Automatic Subjective Answer Evaluation

Vijay Kumari, Prachi Godbole and Yashvardhan Sharma
Birla Institute of Technology and Science, Pilani, Rajasthan, India

Keywords: Natural Language Processing, Machine Learning, Subjective Answer Evaluation, Learning Assessments.

Abstract: The evaluation of answer scripts is vital for assessing a student's performance. The manual evaluation of the answers can sometimes be biased. The assessment depends on various factors, including the evaluator's mental state, their relationship with the student, and their level of expertise in the subject matter. These factors make evaluating descriptive answers a very tedious and time-consuming task. Automatic scoring approaches can be utilized to simplify the evaluation process. This paper presents an automated answer script evaluation model that intends to reduce the need for human intervention, minimize bias brought on by evaluator psychological changes, save time, maintain track of evaluations, and simplify extraction. The proposed method can automatically weigh the assessing element and produce results nearly identical to an instructor's. We compared the model's grades to the grades of the teacher, as well as the results of several keyword matching and similarity check techniques, in order to evaluate the developed model.

1 INTRODUCTION

Exam questions can be broadly categorized as either Objective or Subjective. Objective questions consist of selecting a response from a list of alternatives or providing a word or brief sentence. These types of questions have only one correct answer and can easily be graded automatically by an online assessment platform. On the other hand, Subjective questions require answers in the form of explanations. Essay questions, short answers, definitions, scenarios, and opinion questions are among them. It is vital to include human knowledge of the concepts when grading these detailed answers using Artificial Intelligence techniques, as well as to take into account linguistic factors like vocabulary, sentence structure, and syntax.

Due to the ongoing pandemic, education has undergone a significant transformation that has rapidly increased online learning, where instruction is delivered remotely via digital platforms and online classrooms. As the teaching-learning sessions have become virtual, online descriptive tests and assessments can be the best carriers of skills and personality enhancement. Students can learn better and experiment with their writing patterns by working on spontaneous thoughts regarding the subject. Descriptive questions help students develop a deeper engagement with the electronic content provided to them regularly. Students must adhere to the rules of content, syntax, and punctuation while submitting subjective answers, and they must explain their reasoning by giving examples, writing figures, or even sketching an illustration. The subjective content is more alluring and remarkable as a result of all these factors.

The proposed system's goal is to assist the evaluators in assigning grades to the answers by applying automatic grammar checks, a scan for the existence of important keywords or key phrases, and various similarity measures. The weights for the evaluation parameters are automatically assigned by the developed model.

The significant contributions of the paper are as follows:

1. Developed a model for evaluating answer scripts by taking the question, expected answer, and the student's answer as the input. The proposed model has been trained to weigh the evaluation components automatically. The questions can be weighted, and the student's response can be evaluated based on the question it has the most similarity.
2. A subset of a few students' answers can be tested to find the optimal combination of keyword extraction, summarization, and similarity check methods, and the rest of the answers can be evaluated accordingly.

Kumari, V., Godbole, P. and Sharma, Y.
Automatic Subjective Answer Evaluation.
DOI: 10.5220/0011656000003411
In *Proceedings of the 12th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2023)*, pages 289-295
ISBN: 978-989-758-626-2; ISSN: 2184-4313

3. Instead of only one model answer (expected answer), a set of multiple answers can be provided for each question.

The content of the paper is organized as follows. Section 2 explains the literature review of the automatic answer evaluation methods. Section 3 describes the proposed method for evaluating the descriptive answer. Section 4 presents the experiments and results. Section 5 provides the conclusion and recommendations for future work.

2 RELATED WORK

For the automatic evaluation of subjective answers, several techniques have been developed. Some of them are mentioned as follows:

Assessment of Answers in Online Subjective Examination. The following categories of questions were used to classify the questions: Define, Describe/Illustrate, Differentiate/Distinguish, Discuss/Explain, Enumerate / List / Identity / Outline, Interpret, Justify / Prove with considering answer in one sentence. The paragraph indexing module receives a set of query words from the question processing module, which it utilizes to carry out the information retrieval. For the answer, part-of-speech tagger (e.g., Python POS tagger), shallow parsing was performed to extract only the relevant word or phrase. Lexical resources like WordNet (Synonyms) for correctness were used. Paraphrasing (synonym based, lexical/structural-based, alteration based) was done to focus more on the answer intention. Semantic analysis was carried out using a word net dictionary, which determines the density of each word in a given sequence; if more than 50 % of the words in a sentence matched, the sentence was termed as correct. The overall performance of the system was found to be 70%. The major constraint of the system was that the questions, which included mathematical formulas, diagrams, and examples, were not considered (Dhokrat et al., 2012).

Artificial Intelligence-Based Online Descriptive Answer Verification System. The Cosine Similarity module and Text Gears Grammar API were two independent modules that made up the Answer Verifier Unit. Text Gears grammar API allows the integration of language processing methods. If the grammar is flawless, the API outputs 1, whereas if there are any errors in the sentence, the API outputs 0. The three attributes that made up the Result Set Unit were: Grammar, keywords, and QST (Question Specific Terms). Keywords had a value from 1 to 6, with 1 denoting excellent and 6 denoting poor. The grammar attribute has values between 0 and 1, with 1 denoting correct usage. Class values varied from 1 to 9, with 1 being the best and 9 representing the worst. The two main components of the system were the Information Extraction module and the Weighing Module. The system's main strength was its use of Cosine Similarity to match keywords. Fuzzy Wuzzy, a Python module, was utilized to determine an answer's grade (Jagadamba et al., 2020).

Machine Learning-Based Subjective Answer Evaluation. The system used Wordnet, Part of Speech Tagging, Lemmatization, and Tokenization of words and sentences to analyze the subjective answers. Data from the scanned images have been appropriately retrieved and organized. The examiner provides the input, which consists of the keywords and model response sets. Using machine learning techniques, sentences in the model answer have been clustered according to the ontology concepts and combined with the ontology map. The words in the model answer were merged with Ontology concepts once the words were fetched from the Ontology. The score for every keyword was determined by dividing the number of times each word appeared in the student's answers by the total number of words in their responses. (Bashir et al., 2021).

Evaluation of Descriptive Responses Using Semantic Relational Features. The model utilizes text patterns taken from the responses to be categorized into the answers. The Naive Bayes classifier was used to classify the questions into factual, inductive, and analytical categories. Retrieval of facts from the question is required for the factual questions. Who, where, when, how, what, or which inquiry categories were used to identify these queries. By using named entity recognition or stemming to separate the question's phrase or tag from the question, it was possible to infer the answer's emphasis. The categories for providing answers included explanation, comparison, cause and effect, sequence, and problem and solution. Cosine similarity and Jaccard matrices are used to obtain the similarity score. The total score is calculated by adding the value of the similarity score and the number of keywords. As a result of the various ways students may choose to represent the answer, further improvements in vocabulary are required. Additionally, grammatical analysis and fingerprinting can be used for evaluation to examine the meaning provided in responses (Nandini and Uma Maheswari, 2020).

Automatic Answer Script Evaluation Using NLP. For measuring similarities, various techniques like cosine similarity, Jaccard similarity, bigram similarity, and synonym similarity were utilized. Another strategy involved multiplying the parameter value and

weight value after giving each parameter a weight value depending on relevance. The text in the image has been extracted using the Python module pytesseract. To provide an automatic summary of the lengthy text, lemmatization and tokenization techniques were used. Based on the different types of questions, different weight values were given to each parameter. When assessing the answer script, the synonym parameter was given a higher weight than the grammar-checking parameter. The automatically awarded scores were very similar to the manually awarded scores when the student's answer and the correct answer have more structural and synonym similarities. In contrast, there was a sharp difference between the automated and manually scored marks when the student's answer and the correct answer had more Jaccard and Cosine similarity than structural similarity. The machine learning algorithms that can be trained by various determined parameters to forecast the marks of that answer script can be used for further improvements (Rahman et al., 2020).

3 PROPOSED SYSTEM

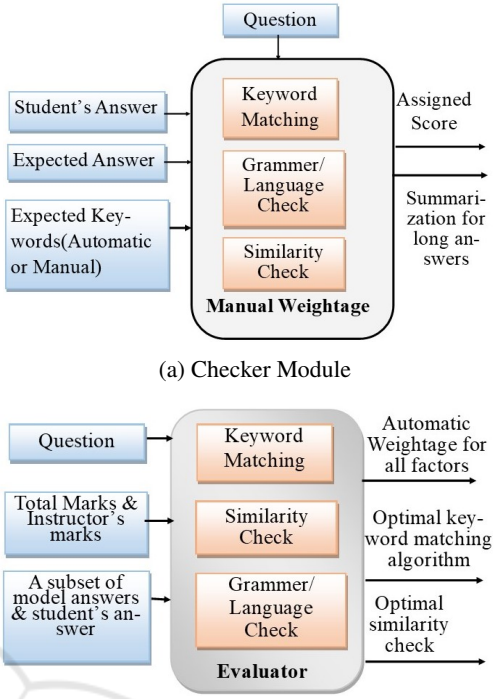
The techniques and methods work differently based on the type of question asked; the proposed model allows for multiple method combinations to achieve the optimal grade. For assessing the answers, not every evaluation criterion has to be given equal weight. As a result, the proposed method enables the marks in accordance with the weightage of the evaluation criteria. The proposed model has the following evaluation criteria:

1. **Keywords Matching.** Check for the presence of important keywords.
2. **Similarity Check.** Find the sentence similarity between the student's and model's responses.
3. **Grammar/Language Check.** Language score is determined by examining spelling and grammatical errors.

The developed model, as presented in Figure 1, consists of two parts: one is for the evaluation of answers (Checker), and another is for finding the optimal combination of evaluation techniques that can be used to evaluate answers to a particular question (Evaluator). The sum of the similarity, language/grammar, and keyword scores determines the final score.

3.1 Evaluator

A sample of the student's responses can be tested to find the most effective method combination for keyword extraction, summarization, and similarity check. The remaining responses can then be evaluated in accordance with the results.



(a) Checker Module

(b) Evaluator Module

Figure 1: Architecture of the proposed automated answer script evaluation model. A question, student answer, expected answer, and expected keywords (optional) are provided to the checker. The checker module matches the important keywords, grammar, and similarity with the expected answer and gives the final score assigned to the answer. The evaluator module will find the automatic weights for the evaluation criteria, optimal keyword matching, and optimal similarity check algorithm from the different combinations.

The user needs to input a question, model answers, student's answer, total marks, and marks awarded by the instructor for a set of answers. Weightage of grammar, keyword matches, and similarity checks are optional inputs. The model then computes the optimal combination of keyword, similarity, and grammar evaluation methods. The weighted sum of evaluation factors becomes the total marks that the model has awarded. If weights are not mentioned, each evaluation criterion is assigned a weight automatically by doing a similarity check between the input question and a list of previously evaluated questions. This is done by maintaining a CSV file (as shown in Figure 2) of previously evaluated questions. According to the evaluation of the questions, the CSV file is automatically updated. If the question is not present in the dataset, then the question and weights get added;

hence, the weights previously assigned to the most similar question are used for the new question.

3.2 Checker

The implemented system offers the user a choice of techniques for sentence similarity analysis, keyword extraction, and summarization. The combination obtained by an evaluator can be used in this part, or the user can try their own combinations. The system is implemented as a web application using Flask that takes a question with up to three model answers (Expected Answers), a student's answer, and total marks as user input.

For the manual option, the user must enter the desired keywords, separated by commas, and for the automatic option, the user must provide the desired number of keywords. Additionally, a selection for the keyword-matching technique is provided. In case the student's response is too lengthy, the user can then select a technique to summarise it. The maximum number of grammatical mistakes that may appear in the response can be specified. When comparing expected and student replies, the user has a choice of methods. The user must enter the percentage weighting of grammar, keyword matches, and similarity checks in order to calculate the final marks.

The process of extracting keywords involves selecting the most pertinent words and phrases from the text. Both the Manual and Automatic options are provided for keyword extraction. For the manual method, an input of comma-separated keywords is required. Methods used for automatic keyword extraction are:

1. **Term Frequency- Inverse Document Frequency (TF-IDF).** It is a statistical method for determining how pertinent a word is to a document within a group of documents. To accomplish this, the frequency of a word within a document and its inverse document frequency across a collection of documents are multiplied (Ramos et al., 2003).
2. **CountVectorizer.** It is a utility offered by the Python scikit-learn package that turns a given text into a vector based on the frequency of each word that appears across the full-text (Cou, n.d.).
3. **SpaCy.** It is a Python and Cython programming-based open-source natural language processing library. For trainable features such as named entity recognition, part-of-speech tagging, dependency parsing, text classification, and entity linking, it has built-in support. It segments the paragraph into pieces, and keywords can be identified by using parts of speech tagging and noun extraction (spa, n.d.).
4. **Rapid Automatic Keyword Extraction Algorithm (RAKE).** To identify the significant words or phrases in a document's text, it employs a set of stopwords and phrase delimiters (rak, n.d.).
5. **Yet Another Keyword Extractor (YAKE).** It is a simple unsupervised automatic keyword extraction technique that chooses the most significant keywords from a text by using statistical text features acquired from individual documents. (yak, n.d.).

3.2.1 Methods Used for Keyword Matching

For keyword matching, the RAKE and YAKE approaches are utilized, with the optimum strategy chosen based on the needs. The extracted keywords of the model answers are compared with the keywords of the student's response. The keywords are also matched with the synonyms of the keywords that were extracted from the model answer if the check synonym option is chosen.

3.2.2 Summarization Method

Summarization can be defined as the task of producing a concise and fluent summary while preserving essential information and overall meaning. If a student's answer is lengthy, summarization will help the evaluator to understand the answer's gist and determine the student's level of subject understanding. Methods used for summarization are:

1. **Cosine Similarity.** It is a Natural Language Processing method used for measuring the text similarity between two documents regardless of their size. The similarity between each pair of sentences in a paragraph is calculated and ranked. The highest-ranked sentences are used in summary (Rahutomo et al., 2012).
2. **BM25 Okpi.** BM is an abbreviation for best matching. It's a ranking algorithm that assigns a bunch of documents a ranking based on the search phrases that exist in each one of them, independent of how a document's search phrases relate to each other. (Robertson et al., 2009).
3. **BM25L.** It is an extension of BM25, which was developed to overcome the previous model's unfair preference for shorter documents over longer ones (Lv and Zhai, 2011). We observed that the BLEU (bilingual evaluation understudy) score and ROUGE (Recall-Oriented Understudy for Gisting Evaluation) score for BM25L were the best among other summarization methods.

	question	totalMarks	grammarwt	Keywordswt	similarityWt	questionType
1	what is Machine Learning?	8	20	60	20	Define
2	what is AI?	8	20	60	20	Define
3	What are the disadvantages of not confirming the dimensions that are shared among the fact tables of a data warehouse?	2	10	70	20	Explain
4	What are the advantages of dividing a longer load into a number of smaller sub-loads?	5	10	70	20	Explain
5	What is the difference between a closed cube and a full cube?	2	10	70	20	Comparision
6	How does the architecture of an online transaction processing system differ from that of a data warehouse from the perspective of scope	5	10	70	20	Explain
7	Write a paragraph on channel distribution.	8	40	40	20	Explain

Figure 2: A snapshot of the CSV is as follows (weightage given in percentage).

3.2.3 Grammar Method

Grammar is the structural foundation of one's ability to express oneself. It can help foster precision, detect ambiguity, and exploit the richness of expression available in the language. Automated grammar check is implemented using the Language_check library of Python, which specifies the mistakes along with the document's Rule Id, Message, Suggestion, and line number. The user can choose the maximum number of errors permitted as a cutoff point at which the grammar marks can be deducted.

3.2.4 Similarity Check

The similarity between the student's and model's answers is checked to determine how closely the student's response corresponds to the model's response. Methods used for checking similarity are:

1. **FuzzyWuzzy.** It is a Python library for string matching that uses Levenshtein distance to determine the differences between sequences. The Levenshtein distance between two words is the smallest number of insertions, deletions, or character swaps (single-character changes) required to change one word into another. (Fuz, n.d.).
2. **Jaccard Similarity.** Also known as the Jaccard index and Intersection over Union. It is a metric used to determine the similarity between two text documents by finding common words that exist over total words (Bag et al., 2019).
3. **TF-IDF.** It displays a word's frequency in a document as well as its inverse document frequency for a collection of documents (Ramos et al., 2003).
4. **BERT.** Bidirectional Encoder Representations from Transformers (BERT) is a transformer-based model to measure the semantic similarity between sentences. It converts all the sentences into a vector form and then determines the sentences that are closest in proximity to one another in respect of Euclidean distance or cosine similarity. (Devlin et al., 2018).

3.3 Weightage of Evaluation Components

There are different kinds of questions that can be asked, such as direct answers, conceptual explanations, definitions, and others. In terms of grammar, keyword matching, and similarity to the model answer, all of these questions cannot be assigned the same weighting. As a result, the developed system has the option of giving each component a variable weight. The marks calculated are based on the percentages assigned to each component.

4 EXPERIMENTS AND RESULTS

The end-to-end model is implemented using Python programming, Flask, HTML, and CSS on the front end. For an introductory computer science course, the assessment model has been tested with more than 20 questions and responses from 14 students.

The expected answer and student responses are compared to determine the similarity score. The keywords, grammar, and semantics of words are checked to ensure that the response is accurate. The evaluator module determines the best approach for similarity checking, or it can also be selected manually.

The total score is the sum of the similarity, grammar/ language, and keyword scores. According to Table 1, the student with ID8 does remarkably well. In order to evaluate how effectively the developed model performed, we compared the model's grades to the

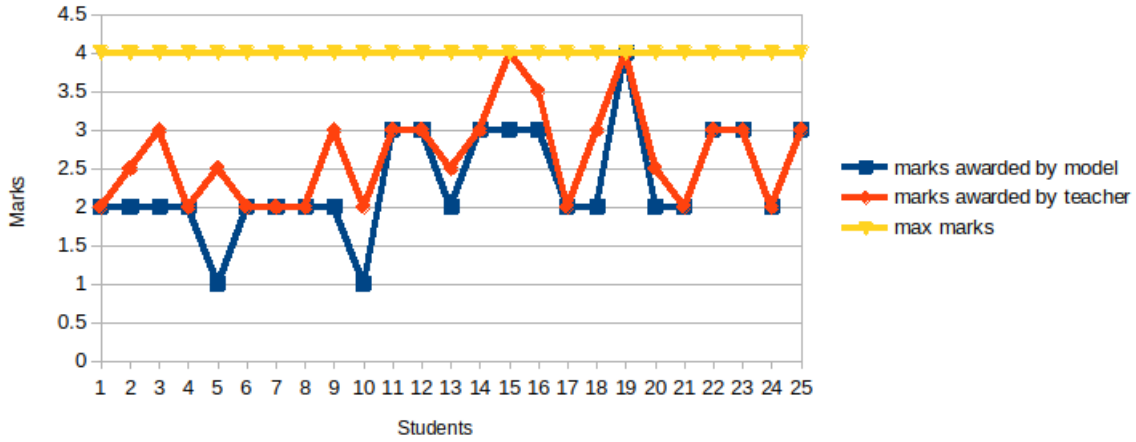


Figure 3: A comparison of the final score obtained by the students using the proposed model and marks awarded by the teacher.

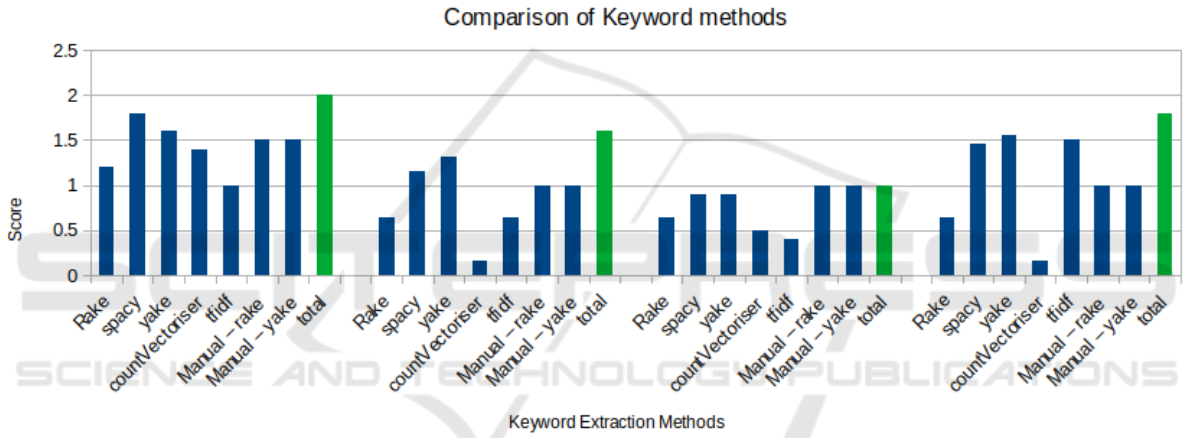


Figure 4: A comparison of the marks obtained by the students using various keyword-matching methods.

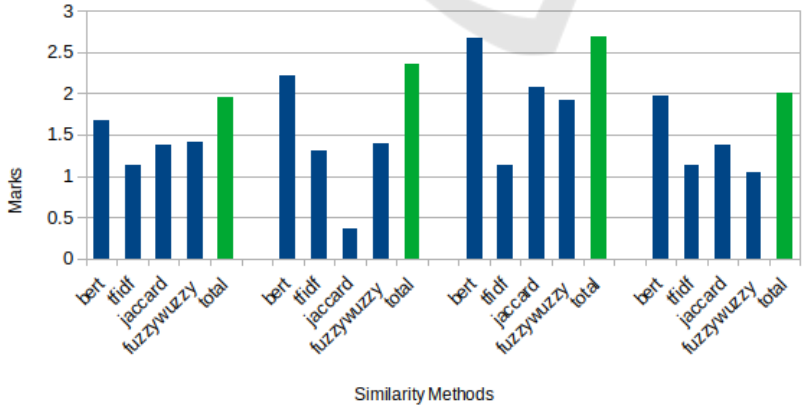


Figure 5: A comparison of the marks obtained by the students using various similarity methods.

Table 1: Scores obtained by the students in Q1.

Ques no	Student id	Marks awarded by model	Total marks
Q1	id1	2	4
Q1	id2	2	4
Q1	id3	3	4
Q1	id4	1	4
Q1	id5	2	4
Q1	id6	3	4
Q1	id7	2	4
Q1	id8	4	4

teacher's grades, and the results are shown in Figure 3. Figures 4 and 5 compare the scores given by different keyword matching and similarity check algorithms.

5 CONCLUSION AND FUTURE WORK

The grading of student responses is more difficult under the existing evaluation procedure. The evaluation scheme has significant issues that require a lot of human resources, time, and expertise. To overcome these challenges, this work developed a mechanism for automatically assessing answer scripts that use the question, the expected answer, and the student's response as an input. The proposed model is trained to categorize questions according to marks, which can assist in automatically assigning weights to components. The proposed method does not consider answers that include non-textual information like equations, graphs, and tables, which could be the direction of future research. Additionally, batch processing of all students' responses is a viable alternative to accessing a question at a time.

REFERENCES

CountVectorizer. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html.

Fuzzywuzzy. https://pypi.org/project/fuzzywuzzy/.

Rapid Keyword Extraction (RAKE). https://www.analyticsvidhya.com/blog/2021/10/rapid-keyword-extraction-rake-algorithm.

spaCy. https://spacy.io/.

yake. https://github.com/LIAAD/yake.

Bag, S., Kumar, S. K., and Tiwari, M. K. (2019). An efficient recommendation generation using relevant jaccard similarity. *Information Sciences*, 483:53–64.

Bashir, M. F., Arshad, H., Javed, A. R., Kryvinska, N., and Band, S. S. (2021). Subjective answers evaluation using machine learning and natural language processing. *IEEE Access*, 9:158972–158983.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dhokrat, A., Gite, H., and Mahender, C. N. (2012). Assessment of answers: Online subjective examination. In *Proceedings of the workshop on question answering for complex domains*, pages 47–56.

Jagadamba, G. et al. (2020). Online subjective answer verifying system using artificial intelligence. In *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*, pages 1023–1027. IEEE.

Lv, Y. and Zhai, C. (2011). When documents are very long, bm25 fails! In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 1103–1104.

Nandini, V. and Uma Maheswari, P. (2020). Automatic assessment of descriptive answers in online examination system using semantic relational features. *The Journal of Supercomputing*, 76(6):4430–4448.

Rahman, M., Siddiqui, F. H., et al. (2020). *NLP-based automatic answer script evaluation*. PhD thesis, DUET Journal.

Rahutomo, F., Kitasuka, T., and Aritsugi, M. (2012). Semantic cosine similarity. In *The 7th international student conference on advanced science and technology ICAST*, volume 4, page 1.

Ramos, J. et al. (2003). Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer.

Robertson, S., Zaragoza, H., et al. (2009). The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.