with many features. Support vector identification depends on vector geometry and requires complex arithmetic [33].

Support vector machines (SVMs) can be used for both regression and classification tasks, depending on the nature of the data and the goals of the task. In regression tasks, the objective is to predict a continuous output value, such as a numeric score or a real-valued quantity. In classification tasks, the goal is to predict a discrete class label, such as "positive" or "negative" for sentiment analysis. In the context of text similarity; In regression, the goal would be to predict a continuous similarity score for two pieces of text, such as a numeric value between 0 and 1. In classification, the goal would be to predict whether two pieces of text are similar or dissimilar, based on a set of predefined classes [34].

4.5 The Mohler Dataset

It consists of 10 assignments with 4-7 questions each and two tests with 10 questions each [7]. These assignments/exams were given to students in an introductory computer science class at the University of North Texas. There are 87 questions in total, and each question has an optimal answer provided by the examiner. Each question was answered by 26– 31 students. Each answer in the assignment is scored on a scale of 0 (not correct) to 5 (completely correct) by two computer science experts as evaluators. The standard score for each answer is the average of the two evaluators' scores. Table 1 shows a sample of the questions and their reference answers with samples of the students' answers and the average grade of the two evaluators.

Table 1: A sample of questions with the reference answers and answers provided by the student and the average grades of two assessors.

	Questions, reference answers, and student answers	Grades
Question: **Correct answer:**	What is the role of a header file? To store a class interface, including data members and member function prototypes.	
Student answer 1:	To promote function reusability	3
Student answer 2:	class definitions are placed here	3
Student answer 3:	A header file usually contains class and/or function prototypes.	4.5
Question: **Correct answer:**	What are the similarities between iteration and recursion? They both involve repetition; they both have termination tests; they can both occur infinitely.	
Student answer 1:	they are methods of repeating the same task.	2

Student answer 2:	Both involve a termination test. They use a control statement and repetition to solve the problem. They can also result in an infinite loop	5
Student answer 3:	They both use repetition, control, or test to terminate, and both can infinitely repeat if not defined correctly.	5

IMPLEMENTATION

The proposed method is implemented using Python programming language. Several text processing libraries provided by the programming language were employed, and for stop word removal, the NLTK [35] English stop word list was used. The implementation of SVR through Scikit-learn [36] was adopted for training with the Radial Basis Function (RBF) kernel. The dataset was randomly divided into 70% for training and the remaining 30% for testing.

Experimental Results

As evolution metrics, the Pearson correlation coefficient and the root mean squared error RMSE were calculated.

The Pearson correlation coefficient is a measure of the linear relationship between two variables. It is a statistical measure that ranges from -1 to 1, where -1 indicates a strong negative correlation, 0 indicates no correlation, and 1 indicates a strong positive correlation [37].

The Pearson correlation coefficient is calculated as follows:

$$r = (n\Sigma xy - \Sigma x\Sigma y) / \sqrt{((n\Sigma x^2 - (\Sigma x)^2)(n\Sigma y^2 - (\Sigma y)^2))} \quad (6)$$

where:
n is the number of observations
x is the variable for which you are calculating the correlation coefficient
y is the variable you are comparing x to
Σ is the sum symbol
x^2 is the square of x
y^2 is the square of y

Root mean squared error (RMSE) is a measure of the difference between the values predicted by a model and the true values. It is commonly used to evaluate the performance of a model, particularly for regression tasks [38].

The RMSE is calculated as the square root of the mean squared error (MSE), which is the average of the squared differences between the predicted values and the true values. The MSE is calculated as [38]:

$$MSE = (1/n)\ \Sigma(y\_predicted - y\_true)^2 \quad (7)$$

where:
n is the number of observations
y_predicted is the predicted value for a given observation
y_true is the true value for a given observation

Table 2 shows the obtained results of the proposed model on the Mohler dataset trained on all features and a different subset of features. The objective was to analyze the significance

of each feature, determining the most influential features that contributes to the student's answer grading.

Table 2: The results on the Mohler dataset with all features and with the exclusion of each feature.

features	Pearson r	RMSE
All features	0.631	0. 834
Without RASW	0.613	0.861
Without SASW	0.598	0.851
Without WO	0.584	0.849
Without LR	0.627	0.838

'

As shown, the results decline with the exclusion of each feature. The most influential features are the semantic weights of reference and student answer, while the word overlapping feature is the second most influential feature. Meanwhile, the length ratio has the least effect on performance.

Compared to existing studies, the proposal study shows a significant improvement as the results outperform all previous work on the same dataset. Table 3 provides a comparison with previous work on the Mohler dataset

Table 3: Comparison with other studies.

System	Pearson's r	RMSE
tf-idf[7]	0.281	1.085
Lesk[7]	0.450	1.034
Mohler et al.[7]	0.518	0.978
Sultan et al.[8]	0.592	0.887
Saha et al.[9]	0.570	0.902
Proposed Method	0.631	0.834

In most cases, the estimated score and the real score in the proposed method are very close. In the few cases where the difference was greater, the manual check showed that this happens because the student's answer came in a mathematical form while the reference answer was textual. Hence, the textual similarity measurement is not possible between the two answers. Students' answers also contained spelling errors that the teacher overlooked and considered correct. For example, the question "How many comparisons does it take to find an element in a binary search tree?" has the correct answer as "The height of the tree", while some student's answers were "log N". These answers, formulated as a mathematical expression, were considered correct and given a full score. However, the proposed method cannot derive a semantic textual similarity between a textual answer and a mathematical expression.

CONCLUSION

To cope with unprecedented times in teaching, a method is presented for automatic short answer grading by assessing the semantic similarity between the learner's answer and the reference answer. It was done by building a semantic network to denote the relationship between the two compared texts. Several features were extracted from this network to be used as input to the SVM model to predict the degree of similarity. In most cases, the proposed method gave a very close score to the human-based ones. Out of the evaluated 680 student answers, 396 cases saw the difference between the scores of the proposed method and the

teacher having less than 0.5 (Scaled from 0 to 5), and less than 1 in 547 cases (however, the difference was greater in a few cases). With manual checking, it was deduced that either the student's answer is in a mathematical form or contains spelling errors. In general, the obtained results were positive and encouraging, evidently superior to previous methods.

For future work, utilizing word embedding vectors trained on domain-specific corpora and using domain-specific lexical databases could be considered to give better results. The method could also be improved by extracting more features for more accurate results. In addition, automatic spelling correction of students' answers could refine the proposed method by increasing the chances of matching them with the reference answer.

Disclosure and conflict of interest

Conflict of Interest: The authors declare that they have no conflicts of interest.

References

[1] S. Burrows, I. Gurevych, and B. Stein, "The eras and trends of automatic short answer grading," *International Journal of Artificial Intelligence in Education*, vol. 25, no. 1. pp. 60–117, 2015. doi: 10.1007/s40593-014-0026-8.

[2] R. Shaw, Y. kyun Kim, and J. Hua, "Governance, technology and citizen behavior in pandemic: Lessons from COVID-19 in East Asia," *Prog. Disaster Sci.*, vol. 6, 2020, doi: 10.1016/j.pdisas.2020.100090.

[3] M. Sabeeh and F. Khaled, "Plagiarism Detection Methods and Tools: An Overview," *Iraqi J. Sci.*, vol. 62, no. 8 SE-Computer Science, pp. 2771–2783, Aug. 2021, doi: 10.24996/ijs.2021.62.8.30.

[4] M. Mohler and R. Mihalcea, "Text-to-text semantic similarity for automatic short answer grading," in *EACL 2009 - 12th Conference of the European Chapter of the Association for Computational Linguistics, Proceedings*, 2009, pp. 567–575. doi: 10.3115/1609067.1609130.

[5] S. H. Mijbel and A. T. Sadiq, "Short Answers Assessment Approach based on Semantic Network," *Iraqi J. Sci.*, vol. 63, no. 6 SE-Computer Science, pp. 2702–2711, Jun. 2022, doi: 10.24996/ijs.2022.63.6.35.

[6] S. H. Mijbel, P. Liatsis, and A. T. Sadiq, "Text Similarity Approach based on Semantic Networks and Words Description," *Des. Eng.*, pp. 15217–15228, 2021.

[7] M. Mohler, R. Bunescu, and R. Mihalcea, "Learning to grade short answer questions using semantic similarity measures and dependency graph alignments," in *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, vol. 1, pp. 752–762.

[8] M. A. Sultan, C. Salazar, and T. Sumner, "Fast and Easy Short Answer Grading with High Accuracy," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1070–1075. doi: 10.18653/v1/N16-1123.

[9] S. Saha, T. I. Dhamecha, S. Marvaniya, R. Sindhgatta, and B. Sengupta, "Sentence level or token level features for automatic short answer grading?: use both," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 10947 LNAI, pp. 503–517. doi: 10.1007/978-3-319-93843-1_37.

[10] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," in *EMNLP 2017 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, 2017, pp. 670–680. doi: 10.18653/v1/d17-1070.

[11] M. O. Dzikovska *et al.*, "SemEval-2013 Task 7: The joint student response analysis and 8th recognizing textual entailment challenge," in **SEM 2013 - 2nd Joint Conference on Lexical and Computational Semantics*, 2013, vol. 2, pp. 263–274.

[12] A. Condor, "Exploring Automatic Short Answer Grading as a Tool to Assist in Human Rating," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2020, vol. 12164 LNAI, pp. 74–79. doi: 10.1007/978-3-030-52240-7_14.

[13] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2019, vol. 1, pp. 4171–4186.
[14] 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.
[15] P. Lauren, G. Qu, J. Yang, P. Watta, G.-B. Huang, and A. Lendasse, "Generating Word Embeddings from an Extreme Learning Machine for Sentiment Analysis and Sequence Labeling Tasks," *Cognit. Comput.*, vol. 10, no. 4, pp. 625–638, Aug. 2018, doi: 10.1007/s12559-018-9548-y.
[16] A. Alqahtani, H. Alhakami, T. Alsubait, and A. Baz, "A Survey of Text Matching Techniques," *Eng. Technol. Appl. Sci. Res.*, vol. 11, no. 1, pp. 6656–6661, 2021, doi: 10.48084/etasr.3968.
[17] J. H. Assi and A. T. Sadiq, "Iterated Stimulation Artificial Recognition Immune System," *J. Comput. Theor. Nanosci.*, vol. 16, no. 3, pp. 1155–1158, Mar. 2019, doi: 10.1166/jctn.2019.8010.
[18] K. R. Chowdhary, *Fundamentals of Artificial Intelligence*. New Delhi: Springer India, 2020. doi: 10.1007/978-81-322-3972-7.
[19] R. Beckwith, C. Fellbaum, D. Gross, and G. A. Miller, "WordNet: A lexical database organized on psycholinguistic principles," in *Lexical acquisition: Exploiting on-line resources to build a lexicon*, Psychology Press, 2021, pp. 211–232.
[20] J. Pennington, R. Socher, and C. Manning, "Glove: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. doi: 10.3115/v1/D14-1162.
[21] N. J. Venhuizen, P. Hendriks, M. W. Crocker, and H. Brouwer, "Distributional formal semantics," *Inf. Comput.*, vol. 287, p. 104763, Sep. 2022, doi: 10.1016/j.ic.2021.104763.
[22] D. Chandrasekaran and V. Mago, "Evolution of Semantic Similarity-A Survey," *ACM Computing Surveys*, vol. 54, no. 2. 2021. doi: 10.1145/3440755.
[23] C. Fellbaum, "WordNet," in *Theory and applications of ontology: computer applications*, Springer, 2010, pp. 231–243.
[24] X. Zhang, S. Sun, and K. Zhang, "A new hybrid improved method for measuring concept semantic similarity in wordnet," *Int. Arab J. Inf. Technol.*, vol. 17, no. 4, pp. 433–439, 2020, doi: 10.34028/iajit/17/4/1.
[25] Y. Li, D. McLean, Z. A. Bandar, J. D. O'Shea, and K. Crockett, "Sentence similarity based on semantic nets and corpus statistics," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 8, pp. 1138–1150, 2006, doi: 10.1109/TKDE.2006.130.
[26] Z. Yin and Y. Shen, "On the dimensionality of word embedding," *Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.
[27] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," *Trans. Assoc. Comput. Linguist.*, vol. 5, pp. 135–146, Dec. 2017, doi: 10.1162/tacl_a_00051.
[28] L. A. Mullen, K. Benoit, O. Keyes, D. Selivanov, and J. Arnold, "Fast, consistent tokenization of natural language text," *J. Open Source Softw.*, vol. 3, no. 23, p. 655, 2018.
[29] M. Lusetti, T. Ruzsics, A. Göhring, T. Samardžić, and E. Stark, "Encoder-decoder methods for text normalization," 2018.
[30] R. D. Deshmukh and A. Kiwelekar, "Deep Learning Techniques for Part of Speech Tagging by Natural Language Processing," in *2nd International Conference on Innovative Mechanisms for Industry Applications, ICIMIA 2020 - Conference Proceedings*, 2020, pp. 76–81. doi: 10.1109/ICIMIA48430.2020.9074941.
[31] A. Thota, P. Tilak, S. Ahluwalia, and N. Lohia, "Fake news detection: a deep learning approach," *SMU Data Sci. Rev.*, vol. 1, no. 3, p. 10, 2018.
[32] I. Boban, A. Doko, and S. Gotovac, "Sentence retrieval using stemming and lemmatization with different length of the queries," *Adv. Sci. Technol. Eng. Syst.*, vol. 5, no. 3, pp. 349–354, 2020.
[33] Q. Li and T. Li, "Research on the application of Naive Bayes and Support Vector Machine algorithm on exercises Classification," in *Journal of Physics: Conference Series*, 2020, vol. 1437, no. 1, p. 12071.
[34] R. M. Aziz, A. Hussain, P. Sharma, and P. Kumar, "Machine learning-based soft computing

regression analysis approach for crime data prediction," *Karb Int J Mod Sci*, vol. 8, no. 1, pp. 1–19, 2022.

[35] W. Wagner, "Steven Bird, Ewan Klein and Edward Loper: Natural Language Processing with Python, Analyzing Text with the Natural Language Toolkit," *Lang. Resour. Eval.*, vol. 44, no. 4, pp. 421–424, 2010, doi: 10.1007/s10579-010-9124-x.

[36] P. Pedregosa, F; Varoquaux, G; Gramfort, A; Michel, V; Thirion, B; and Grisel, O. and Blondel, . and Prettenhofer *et al.*, "Scikit-learn: Machine Learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

[37] P. Schober, C. Boer, and L. A. Schwarte, "Correlation coefficients: appropriate use and interpretation," *Anesth. Analg.*, vol. 126, no. 5, pp. 1763–1768, 2018.

[38] T. O. Hodson, "Root mean square error (RMSE) or mean absolute error (MAE): when to use them or not," *Geosci. Model Dev. Discuss.*, pp. 1–10, 2022.