2022

Monograph on Theoretical Answer Evaluation System

Dr. Manish M. Goswami

Mr. Deepratna Awale

Mr. Sachin Jambhulkar

Mr. Alok Chauhan

[IJSRP Inc.]

1/15/2022

Monograph on

Theoretical Answer Evaluation System

Dr. Manish M. Goswami
Mr. Deepratna Awale
Mr. Sachin Jambhulkar
Mr. Alok Chauhan

Publishing Partner:
IJSRP Inc.
www.ijsrp.org

Preface

Theoretical Answer Evaluation System (T. A. E. S.) evaluates textual answers available in digital form and marks them based on a provided answer key. The system can identify if points given in the answer-key are similar to a point in the provided answer in any random order using cosine similarity. It can also check for plagiarism and grammaron a preliminary level. The study revolves around how a paper-checker without any domain knowledge of a subject can check a paper of a given subject, and mimics this process using Machine Learning. This system allows automated checking of subjective answers without the aid of human benefactor. It can find similarity in text passages with high accuracy and also distribute and cut marks based on number of points found and grammatical errors in the answer respectively. It is an essential in the time of a pandemic as it allows having subjective type tests, with automated checking. This system's score coincides with human given scores approximately 8 out of 10 times on a database of 1379 sentences and is faster than any human checker by only requiring 1 second for a passage of more than 200 words.This monograph shows working of TAES in complete detail.

We express our sincere thanks to all those who were involved in guiding and giving suggestions and encouragement to make this monograph complete.

Publication Partner:
International Journal of Scientific and Research Publications (ISSN: 2250-3153)

Copyright and Trademarks

Authors

1) Dr. Manish M. Goswami
Associate Professor,
Department of Information Technology,
Rajiv Gandhi College of Engineering and Research,
Nagpur.

2) Mr. Deepratna Awale,
Ex-Student,
Department of Information Technology,
Rajiv Gandhi College of Engineering and Research,
Nagpur.

3) Mr.Sachin Jambhulkar,
Assistant Professor,
Department of Electronics & Communication,
Vidarbha Institute of Technology, Nagpur.

4) Mr. Alok Chauhan,
Assistant Professor,
Department of Information Technology,
Rajiv Gandhi College of Engineering and Research,
Nagpur.

Table of Content

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Inspiration behind this project is the inability of fair evaluation system which will evaluate theoretical answers while checking papers of examination of students. Due to the sheer volume of students at our university it is physically impossible for faculty to only check papers that are of their own domain. Thus, it so happens that professors from various backgrounds end up evaluating papers of subjects they have lost mastery over. To evaluate these sheets, they use an answer key, which often might be a guide or one provided by the university. Due to this there is a weird distribution of marks, the papers checked my domain knowledge holders might be lenient with the points covered as long as the question is answered. And the one's checking from answer key will look for every single point in the answer key, if not found then the examinee will unfairly lose marks.

TAES aims to check papers based on answer key which will perpetuate a similar marking system for everyone. It will also automate the whole process of checking papers. However, to set realistic goals, we will skip the OCR part and only focus on evaluating textual answers. Our goal is to use NLP to automate paper checking, not to scan papers. We will not be going for domain knowledge acquisition as it over complicates the problem by having to maintain a knowledge base. This knowledge base will again be very contextual, for example for chemistry we will require a different set of jargon. It would be impossible to store answer to every single question and then use those answers to evaluate an examinee, we simply won't have enough time to build that database. Thus to have realistic goals we will try to batch evaluate an answers from an answer key.

Amidst the COVID-19, most of the examinations are now online and in MCQ form or one word answer form. The semester examinations of most universities have moved from subjective type to MCQ type. The team of T. A. E. S. wanted to be able to write subjective type answers and check it on machines. However, there are several hardships and some demerits of doing so, like students with low typing speeds or students without a computer would be typing slowly, besides not everyone has a computer. Even after considering these demerits, we found that the application of such a system is vast and can change most conventional methods to evaluate examinees. Some of the use cases for the Theoretical Answer Evaluation System (here on out referred to as T. A. E. S.) are:

a) Subjective University Examination

b) Websites can leverage T. A. E. S. to include long answers.

c) Companies and Universities can shortlist the candidates using value-based questions like, "Why would you like to work for our organization?"

T. A. E. S. changes what online examination stands for and leverages machine learning to reduce the strenuous task of long answer evaluation. It mimics the technique used by a professor with no domain knowledge of a subject to check any given answer. The technique used by such a professor is similar to the one given below:

a) Check if answer length matches the expected passage length.

b) Make/ acquire answer-key of necessary points that should exist in the answer.

c) Compare if examinee has stated enough points according to marking structure.

d) Cut marks for incomplete/ missing points and grammatical errors.

e) If evaluator finds that two students have copied answers, he can penalize or mark it as cheating.

f) Deduct marks based on word count.

A. Problem Statement

Our goal is to make an automated system to evaluate answers; the input will be answer-key and the batch of answers to check. To use answer-key based method, we will need the following:

a) Answer key
b) Answer (in our case batch of answers)
c) Stop words to reduce processing times.

Stop words are the words that can stop existing in the sentence but the sentence meaning still does not change. We use these inputs to evaluate the similarity between two answers, which will help us to mark the answers.

Our novel approach aims to use machine learning to automate the theoretical answer checking task (of digital format answers) to help innovate a system that can do paper checking. The project will evaluate an answer based on a given answer key by using Natural Language Processing algorithm and regex string matching. After a definite finite iteration, we will reduce the human interventions to rechecking of the papers.

B. Aim and Objective

Following define our aim and objective for making this project.

a) The project will evaluate an answer based on a given answer key by using Natural Language Processing algorithm and regex string matching.
b) This project will summarize given answers and compare the summary in the answer key, which will be quicker than human checked answers due to the enormous amount of text to go through.
c) We can implement T. A. E. S. on websites, which will give us the ability to check theoretical answers along with multiple-choice questions and single word answers.
d) Depending on the similarity of the answer and the expected answer (in the answer key), the system will automatically mark the answer correct.
e) After a definite finite iteration, we will reduce the human interventions to rechecking of the papers.

C. Summary

The chapter focuses on the introduction to the industry requirement for the development of a software system. In addition, the chapter provides details about the aim of the project/thesis, and gives a general idea about how we can achieve the corresponding tasks. Moreover, how we can create the complete system.

This project will summarize given answers and compare the summary in the answer key, which will be quicker than human checked answers due to the enormous amount of text to go through. We can implement T. A. E. S. on websites, which will give us the ability to check theoretical answers along with multiple-choice questions and single word answers. Depending on the similarity of the answer and the expected answer (in the answer key), the system will automatically mark the answer correct.

2. REVIEW OF LITERATURE

As AI has evolved through time, we have made numerous tries to make our computers understand us. We have tried but have failed miserably in a broad perspective. However, the requirement and curiosity of this utopian future where machines understand us humans has led to the field of Natural Language Processing (NLP). Even though we have a vast sub field of NLP today, realistically we are merely using statistics and making our computers learn what something ‘often’ means. The drawback of this approach is our system acts like a monkey mimicking the past, but it doesn’t truly understand what is going on. But due to the powerful computers and programs we have, we’ve been able to crunch numbers to predict what something means. Evaluating if a statement is true or false has been one of the most favorite topics in NLP. For a machine the sentences are merely a sequence of 1’s and 0’s, imagine trying to tell if something was 1 or 0 with just those two as input. So, what scientists have come up with is understand the general direction of a sentence (where a sentence points) and if it is pointing towards something true (based on historic knowledge), we mark it as true. Now, true or false requires fact-based knowledge even for humans, but one thing that doesn’t is checking if two things are similar. Us humans can just look at things and judge if they are similar or not. Let’s explore how it is done in machines
We have searched more by leaving these four points:

a) Answer Evaluation Using Machine Learning McGraw Hill 2018[1] uses Neural Nets with ReLU activation function to perform OCR. This paper tests if the words in the answer are synonymous to the words they used in their database. Though they have an accuracy of up to 87 percent it does not check for answers that require more than a word to be answered. Furthermore, this paper fails to check grammatical soundness, and meaning of the sentence. The examinee only needs to write keywords to get a score. It also assumes that the text from the OCR has a 100 percent accuracy which is impossible even for the best OCR models out there.

b) Another research project Subjective Answer Evaluation Using Machine Learning [2], uses a comprehensive technique with Naïve Bayes Classifier, it also accounts for special requirements of questions, grammatical soundness and does extract key phrases. The inputs required are: Answer written by examinee and sample answer sets. The sample answer sets define what the score will be, distance between examinee answer and set is calculated and is put in that set category with lowest distance. The research states that the agreement of their model and examiner is 90% which is impressively high.

c) Semantic Analysis of Long Answers[3], International Research Journal of Computer Science, Volume 8, Issue 4, sets out an applicative proposition for checking long answers. It proposes it should be done in two phases, keyword extraction and matching those keywords in the given answer, though it is unclear on how this should be implemented, the thought process and methodology is something we can definitely use to benefit our system. T. A. E. S follows the workflow of this paper and introduces stronger and better techniques along with additional features. One of the limitations of this paper is that it just considers the frequency of words to make key phrases. It doesn't actually use NLP to extract key phrases from sentences, rather uses lists of words and punctuations to omit. It uses direct word comparison which accounts for synonymousness and not semantic similarity.

d) Automatic keyword extraction from individual documents introduces a technique called RAKE (Rapid automatic keyword extraction), which utilizes three different evaluation metrics. RAKE starts by making an array of words using specified word delimiters. Then the array needs to split into arrangement of adjoining words at phrase delimiters and stop word positions. This technique is a quick and easy way to extract key-phrases, which we will be using as a part of T. A. E. S. for key-point extraction. [4]

Leacock C. & Chodorow M. published a paper called C-Rater: Automated Scoring of Short-Answer Questions [5] which is an automatic scoring system that evaluates essays. It utilizes predicate-argument structures, pronominal reference, morphological investigation, and a lot of domain knowledge. It has a high score of 84% when it comes to human agreeableness on evaluation.

There are also some popular works like Automatic Essay Assessment, [6] An Approach to Evaluate Subjective Questions for Online Examination System, [7] An Intelligent System for Evaluation of Descriptive Answers, [8] and Unsupervised Learning by Probabilistic Latent Semantic Analysis. [9] These prominent works do provide multiple ways, some naïve and some extremely field oriented, some statistic intensive, and some algorithm based, however they do not fit into our need.

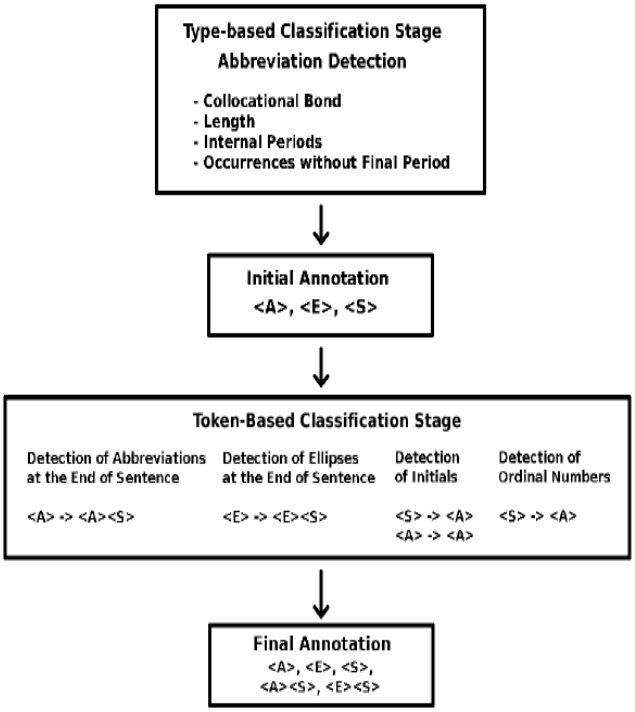

Fig. 2.1 – Architecture of Punkt System

3. METHODOLOGY

Now it is the time to articulate the research work with ideas gathered in above steps by adopting any of below suitable approaches:

A. Overview

This chapter describes the work done for the software system. It contains all the information about the tools and technologies, languages used in development of the system. Each sub chapter contains brief about the technologies used along with the implementation of the whole system.

Theoretical Answer Evaluation System or T. A. E. S. evaluates an answer based on a given answer key by using Deep Averaging Network and regex string matching. This project will summarize given answers and compare the summary in the answer key, which will be more accurate than human checked answers due to the enormous amount of text to go through.

We can implement T. A. E. S. on websites, which will provide them the opportunity to examine amazingly theoretical responses as well as astonishingly multiple-choice questions and fill in the gaps. We can use the rake module to significantly cut text lengths and extract the gist of any type of content.

B. Workflow

T. A. E. S will improve upon and follow the workflow of the paper 'Semantic Analysis of Long Answers'that is summarized below

a. Decomposing Passages
b. Acquiring key-words
c. Evaluating semantic similarity
d. Marking the answers

In accordance to the above steps, we have decided to perform the following steps:

a) Acquire sentence list by decomposing paragraphs using NLTK and regex.
b) Get key-phrases using RAKE algorithm.
c) Encode sentences using Google Universal Sentence Encoder.
d) Evaluate Semantic Similarity by applying cosine similarity.
e) Mark the answer based on points found.
f) Check grammar using python library.
g) Check for plagiarism among set of examinee answers.

The system will automatically mark the answer based on the similarity of the answer and the expected response (in the answer key). After a set number of repetitions, human interaction will be limited to rechecking the papers. The breakdown of the workflow is as follows:

B.1 Decomposition of Pages

T. A. E. S takes replies longer than a sentence and breaks them down into little easy words using NLP methods. We utilize the sentence tokenizer from the NLTK tokenize package to split up texts into sentences. To create a model for abbreviated words, collocations, and words that start sentences, the NLTK package employs PunktSentenceTokenizer, [4] an unsupervised method.

By default, the intention of the model is to preserve whitespaces and to learn parameters (including abbreviations) in an unsupervised way from a corpus identical to the target domain. (Fig 2.1).

We clean the obtained sentences, and eliminate any empty sentences comprising only whitespace characters. Using regex, we remove unnecessary whitespace characters and symbols and replace them with a single whitespace character. We save these filtered sentences in a Python list and refer to them as the 'sentence_list' from here on.

B.2 Acquiring Key Phases

Rapid Automatic Keyword Extraction [RAKE] extracts important words from a sentence list or text using statistical approaches. It does provide keyword extraction from texts, but for more accuracy, we will use the

sentence list obtained with NLTK. RAKE, a Python module based on the NLTK module, allows us to set the minimum and maximum length of keystrokes, which we have set to 3 and 10 after proper tuning.

RAKE has a handy method called get_ranked_phrases (), that returns the phrases found in the sentence sorted by rank. The following ratio defines rank:

$$Rank = \frac{degree(word)}{frequency(word)} \quad (1)$$

This ranking system favors words that appear frequently in longer candidate keywords. We save the newly created keys in a separate Python list, and referenced as 'answer_key' from now on.

B.3 Sentence Encoding

Sentence Encoding means to represent sentences in a form such that they become useful to us. English language or for that part any language is not understood by a machine, thus we need to represent it in a binary form. But that doesn't solve any problems as that data is unusable. Recent paradigms of NLP propose to represent sentences as vectors and then compare those vectors. The paper Semantic Analysis of Long Answers, highlights use of deep averaging network to acquire sentences encoded into high dimensional vectors. We will be using the same on our sentence_list and answer_key.

Google's Universal Sentence Encoder (v4)[10] encodes a larger than word corpus into a 512-dimensional vector that can be used for various NLP tasks. The model is prepared and enhanced for large text corpuses and trained on a variety of text chunks from twitter feeds, forms, literature paragraphs, etc. It uses a deep averaging network (DAN) encoder to encode the sentences.

Figure 3.1 depicts the operation of Google Sentence Encoder. The figure below shows the working of Google Sentence Encoder.

Fig. 3.1 – Deep Averaging Network Encoder

In T. A. E. S the encoding is done by a method promptly named encode(). It returns a l2-normalised vector embedded using TensorFlow 2 library. Following DAN's documentation it is clear that, the longer the passage the more diluted the vectors. Thus, we will use small sentences from the sentence_list and answer_key to mitigate the dilution problem.

B.4 Evaluating Semantic Similarity

We use cosine similarity to evaluate semantic similarities. Cosine similarity calculates the similarity of two vectors in an internal product space. We calculate it using the cosine of the point between two vectors and determine whether two vectors are generally pointing in the same direction. In text analysis to quantify report likeness, cosine similarity is highly popularized.

Cosine similarity is a measure of similarity that can compare documents or rank them in relation to a given vector of query words. Assume x and y are two vectors to be compared. When we use the cosine measure as a similarity function, we get:

$$sim(x,y) = \frac{x.y}{||x||||y||} \quad (2)$$

In (2), ||x|| denotes the Euclidean norm of vector x = (x1, x2, ... xp), which is defined as $x_1^2 + x_2^2 + x_3^2 + \ldots + x_p^2$.

In theory, it is the vector's length. The metric computes the cosine of the angle formed by the vectors x and y A cosine value of 0 indicates that the two vectors are perpendicular to each other (orthogonal) and do not have a match. The closer the cosine value is to one, the smaller the angle and the greater the vector-vector match.

T. A. E. S. provides a calculate cosine similarity () method that uses TensorFlow 2 methods reduce sum (), multiply (), and clip by value () to return a score ranging from -1 to 1. Because the encoder supports and prefers long text blocks, sentences, and more-than-word-length phrases, we can evaluate answers using the configurations shown in Table 1:

Answer Key Form	Answer Form
Key Phrase	Passage
Key Phrase	Sentence
Key Phrase	Key Phrase
Sentence	Passage
Sentence	Sentence
Sentence	Key Phrase
Passage	Passage
Passage	Sentence
Passage	Key Phrase

Table 3.1 – Answer Key and Answer Configurations

The recommended configurations are discussed in the section Results and Discussion.

B.5finding the best Match for Key Phrase

Let us look at how T. A. E. S checks for key-phrases in answers. It requires only two things, the answer-key and the answer. Below is the algorithm for finding the best fit for a key-phrase:

1. Start
2. Initialize a hash map (python dictionary) called hitlist with the key-phrases as the keys of the dictionary.
3. Initialize an empty list to store score, say ‘score_list’.
4. Copy sentence_list to a new list, say ‘alist’
5. For every key in hitlist:
    a. If alist is empty:
        i. Append ‘None’ as value of the current key.
        ii. Store 0.0 as the similarity score in a score_list.
    b. Else:
        i. Initialize score at 0.0
        ii. Initialize new_score at 0.0
        iii. For every sentence in alist:

1. Calculate new_score
2. If new_score> score:
    a. Score = new_score
    b. Append sentence as the value of current_key
3. Else: Continue

iv. Append score to score_list

v. Delete sentence in alist

6. Stop

In the above algorithm, new_score is the cosine similarity of current key and the sentences.

B.6Marking System

T. A. E. S provides an extensive marking system with multiple parameters. It can also mark an answer partially if the sentence similarity match is weak.

The score for one single point is given by the fraction of total marks and number of points in that answer. E.g., if there are 5 points in the answer-key and the question paper marks the answer out of 5 marks, score for a single point will be 5 marks/ 5 points, which is 1. If 'm' is score rewarded per point, then m is defined as in (3).

$$mpp = \frac{total\ marks}{total\ number\ of\ points} \tag{3}$$

The total marks thus will be given by the summation of all m. TAES also lets us the grammatical soundness of a sentence. The max penalty for grammar can also be set. The grammatical penalty is calculated as shown in (4).

$$gp = \left((T - t) * p\right)\ \%\ M \tag{4}$$

In equation (4), gp is total marks to cut for grammar, T is total grammatical errors, t is the threshold, r is points to cut per mistake and M is the max penalty for grammatical errors. Note: '%' here represents remainder operator and not percentage sign.

$$Total\ Marks = \sum_{i=0}^{n} mpp_i - gp \tag{5}$$

Note: In (5), gp is subtracted from summation of m for i in range 0 to n, where n is the number of points in the answer.

The multiple parameters for calculating marks are:

1. Maximum marks

2. Total Points

3. Accuracy Filter

4. Acceptable error

The user enters maximum marks, and they represent the total number of marks awardable for a given answer. Total Points is calculated; it is the number of points in the 'answer_key'.

Accuracy Filter is the percentage of similarity that a point in the answer must have to a key in 'answer_key'. For example, a 60 percent accuracy filter means that the cosine similarity between the key and the point must be 0.60 or higher in order to receive full marks.

Acceptable Error is the tolerance of accuracy in order for a point 12 of the mpp. This is set to ten percent by default. For example, if the accuracy filter is set to 60% and the acceptable error is set to 10%, any point with a score between 50% and 60% will be marked as 12 of the mpp.

Assume there are 5 points in the answer key and five marks. If the accuracy filter is set to 60 and the acceptable error is set to 10.

1. A point with a score of 60% or higher will receive one mark.
2. A point is worth 12 points if it has a score of 55 percent (which is between 50 and 60 percent).
3. If a point's score is less than 50%, it receives 0 points. With the help accuracy filter and acceptable error, it is possible to mark answers partially.

B.7 Grammar and Plagiarism

T. A. E. S. does not include a robust grammar and plagiarism checker. They are, however, operational and in the beta phase. We check the grammar with the language tool library in python, and check the plagiarism with the help of cosine similarity module. Because the grammar checker detects minor infractions, we will have to filter those out (offenses like extra whitespace, word redundancy). Finally, we deduct the marks in accordance with equation (4).

The plagiarism checker employs cosine similarity once more and anticipates a path containing text files containing answers in passage form. It checks the two files for similarities automatically but we do not recommended doing that with low similarity scores because the checking of answers is also dependent on similarity and the plagiarism checker does the same.

4. RESULTS AND DISCUSSION

A. Single Response Testing

T. A. E. S. can evaluate a single answer written by the user in a specific format. For testing purposes, we need to provide an answer key. It stores and prints a data frame of the marks allotted, the marks awarded to points, and the score of the point after you set the maximum marks and accuracy filter as shown in Fig. 4.1.

The two passages fed into the system are similar and paraphrased to vary the similarity. The answer-key contains the actual answer as well as an additional point "Diplomatic point here to show that if it exists, marks will

not be allotted." We use this point to demonstrate that if we are not able to find a point in the answer key then we will not allot any marks.

MARKS OBTAINED: 4.7
Total points found in answer: 14 of 15

	Answer-Key	Sentence	similarity	marks
0	Machine learning is a branch of artificial int...	ML is a part of AI for building applications t...	0.712884	0.333333
1	In data science an algorithm is a sequence of...	In data science an algorithm is sequential st...	0.881099	0.333333
2	In machine learning algorithms are trained ...	In ML algorithms are trained to discover patt...	0.831787	0.333333
3	The better the algorithm the more accurate th...	The better the algorithm the more exact the p...	0.777781	0.333333
4	Today examples of machine learning are all ar...	Today instances of ML are surrounding us	0.700668	0.333333
5	Digital assistants search the web and play mus...	Computerized partners like Siri Alexa etc se...	0.720546	0.333333
6	Websites recommend products and movies and son...	Sites suggest items and films and songs depend...	0.762458	0.333333
7	Robots vacuum our floors while we do	Robots vacuum our floors while we do	1.000000	0.333333
8	something better with our time	something better with our time	1.000000	0.333333
9	Spam detectors stop unwanted emails from reach...	Spam identifiers prevent undesirable messages ...	0.787989	0.333333
10	Medical image analysis systems help doctors sp...	Clinical picture examination frameworks help s...	0.752076	0.333333
11	And the first self driving cars are hitting th...	What s more the primary self driving vehicles...	0.729809	0.333333
12	We can expect more	We can hope for something else	0.704636	0.333333
13	As big data keeps getting bigger as computing...	As large information continues getting greater...	0.707220	0.333333
14	Diplomatic point here to show that if it exist...	None	0.000000	0.000000

Fig. 4.1 – Representation of Mark Distribution Table

B. Answer Batch Testing

We saved scraped data from a web search for "What is Machine Learning?" into various text files for batch testing. We used Google as the search provider, and scraped all results on the first page. We took those results and input the eight text files containing answer passages into T. A. E. S. for scoring. The answer-key can be loaded from a file or through the user interface, and it returns mark distribution for each student.

Marks have been processed.

	Student	Marks Obtained	Total points found (out of 14)
0	emerj	2.57	7
1	expertai	4.29	14
2	ibm	6.00	14
3	royalsociety	1.93	7
4	technologyreview	2.57	10
5	wiki	4.29	14
6	mygreatlearning	4.71	14
7	infoworld	2.36	6

Fig. 4.2 – Batch Evaluation of Answers

C.Using Metrics to Evaluate Model

Correlation is how close in distance are two data sets to each other. Pearson Product Moment Correlation (PPMC) shows the two-dimensional similarity of two data sets. Two letters are used to represent the Pearson correlation: Greek letter rho (ρ) for a population and the letter r for a sample. It is defined as in (6).

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{\left[n\sum x^2 - (\sum x)^2\right]\left[n\sum y^2 - (\sum y)^2\right]}} \quad (6)$$

In statistics, the p-value is the probability of obtaining results at least as extreme as the observed results of a statistical hypothesis test, assuming that the null hypothesis is correct. Key takeaway is that the higher the p value the higher will be the consistency. Lower p-value means that results are more dynamic.

We will be using PPMC and p-value to evaluate our model. PPMC (higher the better, can be used as percentage if multiplied by 100, ranges from 0 to 1). P-value will tell us how consistent our model will work (ranges from 0 to 1, higher is better).

.

D.Benchmark for Semantic Textual Similarity

The Semantic Textual Similarity (STS) Benchmark assesses the degree to which similarity scores computed using sentence embedding align with human judgments. The benchmark requires systems to return similarity scores for a diverse selection of sentence pairs formalized

Figure 4.1: Mark Distribution Table Representation

Figure 4.2. shows a batch evaluation of answer-return similarity scores for a wide range of sentence pairs. We use the Pearson correlation to assess the accuracy of the machine similarity scores in comparison to human judgments. The PPMC and p-value score of the DAN model used in the encoder as shown in Table 4.1.

Scale	Value
PPMC	0.7821
P-value	3.8156701636229695e-285

Table 4.1 – Accuracy Metrics of Model

E. Comparison with Other Models

MODEL	% ACCURACY
Glove	40.6
Word2Vec skipgram	56.6
Doc2Vec (SEF@UHH)	59.2
C-Phrase	63.9
PV-DBOW Paragraph vectors, Doc2Vec	64.9
Charagram (uses PPDB)	71.6
Paragram-Phase (uses PPDB)	73.2
InferSent (bi-LSTM trained on SNLI)	75.8
T.A.E.S	78.21

Table 4.2 – Comparison with Other Models

DAN outperforms the majority of models that evaluate semantic similarity despite the fact that it is not one of the best model available but it performs with high accuracy. It is worth noting that we are testing DAN, which is the model used by Google Universal Sentence Encoder.

In T. A. E. S., we use the same encoder. The model has not been hyper-tuned or specified for this task, but it still outperforms it. Overall, it provides a trustworthy model for encoding our text.

We discovered that it is best to embed phrases obtained by RAKE as answer-key and sentences obtained by nltk.tokenize for answers after several tries. Alternatively, we can embed sentences for both the answer-key and the answer directly. Both produce similar results. Other configurations are unintuitive and perform only mediocrely.

The recommended accuracy filter ranges from 60 to 70%. The minimum score to consider when determining whether it is a plagiarized answer is 50%. This configuration corresponds to the data we used; we suggest tuning the model based on the data and requirements.

F.Time Taken to Check Answers

No of answer	Time Taken
1	1.156s
8	65.26s

Table 4.3 – Time Comparison

Note: The time required to check 8 answers includes importing text files (which would be faster if the files were on the local machine), calculating similarity and marks for each point, creating a pandas data frame for each answer's mark distribution, and updating a data frame that stores a list of marks obtained by students.
The time it takes to evaluate eight passages is just over a minute, which is far less than the time it would take any human evaluator to read and evaluate the same passages.

5. SUMMARY AND CONCLUSION

A.Summary

To summarize, there is still no way to make computers understand human language. Our best bet is to mimic that understanding. Thus, using deep learning, statistics, natural language processing algorithms, and regular expressions, we were able to extract key-phrases, encode text into vector format and check the answer based on key-phrases with an accuracy of 78.21%. Not only that, but it is also in beta-support for grammar and plagiarism checking.

B. Conclusion

Currently, the NLP techniques for interpreting textual data are primitive and computationally intensive. Simple operations such as comparing two sentences, representing a sentence as a vector, summarizing paragraphs, and evaluating long written format text require a lot of processing power.

The limitations of this system are that it cannot evaluate answers without an answer-key. Due to current computational limitations and existing cognitive machine learning approaches, we cannot provide OCR support for such long answers.

C. Future Scope

Finally, T. A. E. S. can be an essential resource during the pandemic where most educational institutes have switched to an online learning environment. Using T. A. E. S., the universities, companies, and websites can accept and evaluate textual answers present in digital format without human intervention.

6. References

[1] S. Bharadia, P. Sinha, and A. Kaul, *Answer Evaluation Using Machine Learning*. 2018.
[2] P. Patil, V. Miniyar, and Amol, "Subjective Answer Evaluation Using Machine Learning," 2018.
[3] DeepratnaAwale, "Semantic Analysis of long answers," *IRJCS*, vol. 8, no. 4, 2021.
[4] T. Kiss and J. Strunk, "Unsupervised Multilingual Sentence Boundary Detection," *Computational Linguistics*, vol. 32, pp. 485–525, Dec. 2006, doi: 10.1162/coli.2006.32.4.485.
[5] S. Rose, D. Engel, N. Cramer, and W. Cowley, "Automatic Keyword Extraction from Individual Documents," in *Text Mining: Applications and Theory*, 2010, pp. 1–20.
[6] T. Landauer, D. Laham, and P. Foltz, "Automatic Essay Assessment," *Assessment in Education Principles Policy and Practice*, vol. 10, Nov. 2003, doi:10.1080/0969594032000148154.
[7] S. Praveen, "An Approach to Evaluate Subjective Questions for Online Examination System," 2014.
[8] V. Bagaria, M. Badve, M. Beldar, and S. Ghane, "An Intelligent System for Evaluation of Descriptive Answers," in *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*, Dec. 2020, pp. 19–24, doi: 10.1109/ICISS49785.2020.9316110.
[9] T. Hofmann, "Unsupervised Learning by Probabilistic Latent Semantic Analysis," *Machine Learning*, vol. 42, no. 1, pp. 177–196, 2001, doi: 10.1023/A:1007617005950.
[10] D. Cer*et al.*, "Universal Sentence Encoder," *CoRR*, vol. abs/1803.1, 2018, [Online]. Available: http://arxiv.org/abs/1803.11175.
[11] L. S. Daniel Cer, Mona Diab, EnekoAgirre, Iñigo Lopez-Gazpio, "STS Benchmark," *Proceedings of the 10th International Workshop on Semantic Evaluation*. http://ixa2.si.ehu.eus/stswiki/index.php/STSbenchmark.

Glossary

Sr. No.	Abbreviation	Meaning
1.	T. A. E. S.	Theoretical Answer Evaluation System
2.	MCQ	Multiple Choice Questions
3.	COVID-19	Corona Virus Disease – 19
4.	AI	Artificial Intelligence
5.	NLP	Natural Language Processing
6.	ML	Machine Learning
7.	API	Application Programming Interface
8.	NLTK	Natural Language Tool Kit
9.	RAKE	Rapid Automatic Keyword Extraction Algorithm
10.	NAEP	National Assessment for Educational Progress
11.	DAN	Deep Averaging Network
12.	PPMC	Pearson Product Moment Correlation
13.	STS	Semantic Textual Similarity
14.	OCR	Optical Character Recognition

Index