KALYANI GOVERNMENT ENGINEERING COLLEGE
Kalyani - 741235, Nadia, WB

Affiliated to
Maulana Abul Kalam Azad University of Technology

A report on

LEVERAGING EVALUATION OF ANSWERS USING FEW-SHOT LEARNING

Submitted by:

RAJENDRA KUMAR SHAW
Roll No :- 10271022021

Table of Contents

CHAPTER 1

An automated response evaluation system is a technology-driven solution that employs machine learning and natural language processing methodologies to appraise and score student answers. Its primary goal is to streamline the grading procedure, offering swift, precise, and uniform assessments of answer scripts while minimizing the need for manual intervention. This system scrutinizes the substance, accuracy, and caliber of responses, employing predetermined criteria and rubrics to produce impartial evaluations.

Background

Current Scenario: The present state of online question-answer assessment predominantly relies on machine-based evaluation, which is commonly employed for Multiple-Choice Questions (MCQs). In this context, machine-based evaluation utilizes algorithms to autonomously appraise and score MCQ responses based on predefined criteria, such as selecting the correct option or providing the most accurate answer. These algorithms analyze responses and furnish immediate feedback, ensuring efficient evaluation and prompt results. The process integrates crowdsourcing, reputation systems, automated assessment, human moderation, and user feedback to gauge the correctness and quality of online-provided answers.

Assessing descriptive answers online presents multifaceted challenges, primarily owing to the subjective nature of responses. The process is not only time-consuming but is also susceptible to fluctuations depending on the evaluator's mood, introducing an additional layer of unpredictability. The potential for evaluators to overlook certain nuances further exacerbates grading inconsistencies, as individual perspectives may vary significantly. This variation due to the unique perspectives of evaluators makes establishing consistent and objective grading criteria a formidable task, amplifying the complexities inherent in the assessment process. Moreover, the inherent subjectivity introduces the risk of biases, with evaluators potentially favoring specific styles or content structures based on their

personal inclinations. Additionally, the diverse lengths of responses compound the challenge of defining fair and uniform evaluation criteria, as different evaluators may have contrasting thresholds for what constitutes an acceptable length. The absence of contextual information compounds these challenges, hindering the system's ability to accurately assess and contextualize responses. Against this backdrop, the pressing need for swift and efficient evaluation further intensifies the strain on already constrained time and resources.

Devising a machine-driven system to assess subjective responses is pivotal for several compelling reasons:

Efficiency: Automated/Machine-driven systems excel in swiftly evaluating subjective answers, surpassing the speed of human evaluators by effectively processing a substantial influx of responses. This capability preserves significant resources and curtails the time needed to provide feedback or assign grades to students.

Consistency and Objectivity: Automated systems ensure a uniform and impartial assessment of subjective answers, free from biases or personal viewpoints. This commitment to objectivity assures equity and eradicates potential divergences stemming from human subjectivity.

Scalability: As the number of students or participants expands, the manual evaluation of subjective answers becomes progressively daunting and time-intensive. Automated systems showcase notable scalability, adeptly managing a large volume of responses without compromising on quality or speed.

Feedback and Learning: Automated evaluation systems expeditiously furnish feedback to students, aiding them in comprehending their strengths and weaknesses. This immediate feedback promotes performance enhancement and facilitates effective learning from errors.

Standardization: Automated systems rigorously adhere to predetermined evaluation criteria, ensuring unwavering and standardized assessment across varied evaluations and

evaluators. This dedication to standardization heightens the dependability and comparability of assessments, especially in educational settings involving multiple instructors or evaluators.

Cost-Effectiveness: Although the initial implementation of automated evaluation systems may involve some upfront investment and development costs, they ultimately prove cost-effective by obviating the need to recruit and train additional human evaluators. This cost-saving approach diminishes expenses linked to manual grading.

Accessibility and Availability: Automated evaluation systems remain accessible around the clock, permitting learners to submit responses at their convenience. This adaptability boosts accessibility, proving particularly advantageous in online learning environments or when participants are dispersed across different time zones.

Motivation

Advancements in AI, such as scene-based translation and word segmentation, pose challenges in extensive semantic analysis and manual grading of subjective questions. To address this, a model integrating pattern recognition, machine learning, and NLP is crucial for automated subjective answer evaluation. NLP, spanning syntax, semantics, and pragmatics, enables machines to understand and generate human-like text. Applications include chatbots, sentiment analysis, machine translation, and text classification. The synergy between NLP and AI, particularly with deep learning, promises to reshape daily interactions with intelligent systems.

The features of Natural Language Processing (NLP) contribute to text similarity in various ways:

1. Semantic Understanding:

Text Similarity Connection: By comprehending the meaning of words and phrases in context, NLP helps identify similarities in the semantic representation of different texts.

2. Syntax Analysis:

Text Similarity Connection: Analyzing the grammatical structure allows NLP to recognize similarities in the sentence structure between different texts.

3. Named Entity Recognition (NER):

Text Similarity Connection: NER aids in identifying common entities in texts, contributing to the recognition of similarities in terms of named entities.

4. Sentiment Analysis:

Text Similarity Connection: Sentiment analysis can identify similarities in the emotional tone or sentiment expressed in different texts.

5. Machine Translation:

Text Similarity Connection: NLP's ability to translate text between languages enables the identification of similar content in different languages.

6. Information Extraction:

Text Similarity Connection: Information extraction helps identify and compare specific data points, contributing to text similarity assessments.

8. Contextual Understanding:

Text Similarity Connection: Considering the broader context helps in recognizing similarities that may not be apparent from isolated words or phrases.

Objective

a) To devise a machine-driven evaluation model for questions with descriptive answers for quick and accurate assessments.

b) To improve the overall effectiveness, accuracy, and user satisfaction in the question-answer assessment process.

Outline of the project

1. Question set and model answer set preparing
2. Data collection
3. Manual labeling
4. Data pre-processing
5. Vectorization
6. Design of the model
7. Result and analysis

CHAPTER 2

Any machine-driven system requires an ample amount of data, collectively referred to as a dataset, to obtain a precise design of it with maximum accuracy. In this project, a dataset was made available by collecting answers from volunteers.

Data Collection Process

Step 1: In the preliminary phase, ten numbers of questions were chosen from various fields and topics. The question sets are shown in Figure 1.

Step 2: For each of the 10 questions selected, model answers were prepared with the utmost meticulous approach. This model answer set includes ten model answers for each question comprising five correct and five incorrect responses. The inclusion of both affirmative and negative examples enhances the model's ability to discern and evaluate a wide spectrum of possible answers, ensuring a more accurate assessment. The model answers are shown in Figure 2.

Step 3: A total of two hundred sample answers, as shown in Figure 1, were gathered for each of the ten questions. This extensive collection of responses provides a robust foundation for subsequent analysis and model training.

	A	B	C	D	E	F	G	H	I	J	K	L
1	1. What is Computer ?	2. What is Calcu	3. Why do we eat	4. What is Pollution ?	5. What is Interne	6. What is Photo	7. What is Therm	8. Why blood is red in	9. Why do we ne	10. What are the poles a magnet have ?		
2	A COMPUTER IS A ON	It is machine that	For stay alive.	Pollution is a degradat	Internet is a channe	Photosynthesis is	Thermometer is us	Blood is red because of I	For measuring so	Magnet have South and North pole.		
3	Computer is an electron	Calculator is an e	For live in the ecosy	Pollution is a ecologic	Internet is a data fe	Photosynthesis is	Measuring tempara	For RED BLOOD CELLS	To make a line	North and South		
4	A computer is a machin	A calculator is a d	A calculator is a de	Pollution is the introdu	The Internet, somet	Photosynthesis is	A thermometer is a	Within the red blood cells	Scales are often u	The north pole and south pole are the two poles of a ma		
5	an electronic device for	something used fi	To gain energy	the presence in or intr	A computer hardwa	the process by wh	A instrument to me	Because of hemoglobin	To measure distar	North pole, south pole		
6	A calculator	To perform mathe	To gain energy	Harmful effects on envi	A worldwide systen	a process by whic	A device to measu	Because of hemoglobin	To measure any c	North, South		
7	A electrical device whict	A device to perfor	To gain energy	The harmful effects on	A network which is	Plant's process of	Device to measure	It has chlorophyll	To measure distar	North & South		
8	A computer is a machin	A small electronic	To provide energy n	Pollution is the introdu	A global system of	Photosynthesis is	thermometer is an	Blood is red because of t	Scale is importan	The north pole and south pole		
9	A computer is a machin	something used fi	Base your meals or	Pollution is the introdu	The Internet (or inte	the process by wh	Don't know	Don't know	Don't know	Don't know		
10	A device which manipula	An electronic devi	For physical exister	A situation where harn	Interconnection of t	A process by plar	An instrument to m	Due to presence of RBC(	To measure heigh	Two		
11	Man made electroic ma	device to use for	For survive	bad elements excessiv	connects world.	its a method tree	device to measure	himogolobin prrsent in bl	to measure a leng	South north		
12	A computer is a piece o	anything that is u	to supply the body	the existence of somet	a local or restricted	the method by wh	a thermometer is a	Because hemoglobin, wh	Scales are freque	The north pole and the south pole are the two poles of a		
13	an electronic device for	It is used for mak	energy we need to l	Dust around us.	Internet is a global	the process by wh	An instrument for n	Blood is red because of t	To measure.	North & South		
14	Computer is a electronic	It's also Electroni	For getting energy fi	harmful chemicals mix	It's the global netwc	Basically process	Device to measure	Due to the himoglobin ins	To measure lengtl	North and South		
15	Computer is a device wh	Calculator is a de	We eat food for our	Pollution is disbalance	Internet is a source	Photosynthesis is	By which we calcu	Blood is red in color for p	We need a scale	There are two poles, North Pole and South Pole.		
16	an electronic device for	something used fi	We need food for nu	the presence in or intr	a global computer r	Photosynthesis is	Thermometer is a	For Himoglobin	For measurement	East and West		
17	A computer is a electror	Calculator is a de	We eat because we	The presence of harmfi	The internet is a ha	Photosynthesis is	Thermometer is an	Because thehemoglobin	To measure temp	Two poles of a magnet		
18	A computer is an electr	Calculator is usec	To get the strength	It is caused due to har	The free source fror	It's a process by w	It is used for meas	For haemoglobin	To measure a len	South and north pole		
19	A computer is used for	To calculate arith	For living and consu	Hazardus element that	Free source server	Process of food m	To measure body t	For haemoglobin	To measure lengtl	North and south pole		
20	It is a electronic device	It is used for arith	To get power and fo	Harmful particle preser	It is a free source d	Trees make food t	To measure body t	Due to haemoglobin	To measure any c	North and south pole		
21	A computer is a digital c	Calculator is a co	To live	The malifunctioning of	a global computer r	Photosynthesis is	A thermometer is a	Due to RBC	Scale is importan	East and west		
22	A computer is a device t	A calculator is a d	To provide energy n	Pollution is the introdu	The Internet, somet	Photosynthesis is	A thermometer is a	Human blood is red beca	Scales are often u	The two poles of a magnet are called the north pole and		
23	Computer is an electron	Calculator is an e	We eat food for ene	Pollution is the excess	Internet is the world	Photosynyhesis is	Thermometer is an	Blood is red in colour bec	We need a scale	Magnets have 2 poles, a North pole and a South Pole.		
24	Computer is a electronic	Calculator is a mi	We eat food so that	Pollution is polluting th	Internet is connecti	Photosynthesis is	Thermometer is us	Because blood has hemo	To measure some	North pole and west pole		
25	Computer is a thing whic	Calculator is a mi	We eat food to live.	Pollution is the thing w	A network of compu	process of making	thing to measure o	because of the presence	to measure some	North and South		
26	Device for storing and pr	Device for mather	For energy and repa	Harmful element prese	It a network for com	Conversion of gluc	Device to measure	The presence of haemog	To measure thing	East and west		
27	A computer is a device t	Calculator is a mi	We eat food becaus	introduction of harmful	Internet is a great w	Photosynthesis is	It's a device which	Blood contains Iron and I	We need a scale	A North Pole and a South Pole		
28	A computer is a device t	A calculator is a d	We eat because we	Pollution is a form of e	worldwide system c	plants use sunligh	measures tempera	For Haemoglobin	Measures Distanc	The north pole and south pole		
29	an electronic device tha	a device that can	for survival	presence of harmful ele	a large web of inter	the process by wh	a temperature mea	because blood has hemo	to measure lengt	two poles		
30	an electronic device for	something used fi	A well-balanced diet	the presence in or intr	a global computer r	the process by wh	an instrument for n	Within the red blood cells	These divisions ar	The north pole and south pole are the two poles of a ma		
31	An electronic device for	Something used f	To provide energy n	The presence in or intr	A global computer r	The process by w	An instrument for n	Blood is red because of t	Scales are often u	The north pole and south pole are the two poles of a ma		
32	Computer is an electron	Calculator is a sn	We eat food to alive	Pollution is the introdu	Internet is a global s	Photosynthesis is	Thermometer is an	Blood is red because of t	We need a scale	Magnet have the North Pole and the South Pole.		
33	Computer is a electronic	A small electronic	We eat food to prod	The introduction of hari	A world wide netwo	It is the process ir	A instrument which	RBC containts hemoglob	To measure anyth	South pole and north pole are the poles of magnet		
34	An electronic machine fo	A portable electro	It provides energy tc	Any undesirable chanç	A world wide global	The process by w	An instrument whic	Blood is red because of I	We need a scale	A magnet have two poles, south pole and north pole		
35	A computer is an electrc	A calculator is a d	We eat food to provi	Pollution refers to the ¡	The Internet is a glc	Photosynthesis is	A thermometer is a	Blood is red in color bec	A scale is used tc	Magnets have two poles, a north pole and a south pole.		

Figure 1: Data Collection Process

A	B
Question	**Answers**
1. What is Computer ?	A computer is a machine that can be programmed to carry out sequences of arithmetic or logical operations automatically.
	An electronic device for storing and processing data, typically in binary form, according to instructions given to it in a variable program.
	A computer is a device that accepts information (in the form of digitalized data) and manipulates it for some result based on a program, software, or
	A computer is an electronic device that manipulates information, or data. It has the ability to store, retrieve, and process data.
	A computer is a programmable device that stores, retrieves, and processes data.
	A computer is an electronic device that manipulates information or data, which has the ability to store, retrieve, and process data.
2. What is Calculator ?	A calculator is a device that performs arithmetic operations on numbers. Basic calculators can do only addition, subtraction, multiplication and divisi
	More sophisticated calculators can handle exponential operations, square roots, logarithms, trigonometric functions and hyperbolic functions. I
	An electronic calculator is typically a portable electronic device used to perform calculations, ranging from basic arithmetic to complex mathema
	A calculator is a small electronic device that you use for making mathematical calculations.
3. Why do we eat food ?	To provide energy needed to keep the body breathing and alive, for movement and warmth, and for growth and repair of tissues. Some starch a
	Food we eat helps our body in growth and development. Food is necessary to produce energy and meet our body's nutrient demands.
	The foods we eat provide us with a range of nutrients: vitamins, minerals, water, fat, carbohydrates, fibre, and protein.
	Food is essential to life because it provides the energy to run body functions and the building blocks to grow and repair body.
	Food provides the energy that our bodies need to keep going.
	We eat food to provide energy needed to keep the body breathing and alive, for movement, warmth and growth.
4. What is Pollution ?	Pollution is the introduction of harmful materials into the environment.
	Pollution is the introduction of contaminants into the natural environment that cause adverse change.
	Pollution happens when our environment is contaminated or dirtied, by waste, chemicals or other harmful substances.

Figure 2: Question and Model Answer

Manual Labeling

The collected sample answers were provided with their labels, based on the assessment done manually. The “already labeled data”, facilitates to create a core dataset that captures key classifications and patterns related to our project goals. The labeled data is shown in Figure 3.

2. What is Calculator ?		
1	It is machine that is use for calculate.	YES
2	Calculator is an electronic device	NO
3	A calculator is a device that performs arithmetic op	YES
4	something used for making mathematical calculati	YES
5	To perform mathematical operations	YES
6	A device to perform mathematical operations	YES
7	A small electronic machine used for calculating figu	NO
8	something used for making mathematical calculati	YES
9	An electronic device which helps to calculate.	YES
10	device to use for calutation	NO
11	anything that is used to calculate math problems, e	YES
12	It is used for making mathematical calculations, in	YES
13	something used for making mathematical calculati	YES
14	Something used for making mathematical calculati	YES
15	Calculator is a small electronic device for making n	YES
16	A small electronic device which is used for mathem	YES
17	A calculator is a software or hardware device that c	YES
18	What is a calculator? A calculator is a device that	YES
19	Calculator is a machine which is used to calculate	YES
20	Calculator is a device by which we can do mathem	YES

Figure 3: Manual Labeling

CHAPTER 3

Tools and Methods

We used Python Programming Language as our platform, and also we used Google Colab and Jupyter Notebook for our coding implementation. We used various Python libraries like Matplotlib, NumPy, Pandas etc.

Vectorization:

Step 1: Text Tokenization

The first step involves breaking down the text into smaller units, typically words or tokens. This process is known as tokenization and transforms the raw text into a structured format. e.g;
Sentence: "The cat in the hat." One-Hot Encoding:
"The": [1, 0, 0, 0]
"cat": [0, 1, 0, 0]
"in": [0, 0, 1, 0]
"the": [0, 0, 0, 1]
"hat": [0, 0, 0, 0]

Step 2: Building a Vocabulary

Create a vocabulary that encompasses all unique tokens present in the text. Each unique word/token is assigned a unique index or numerical identifier. This vocabulary serves as a reference for the vectorization process.

Types of Vectorization:

One-Hot Encoding

One-hot encoding is a simple vectorization technique where each word/token is represented as a binary vector. The vector has the same length as the vocabulary, with a 1 at the index corresponding to the word's position in the vocabulary and 0s elsewhere.[Figure 4]

Label Encoding

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50

→

One Hot Encoding

Apple	Chicken	Broccoli	Calories
1	0	0	95
0	1	0	231
0	0	1	50

Figure 4 : Example of One Hot Vectorization

Bag-of-Words (BoW) Representation

In the bag-of-words approach, the focus is on the frequency of each word in the text. A vector is created for each document, and each element in the vector represents the count of a specific word in the document.[Figure 5]

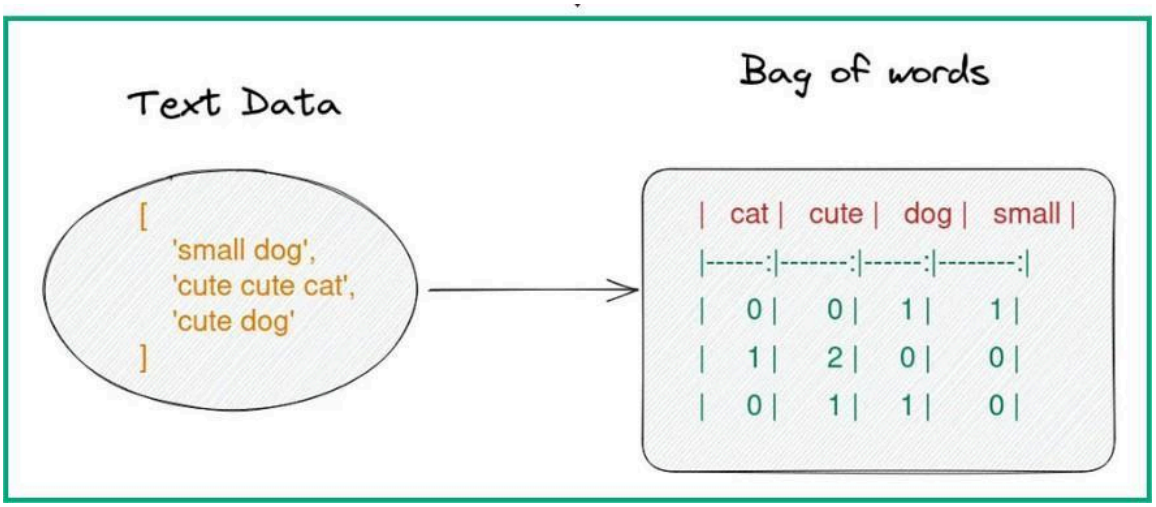


Figure 5 : Example of Bag Of Words

Word Embeddings

Word embeddings involve representing words as dense vectors in a continuous vector space. Techniques like Word2Vec, GloVe, or FastText are used to generate these embeddings, capturing semantic relationships between words. Shown in figure 6.

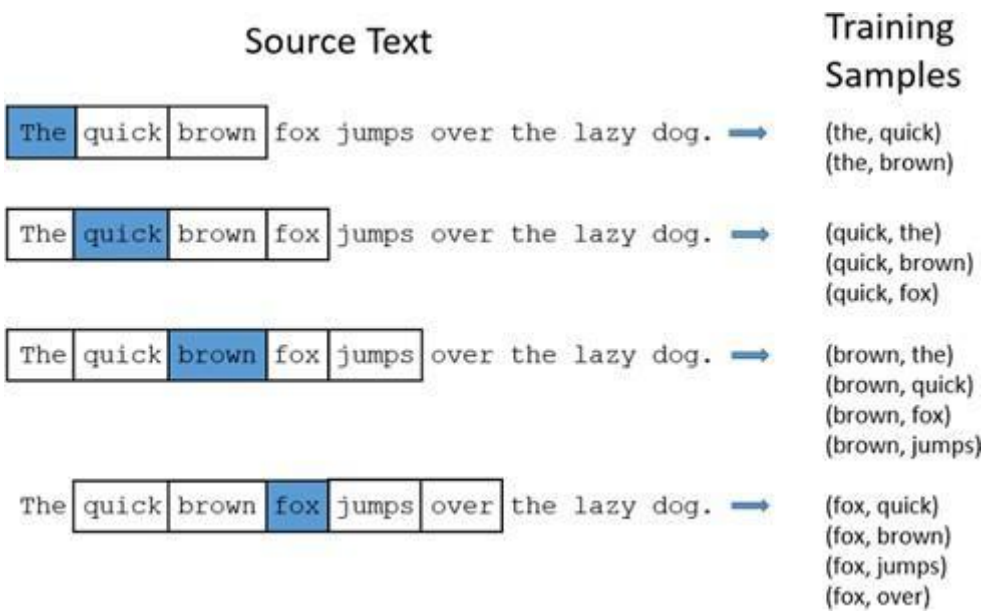


Figure 6 : Example of word2Vec

Step 9: Pre-trained Embeddings

Leveraging pre-trained embeddings, such as those from models like Word2Vec, GloVe, or BERT, allows for the use of contextually rich representations without the need for extensive training on specific datasets.

BERT:

BERT Model Overview:

BERT, an acronym for Bidirectional Encoder Representations from Transformers, represents a groundbreaking advancement in natural language processing (NLP) developed by Google. Departing from the sequential processing of traditional language

models, BERT employs a transformer architecture to comprehensively consider the context of each word bidirectionally within a sentence. Shown on figure 7.

Key Characteristics of BERT:

1. Bidirectional Context Understanding

- BERT's distinctive feature is its consideration of both left and right context for each word in a sentence. This bidirectional approach empowers BERT to capture the entirety of context and dependencies within the text, fostering a more nuanced understanding of language.

2. **Transformer Architecture:**

- Utilizing the transformer architecture, BERT excels in capturing long-range dependencies in sequential data. The architecture's highly parallelizable nature facilitates efficient training and processing, contributing to BERT's computational effectiveness.

3. **Pre-training on Large Corpora:**

- BERT undergoes pre-training on extensive datasets, encompassing portions of the internet and books. This pre-training equips BERT with the ability to acquire rich contextual representations, rendering it highly effective across a diverse array of downstream NLP tasks.

4. Contextual Word Embeddings:

- BERT's hallmark is the production of contextualized word embeddings. This means that the representation of a word varies based on its context within a specific sentence. This context-awareness significantly enhances BERT's proficiency in tasks demanding a profound understanding of context and semantics.

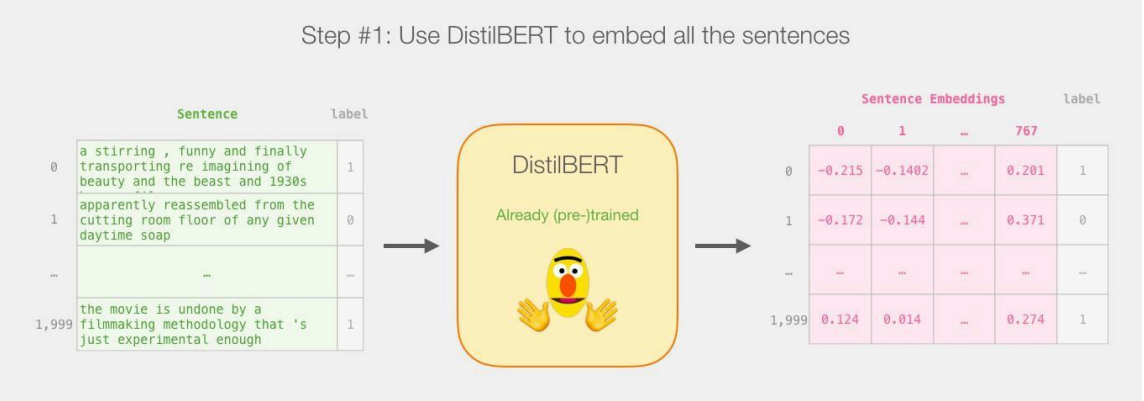


Figure 7: Example of BERT

Advantages of BERT for Small Data Set Training and Testing:

1. Transfer Learning:

- BERT's pre-trained representations serve as a form of transfer learning, where the model acquires general language features through extensive pre-training. This knowledge is subsequently fine-tuned on smaller, domain-specific datasets—a valuable strategy when dealing with limited labelled data.

2. Feature Extraction:

- BERT can be employed as a feature extractor by leveraging its pre-trained layers. This approach enables the extraction of high-quality embeddings for input sentences, proving advantageous for tasks with constraints on training data volume.

3. Effective Handling of Out-of-Vocabulary Words:

- BERT's subword tokenization mechanism adeptly manages out-of-vocabulary words by breaking them into subword tokens. This capability is particularly valuable when encountering domain-specific or rare words commonly found in small datasets.

4. Robustness to Noisy Data:

- Owing to its pre-training on diverse and extensive data, BERT demonstrates robustness to noisy or limited datasets. It retains the ability to generate meaningful representations even when the training dataset lacks extensive coverage.

5. Task Agnosticism:

- BERT's pre-trained representations exhibit task-agnostic qualities, fostering versatility across various NLP tasks. This adaptability proves especially advantageous when addressing diverse applications with small datasets.

Adaptability and Robustness of BERT:

BERT (Bidirectional Encoder Representations from Transformers): Excels in capturing bidirectional dependencies and contextual nuances, particularly suited for tasks demanding a deep understanding of the context within sentences.

Pre-training Advantage: BERT's pre-training on large corpora and transfer learning capabilities make it adaptable and robust, showcasing effective performance even in scenarios with limited training data.

Low training data size and related impact on using traditional Machine Learning Models:

Given the inherent challenge of training robust models with limited data, traditional neural networks, including CNNs, RNNs, and LSTMs, often face difficulties in generalizing patterns effectively. This limitation increases the likelihood of overfitting, where models memorize training data instead of learning to generalize to unseen instances.

To mitigate these issues, the adoption of few-shot learning proves crucial. Few-shot learning methods allow models to make accurate predictions even when provided with a minimal number of examples. This is particularly valuable in scenarios with sparse data, as few-shot learning enhances the model's ability to adapt and make informed decisions based on a limited dataset. By leveraging a smaller set of training instances, few-shot learning aids in overcoming the challenges posed by data scarcity, thereby improving the overall performance and robustness of models in low-data environments.

We decided to adopt the few-shot learning method because of this. Here are key reasons why few-shot learning is essential in such scenarios:

- **Data Scarcity:**In many domains, obtaining a large amount of labelled data is impractical or expensive. Few-shot learning allows models to be effective even with limited labelled examples.
- **Adaptability to New Domains:** When dealing with novel or niche domains where collecting extensive labelled data is challenging, few-shot learning enables the rapid adaptation of models to new and specific tasks.
- **Reduced Annotation Burden:** Manual annotation of data is resource-intensive. Few-shot learning reduces the burden of extensive labelling, making it feasible to train models with minimal labelled examples.
- **Quick Model Deployment:** Few-shot learning facilitates faster model development and deployment since it requires less time and effort in data collection and annotation.
- **Handling Evolving Data:** In dynamic environments where data distributions may change over time, few-shot learning allows models to adapt quickly to new patterns without requiring large retraining datasets.
- **Personalization and Customization:** Few-shot learning is valuable for personalized applications where training data specific to an individual or a small group may be limited, such as in personalized recommendation systems.
- **Transfer Learning:** Few-shot learning often aligns with transfer learning, where knowledge gained from one task with sufficient data can be transferred to related tasks with limited data.
- **Resource-Efficient Training:** Training models with few-shot learning are more resource-efficient as they require less computational power and storage compared to training on large datasets.

Few-shot:

Few-shot learning is a machine learning paradigm designed to address the challenge of training models when labeled data is scarce. In traditional machine learning, models often require substantial amounts of labeled data for effective training and generalization to new, unseen instances. However, in real-world scenarios, acquiring large labeled datasets can be impractical, time-consuming, or costly. This is where few-shot learning comes into play, offering a solution to train models with minimal labeled examples. Shown in Figure 8.

The term "few-shot" implies that the model is provided with only a small number of examples, or "shots," to learn and generalize patterns. This approach acknowledges that, in many situations, obtaining extensive labeled data for every class or task of interest is not feasible. Instead, few-shot learning enables models to achieve meaningful performance with just a handful of examples, making it a valuable strategy in scenarios where data is limited.

There are different settings within few-shot learning:

1. **One-shot learning:** In this scenario, the model is trained with just one example per class. This challenges the model to generalize effectively from a single instance, simulating situations where obtaining multiple examples is extremely challenging.

2. **Few-shot learning (K-shot learning):** This setting involves training the model with a small number (K) of examples per class, where K is typically a small integer. Although more data is available compared to one-shot learning, the focus is on learning from a limited dataset.

3. **Zero-shot learning:** Here, the model is expected to generalize to classes or tasks it has never encountered during training. This setting is particularly useful in situations where novel classes may emerge, and the model needs to adapt without the availability of labeled examples.

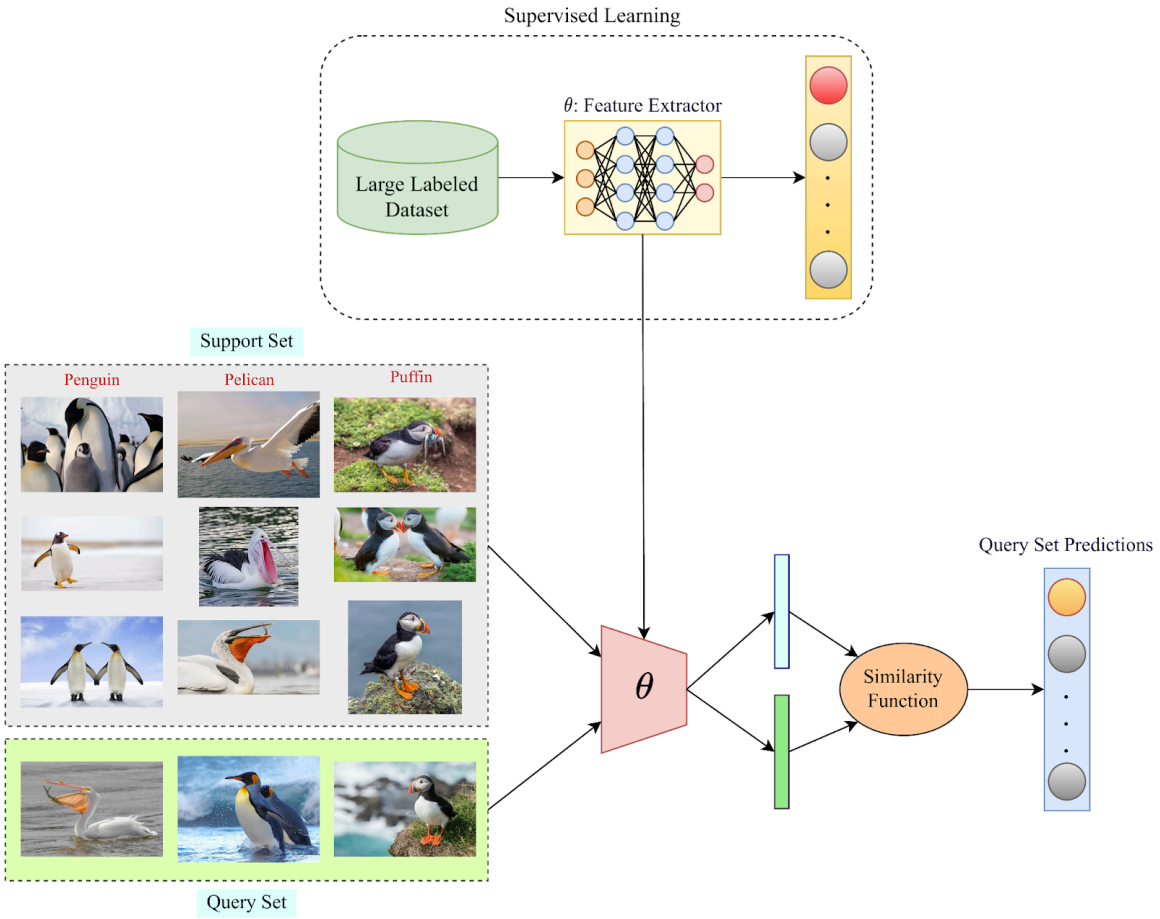


Figure 8: Illustration of Few-Shot Learning

CHAPTER 4

The Model:

The design of the proposed model is described here in a step-by-step manner. The block diagram of the proposed model is shown in Figure 9.

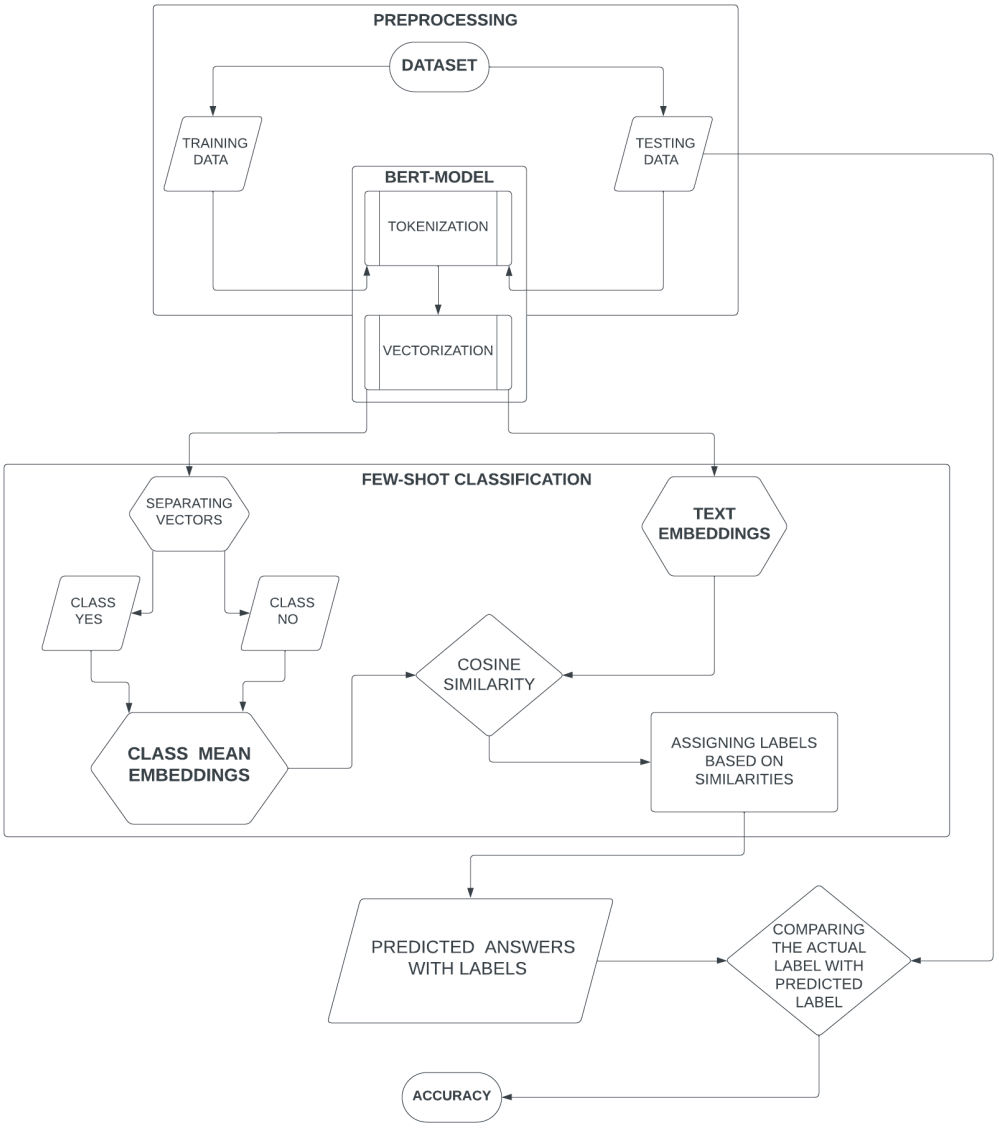


Figure:Process Block Diagram

Preprocessing

Step 1: Data Collection

Data is collected and labeled as described in Chapter 2 above.

Step 2: Tokenization

Tokenization is a fundamental process in natural language processing (NLP) that involves breaking down a text into smaller units called tokens. These tokens can be words, subwords, or even characters, depending on the granularity of the analysis. The primary goal of tokenization is to convert raw text into a format that is suitable for further analysis, enabling machines to understand and process human language. An illustration of tokenization is shown in Figure 10.

```
def get_bert_vector(texts, model_name='bert-base-uncased'):
    # Load pre-trained BERT model and tokenizer
    tokenizer = BertTokenizer.from_pretrained(model_name)
    model = BertModel.from_pretrained(model_name)

    # Tokenize the texts and convert to tensors
    inputs = tokenizer.encode_plus(texts, return_tensors='pt')
    print(inputs)
```

Figure 10: Tokenization Code

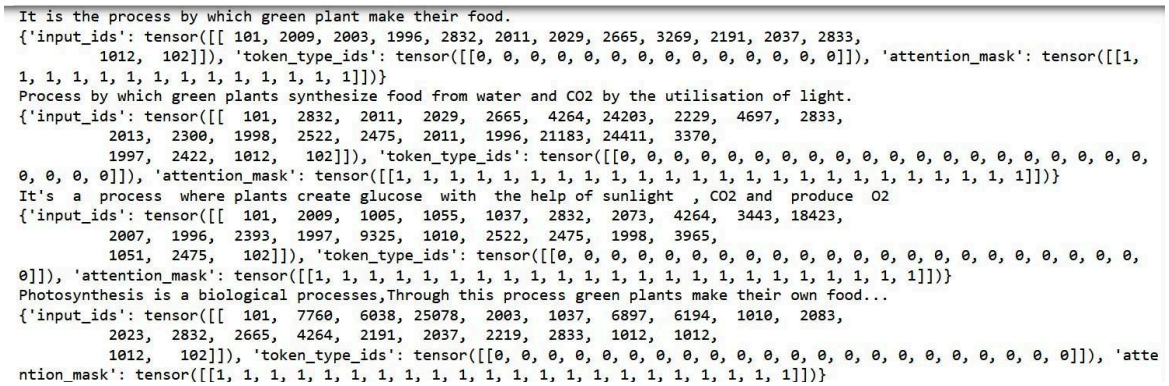


Figure 11: Tokenization Output

Vectorization

1. Uncased Model:

- The Uncased model preprocesses all text by converting it to lowercase, thereby eliminating case distinctions. This preprocessing step not only simplifies the model's vocabulary but also enhances generalization by treating words irrespective of their case. The shape of the vector in the **bert-base-uncased** model is **768.**

Example:

- The word "Embedding" in the Uncased model becomes "embedding," showcasing the case-insensitive transformation.

```python
def get_bert_vector(texts, model_name='bert-base-uncased'):
    # Load pre-trained BERT model and tokenizer
    tokenizer = BertTokenizer.from_pretrained(model_name)
    model = BertModel.from_pretrained(model_name)

    # Tokenize the texts and convert to tensors
    inputs = tokenizer.encode_plus(texts, return_tensors='pt')
    print(inputs)
    # Pass the inputs through the BERT model
    with torch.no_grad():
        outputs = model(**inputs)

    # Get the vector representations from the BERT model's output
    vector = outputs[0].mean(dim=1).squeeze()

    # Convert the tensor to a numpy array
    vector_np = vector.numpy()

    return vector_np
```

Figure 13: code for vectorization using uncased model

```
{'input_ids': tensor([[ 101, 7592,  102]]), 'token_type_ids': tensor([[0, 0, 0]]), 'attention_mask': tensor([[1, 1, 1]])}
[-1.55006588e-01  9.71007645e-02 -2.00396720e-02 -2.73763567e-01
 -2.30861023e-01 -2.15554416e-01  2.62849271e-01  1.02631994e-01
 -1.48881689e-01 -2.30088905e-01 -2.39837542e-01 -1.49520949e-01
  1.16972245e-01  6.56788573e-02 -3.25192958e-01 -3.18355829e-01
 -1.00932121e-01 -2.42468864e-02  1.93195306e-02  3.07083100e-01
 -7.26344585e-02 -6.88353106e-02  9.65527669e-02  1.63818017e-01
  1.16186321e-01 -4.07222472e-02 -1.90643668e-01  1.02965593e-01
 -1.94329932e-01 -2.15187117e-01 -2.01538488e-01 -5.63263297e-02
  8.44498500e-02  1.28897235e-01 -2.66287565e-01 -7.33948573e-02
  6.71947971e-02 -5.94936125e-02 -4.95762378e-01  2.63279885e-01
 -1.91475436e-01 -2.62324363e-01 -7.88278412e-03  6.66253045e-02
 -9.68066335e-04 -4.58025068e-01 -4.31555599e-01 -1.74565896e-01
 -2.72792101e-01  2.06725802e-02 -1.75076291e-01  2.03084454e-01
 -1.45856068e-01  3.50532204e-01  4.03149694e-01  1.13009244e-01
  4.90478873e-02 -2.97557384e-01 -1.41899332e-01  1.41044438e-01
 -1.78879797e-01 -4.05813046e-02 -1.94731560e-02 -1.69807673e-02
  1.60318241e-01  4.46129322e-01  2.10907385e-01 -6.26808405e-03
 -5.69692671e-01  3.97449851e-01 -1.78731874e-01 -3.96313101e-01
```

Figure 14: output for vectorization using uncased model

2. Paraphrase Model:

- The Paraphrase model is engineered with a specific focus on generating sentence embeddings that are close in vector space for paraphrased sentences. Its training objective revolves around understanding and representing the semantic similarity between sentences. The shape of the vector in the **paraphrase-bert-en** model is also **768.**

Example:

- When presented with two paraphrased sentences, the vectors produced by the Paraphrase model exhibit proximity, indicating their semantic equivalence. This makes

the Paraphrase model particularly valuable for tasks involving understanding and generating paraphrased content.

```python
def get_paraphrase_vector(sentence, model_name='bert-base-nli-stsb-mean-tokens'):
    # Load pre-trained paraphrase model
    model = SentenceTransformer(model_name)

    # Compute sentence embeddings
    sentence_embedding = model.encode(sentence, convert_to_tensor=True)

    # Convert the tensor to a numpy array
    vector_np = sentence_embedding.numpy()

    return vector_np
```

Figure 15: code for vectorization using paraphrase model

```
[-1.60809532e-01 -9.41228941e-02  1.18523546e-01 -1.14865743e-01
  1.42015100e-01 -2.42593691e-01 -2.13088542e-02  1.54481933e-01
 -3.96628141e-01 -1.36677340e-01 -4.87172604e-01 -2.06840515e-01
 -2.57251352e-01  8.58896151e-02 -1.73303232e-01 -1.69691011e-01
  5.22821359e-02 -4.06156294e-02 -2.83816963e-01  2.61466026e-01
  7.71800503e-02  1.79906294e-01  3.23327333e-01  5.25773585e-01
  1.36265278e-01  1.53845742e-01  1.48264961e-02  1.96557209e-01
 -8.85728061e-01 -4.55845267e-01 -3.68884474e-01 -3.94190997e-01
  4.28924114e-01 -2.02120379e-01 -3.84477586e-01  6.96020722e-02
 -4.83961916e-03  8.10381994e-02 -2.45132551e-01  2.00922012e-01
 -5.34854420e-02  6.65508136e-02 -1.65521219e-01  1.17833823e-01
 -3.34436566e-01  4.35489751e-02 -4.64393973e-01  2.84934878e-01
```

Figure 16: output for vectorization using paraphrase model

3. LM-V2 (Language Model Version 2) Model:

- The LM-V2 model represents an evolution of the original BERT model, featuring additional training on large corpora to enhance its language modelling capabilities. This extended training refines the model's understanding of language structures and nuances. The shape of the vector in the **paraphrase-MiniLM-L6-v2** model is **384.**

Example:

- LM-V2 demonstrates prowess in predicting missing words in a sentence, showcasing an advanced grasp of contextual language structure. This makes it well-suited for tasks that involve filling in gaps or generating coherent language sequences.

```python
def get_LMV2_vector(sentence, model_name='paraphrase-MiniLM-L6-v2'):
    # Load pre-trained paraphrase model
    model = SentenceTransformer(model_name)

    # Compute sentence embeddings
    sentence_embedding = model.encode(sentence, convert_to_tensor=True)

    # Convert the tensor to a numpy array
    vector_np = sentence_embedding.numpy()

    return vector_np
```

Figure 17: code for vectorization using LM-V2 model

```
[-2.72035897e-01  4.80246097e-01  5.22957921e-01  2.53585111e-02
 -3.37713361e-01 -3.82836819e-01  6.76632404e-01  2.44403277e-02
 -2.47440264e-01  3.00061315e-01  1.42444208e-01 -6.02886736e-01
 -3.19725931e-01 -2.51987308e-01  1.04954876e-01 -1.34700909e-01
  4.49402452e-01 -3.11045974e-01 -3.98145467e-01 -3.15191120e-01
 -3.57390881e-01  4.07207794e-02  3.51113267e-02  2.09172532e-01
 -9.61969327e-03 -1.97599128e-01  7.95240775e-02  5.84617138e-01
  3.23379755e-01 -3.68383795e-01  2.47820616e-01 -3.03901225e-01
  3.78505498e-01 -6.03871047e-03  1.36442721e-01  5.01969516e-01
  2.37175465e-01  7.46904835e-02 -1.95272639e-01 -3.22611444e-02
  2.84747154e-01 -4.40788954e-01  2.05442265e-01  3.62032056e-02
 -7.91357551e-03 -1.53565243e-01 -2.25167334e-01  2.17206314e-01
  3.89554501e-01  1.97616871e-02 -3.77070308e-02  3.28243412e-02
  2.98189726e-02  3.57046992e-01  6.61345065e-01  7.03640997e-01
 -2.25585520e-01  5.81988513e-01 -5.33910096e-02 -2.01934114e-01
 -3.30975540e-02 -2.77155310e-01 -3.50782752e-01  6.04857504e-01
 -9.14101601e-02 -1.80557176e-01 -4.48485374e-01  4.58674580e-01
 -6.16280854e-01 -7.16698885e-01 -2.93577194e-01  6.54614940e-02
 -3.61289412e-01  3.88383359e-01  1.05009437e-01  1.77215540e-03
  1.03037916e-01  2.31512025e-01  1.61027089e-01  3.25015992e-01
  2.27006435e-01  3.09595466e-01 -2.34726548e-01 -5.37918806e-01
  1.60860136e-01  1.85613230e-01  1.23873251e-02  1.00534119e-01
```

Figure 18: output for vectorization using LM-V2 model

Design of the Model

Step 5: Few-shot Learning.

Here, the machine-driven model is designed using Few-shot learning due to the smaller training data size. Followings are the step-by-step process involved in designing and training datasets for binary classification, specifically distinguishing between positive (YES) and negative (NO) responses.

1. Size of Training Data

- In Few-shot Learning, the deliberate choice is to work with a small labelled dataset. For our binary classification task, we carefully curated a dataset consisting of 5 positive (YES) sample answers and 5 negative (NO) sample answers. The sample answer is shown in figure 18.

This intentional constraint challenges the model to generalise effectively from a minimal set of examples. If the model answers are not given then the algorithm takes any five YES and NO from the dataset. The model code is shown in figure 19.

```
train_df_index = [55,33,54,37,86,71,109,96,92,101]
if len(train_df_index) > 0:
    train_df = df[df['No'].isin(train_df_index)]
    print(train_df)
else:
    train_yes = pd.read_csv(file)
    train_yes = train_yes[train_yes['Lebel'] == "YES"].sample(n=5, random_state=32)
    print(train_yes)
    train_no = df[df["Lebel"] == "NO"].sample(n=5, random_state=32)
    print(train_no)
    train_df = pd.concat([train_yes, train_no], ignore_index=True)
    print(train_df)
train_df_arr.append(train_df)
print(train_df_arr)
```

Figure 19: code model answers data with 5 yes and 5 no answers

	No	Answers	Lebel
32	33	It is the process by which green plant make th...	YES
36	37	Process by which green plants synthesize food ...	YES
53	54	It's a process where plants create glucose ...	YES
54	55	Photosynthesis is a biological processes,Throu...	YES
70	71	Photosynthesis is a process to create oxygen a...	NO
85	86	Photosynthesis is the process plants, algae an...	YES
91	92	It is a process by which tree produces oxygen ...	NO
95	96	In which way tree cook.	NO
100	101	Photosynthesis is a electric device used to ca...	NO
108	109	process of turning co2 and h20 into nutrients ...	NO

Figure 20: output of model data with 5 yes and 5 no answers

2. Two Centroid Creation:

- The next step involves creating centroids, representative points in the feature space, for the YES and NO classes. These centroids serve as pivotal reference points that encapsulate each class's key features. In our scenario, the YES centroid encapsulates features indicative of positive responses, while the NO centroid represents the characteristics of negative responses. The two centroid modeling is shown in figure 21 and figure 22.

```
def get_class_embeddings(train_df):
    few_shot_data = train_df.to_dict(orient="records")
    print(few_shot_data)
    class_embeddings = {"YES": [], "NO": []}
    class_counts = {"YES": 0, "NO": 0}
    for data in few_shot_data:
        input_text = data["Answers"]
        print(input_text)
        embeddings = get_bert_vector(input_text)
        # Append the embedding to the respective class
        class_embeddings[data["Lebel"]].append(embeddings)
        class_counts[data["Lebel"]] += 1
    return class_embeddings

def get_class_mean_embeddings(class_embeddings):
    class_mean_embeddings = {}
    for class_label, embeddings in class_embeddings.items():
        mean_embedding = np.mean(embeddings, axis=0)
        class_mean_embeddings[class_label] = mean_embedding
    return class_mean_embeddings
```

Figure 21: code of two centroids created with mean embeddings of model answers

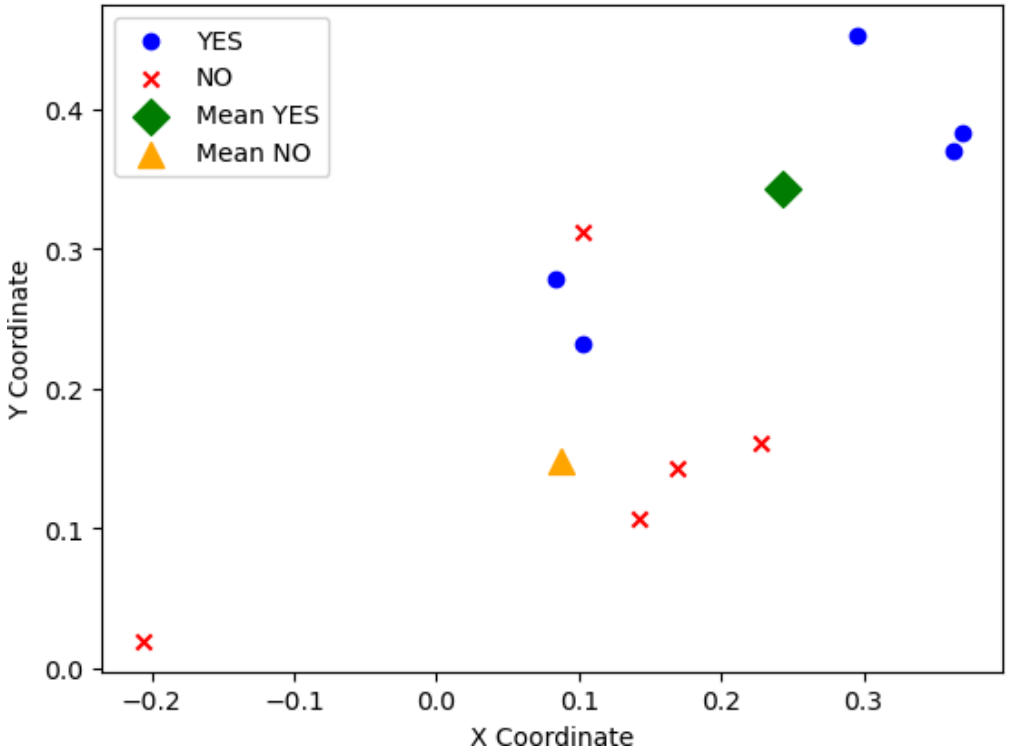


Figure 22: code of two centroids created with mean embeddings of model answers

3. Sample Match with Sample Data and Centroid using Cosine Similarity:

In the application of our model, the matching process between sample data and centroids is executed through the utilization of cosine similarity, a robust metric for assessing similarity between vectors. This methodology involves exposing the model to a constrained set of positive (YES) and negative (NO) sample answers, allowing it to discern essential patterns within each class. Subsequently, two centroids are formed, capturing the essential characteristics of their respective classes. When confronted with a new response, the model computes the cosine similarity between the response vector and both centroids. The classification decision is then determined based on the higher cosine similarity score, effectively assigning the response to either the YES or NO class. This systematic approach ensures that the model not only generalizes from a limited dataset but also adapts dynamically to new responses, making it particularly adept in Few-shot Learning scenarios where labeled data is scarce. The cosine similarity metric serves as a

critical tool in achieving nuanced and accurate classifications within this workflow. The sample matching code is shown in figure 23.

```python
def predict(df, class_embeddings):
    results = []
    class_mean_embeddings =  get_class_mean_embeddings(class_embeddings)
    print(df)
    scatter_plot(class_embeddings, class_mean_embeddings)
    test_answers = df.to_dict(orient="records")
    # print(test_answers)
    # Classify test data
    for test_example in test_answers:
        input_text = test_example["Answers"]
        test_embedding = get_bert_vector(input_text)
        # Calculate cosine similarity with class mean embeddings
        similarities = {}
        for class_label, mean_embedding in class_mean_embeddings.items():
            similarity = cosine_similarity([test_embedding], [mean_embedding])[0][0]
            similarities[class_label] = similarity

        # Predict the class with the highest cosine similarity
        predicted_class = max(similarities, key=similarities.get)
        #updating
        class_embeddings[predicted_class].append(test_embedding)
        scatter_plot(class_embeddings, class_mean_embeddings)
```

Figure 23: code for matching each embedding with centroid

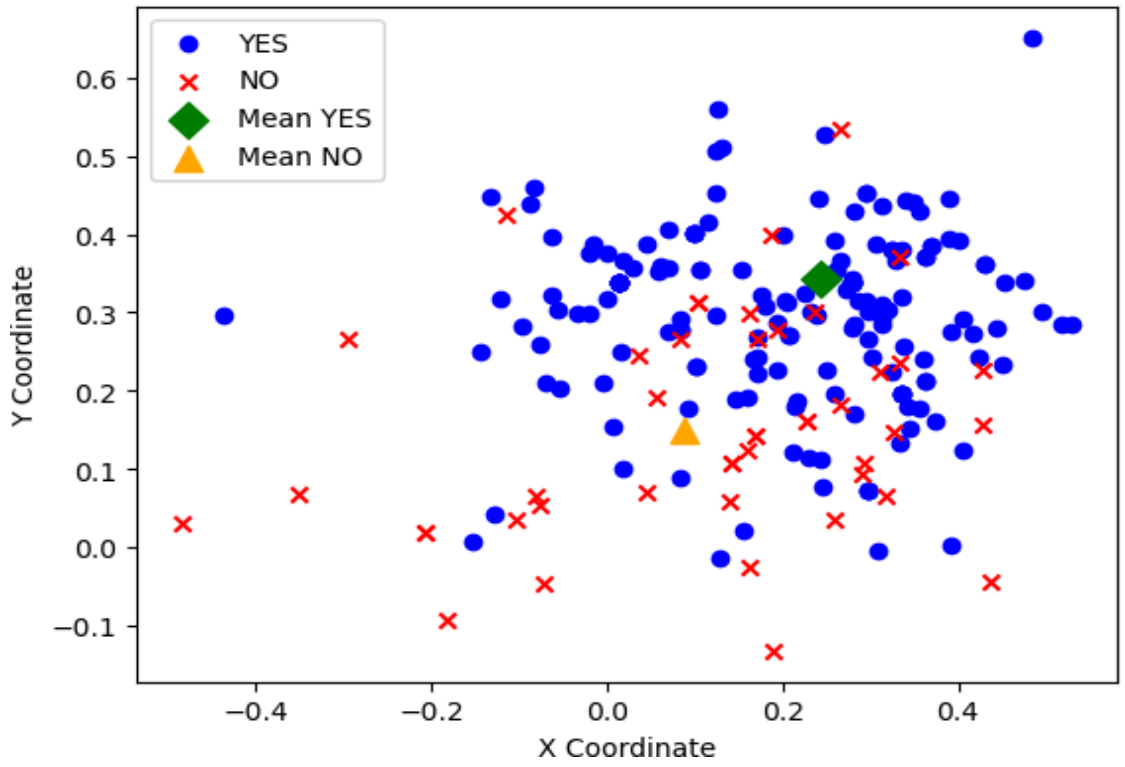


Figure 24: output for matching each embedding with centroids and plot them

Illustrative Example:

- **New Response:** "Absolutely"
- **Cosine Similarity (YES):** 0.87
- **Cosine Similarity (NO):** 0.45
- Classification Decision: The model classifies the response as a YES, as it exhibits higher similarity to the YES centroid.

CHAPTER 5

Following pivotal metrics are utilized in this project for analyzing and interpreting results:

1. Accuracy Score:

Definition: Accuracy measures the proportion of correctly classified instances among the total instances in the dataset. It provides a general overview of the model's correctness.

Application: Suitable for balanced datasets where classes are distributed relatively equally.

2. Confusion Matrix:

Definition: A confusion matrix is a table that outlines the model's performance by presenting the counts of true positive, true negative, false positive, and false negative predictions.

Components:

True Positive (TP): Instances correctly predicted as positive. ***True Negative (TN):*** Instances correctly predicted as negative. ***False Positive (FP):*** Instances incorrectly predicted as positive.

False Negative (FN): Instances incorrectly predicted as negative.
Application: Offers detailed insights into model performance, particularly in imbalanced datasets.

3. F1 Score: The F1 score is a metric commonly used in the field of machine learning and statistics to evaluate the performance of a classification model, particularly when dealing with imbalanced datasets. It is a balance between precision and recall, providing a single score that takes into account both false positives and false negatives.

The F1 score is calculated as the harmonic mean of precision and recall, where precision represents the accuracy of positive predictions among all predicted positives, and recall represents the proportion of actual positives correctly predicted by the model. The harmonic mean gives more weight to lower values, making the F1 score sensitive to both false positives and false negatives.

4. Precision:

Definition: Precision measures the accuracy of positive predictions by calculating the ratio of true positives to the sum of true positives and false positives.
Formula: Precision = TP / (TP + FP)
Application: Important when the cost of false positives is high.

5. Recall:

Definition: Recall, also known as sensitivity or true positive rate, measures the ability of a classification model to capture all the relevant instances (positives) in the dataset. It is calculated as the ratio of true positives to the sum of true positives and false negatives.
Formula: Recall = TP / (TP + FN)
Application: Recall is crucial when the cost of false negatives is high, and it is important to identify and capture as many true positives as possible.
For *example*, in a medical diagnosis scenario, high recall is essential to ensure that all actual cases of a disease are detected, even if it means accepting some false positives.

Result Report Summary

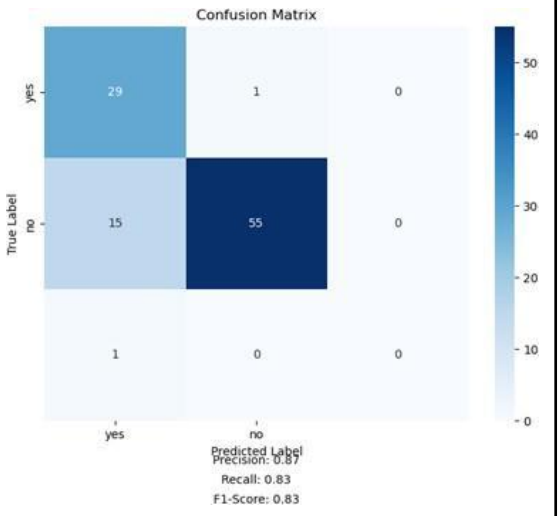
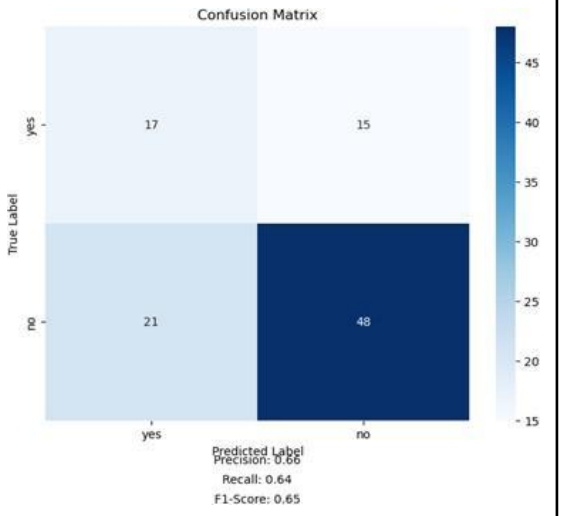
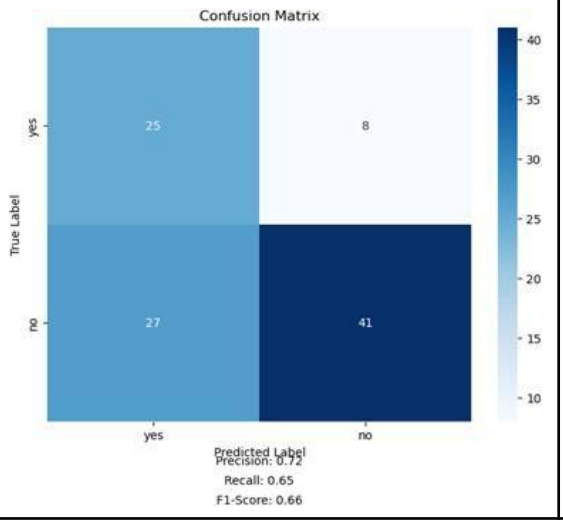
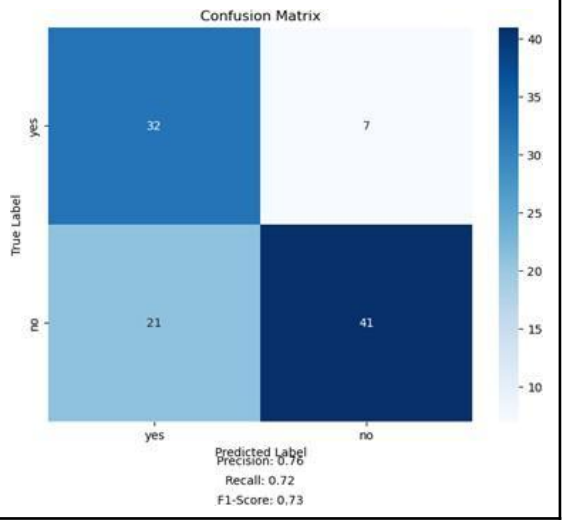
- Uncased model accuracy, F1 score, Precision in **table 1**
- Result of Uncased model confusion matrix in **table 2**
- paraphrase model accuracy, F1 score, Precision in **table 3**
- Result of Paraphrase model confusion matrix in **table 4**
- LM–V2 model accuracy, F1 score, Precision in **table 5**
- Result of LM-V2 confusion matrix in **table 6**

Result Based on uncased model:

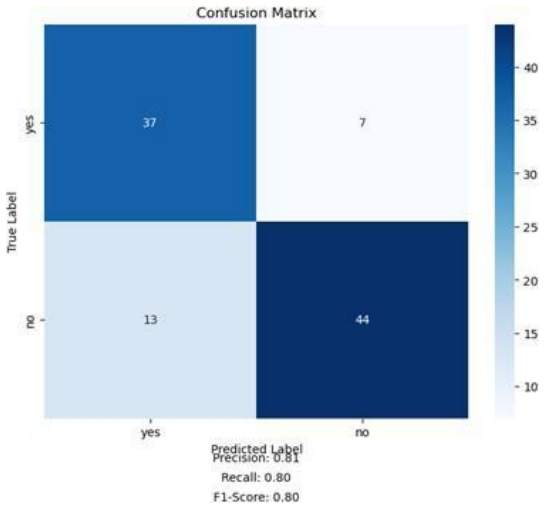
Table 1

Questions	Accuracy	F1 score	Precision
1. What is a Computer?	84.0	0.83476	0.87211
2. What is a Calculator?	48.51	0.65073	0.66224
3. Why do we eat food ?	65.34	0.66405	0.72042
4. What is Pollution?	72.27	0.72622	0.75748
5. What is the Internet?	80.19	0.80279	0.80927
6. What is Photosynthesis?	89.99	0.78221	0.90763
7. What is a Thermometer?	64.35	0.67068	0.82564
8. Why blood is red in colour?	60.39	0.66573	0.90285
9. Why do we need a scale?	59.40	0.62095	0.68601
10. What are the poles a magnet has?	63.33	0.66409	0.75842

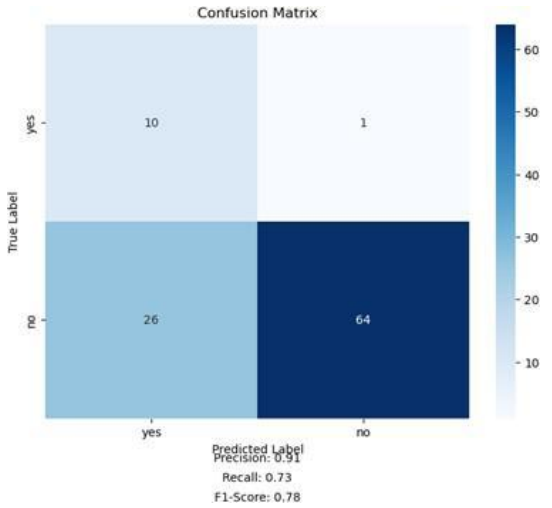
Table 2

Q1	**Q2**
	
Q3	**Q4**
	

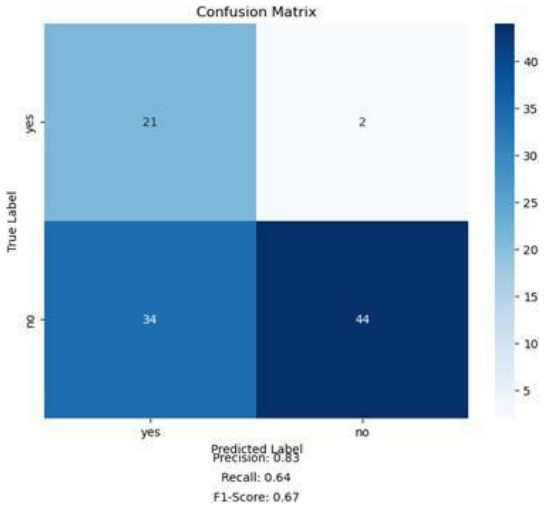
Q5


Confusion Matrix
True Label
yes
no
37
7
13
44
yes
no
Predicted Label
Precision: 0.81
Recall: 0.80
F1-Score: 0.80

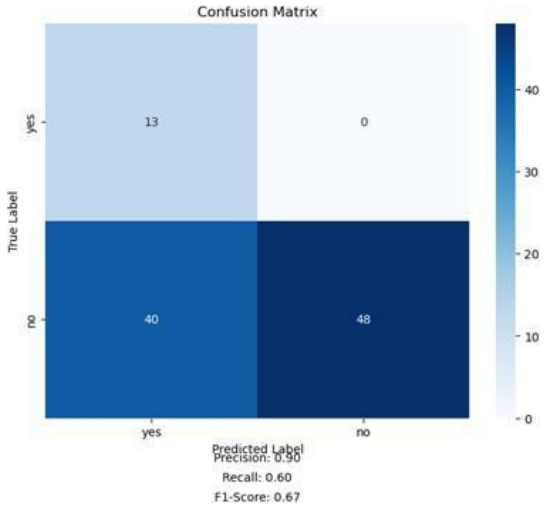
Q6

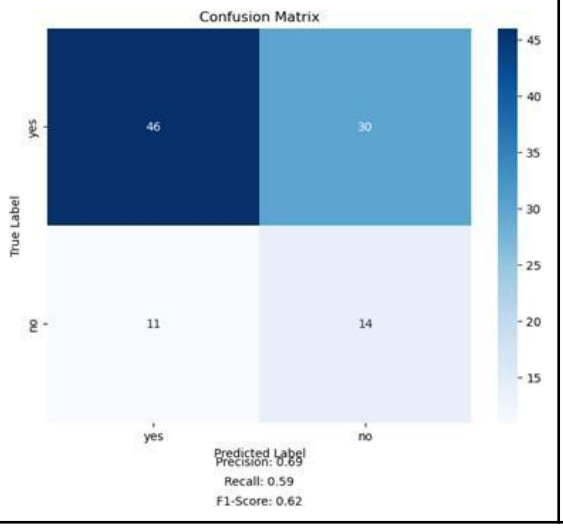
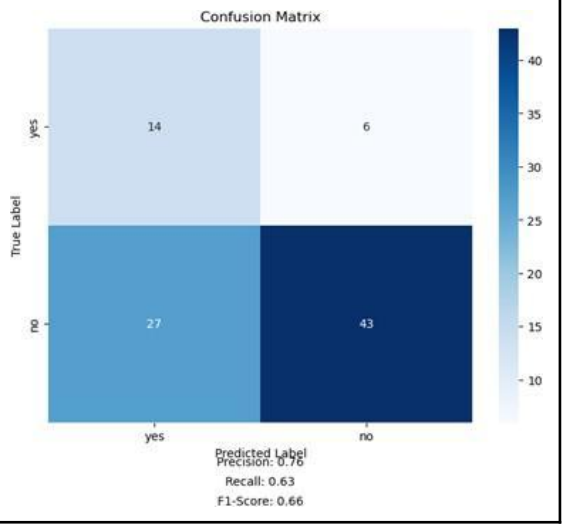

Confusion Matrix
True Label
yes
no
10
1
26
64
yes
no
Predicted Label
Precision: 0.91
Recall: 0.73
F1-Score: 0.78

Q7


Confusion Matrix
True Label
yes
no
21
2
34
44
yes
no
Predicted Label
Precision: 0.83
Recall: 0.64
F1-Score: 0.67

Q8


Confusion Matrix
True Label
yes
no
13
0
40
48
yes
no
Predicted Label
Precision: 0.90
Recall: 0.60
F1-Score: 0.67

Q9	Q10
	

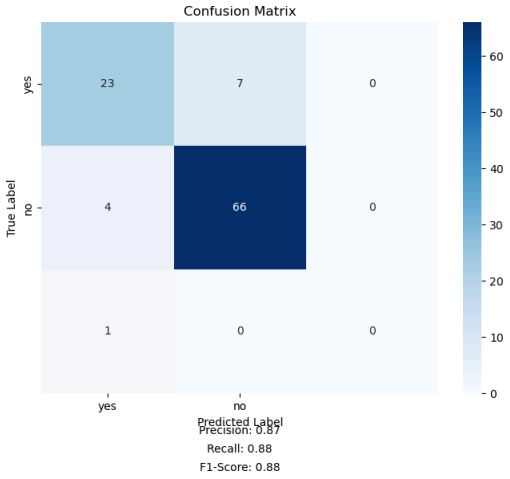
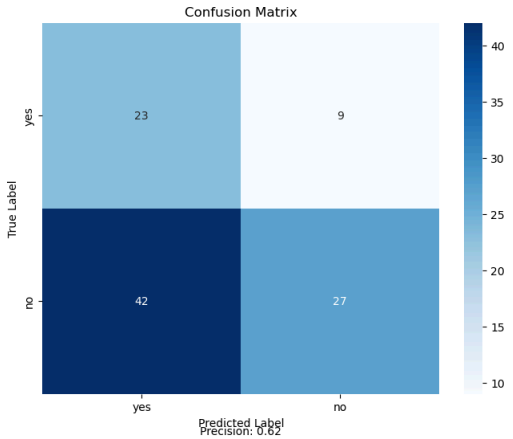
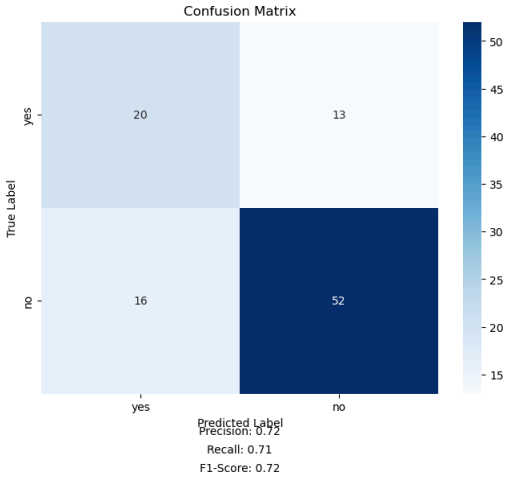
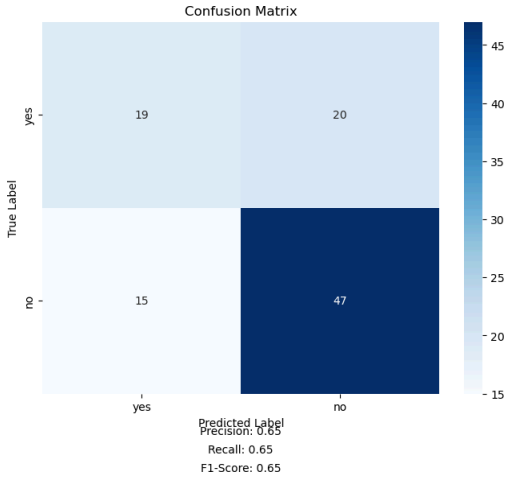
Result Based on Paraphrase model:

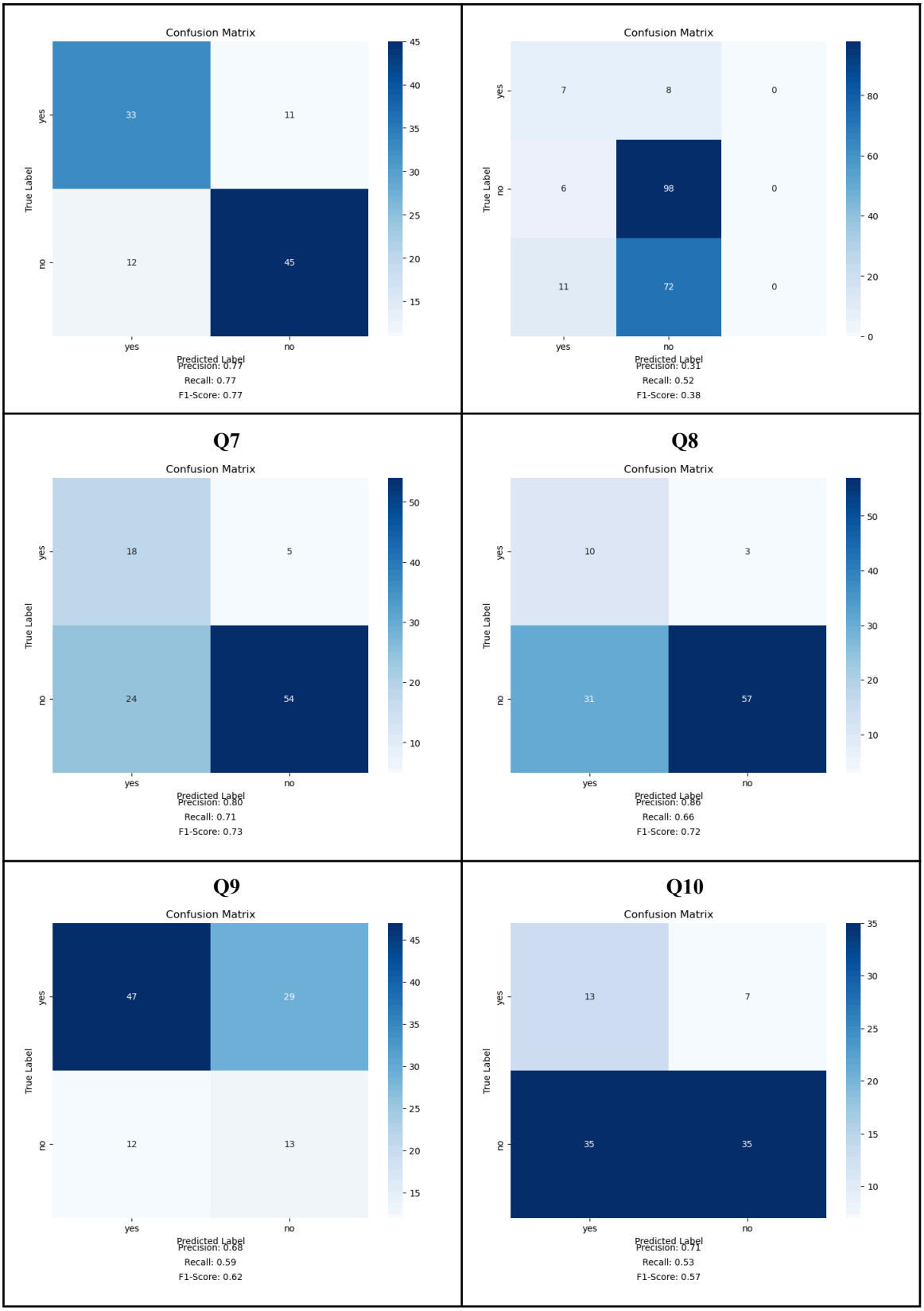
Table 3

Question	Accuracy	F1 score	Precision
1. What is a Computer?	89.0	0.87533	0.87059

2. What is a Calculator?	49.50	0.50159	0.62448
3. Why do we eat food ?	71.28	0.71587	0.72013
4. What is Pollution?	65.34	0.64831	0.64640
5. What is the Internet?	77.22	0.77254	0.77297
6. What is Photosynthesis?	87.15	0.38449	0.30511
7. What is a Thermometer?	71.28	0.73492	0.80442
8. Why blood is red in colour?	66.33	0.71879	0.85911
9. Why do we need a scale?	59.40	0.62000	0.67604
10. What are the poles a magnet has?	53.33	0.57107	0.70833

Table 4

Q1  Precision: 0.87 Recall: 0.88 F1-Score: 0.88	**Q2**  Precision: 0.62 Recall: 0.50 F1-Score: 0.50
Q3  Precision: 0.72 Recall: 0.71 F1-Score: 0.72	**Q4**  Precision: 0.65 Recall: 0.65 F1-Score: 0.65
Q5	**Q6**


Confusion Matrix
True Label
yes
no
33
11
12
45
Predicted Label
Precision: 0.77
Recall: 0.77
F1-Score: 0.77
Confusion Matrix
True Label
yes
no
7
8
0
6
98
0
11
72
0
Predicted Label
Precision: 0.31
Recall: 0.52
F1-Score: 0.38
Q7
Confusion Matrix
True Label
yes
no
18
5
24
54
Predicted Label
Precision: 0.80
Recall: 0.71
F1-Score: 0.73
Q8
Confusion Matrix
True Label
yes
no
10
3
31
57
Predicted Label
Precision: 0.86
Recall: 0.66
F1-Score: 0.72
Q9
Confusion Matrix
True Label
yes
no
47
29
12
13
Predicted Label
Precision: 0.68
Recall: 0.59
F1-Score: 0.62
Q10
Confusion Matrix
True Label
yes
no
13
7
35
35
Predicted Label
Precision: 0.71
Recall: 0.53
F1-Score: 0.57

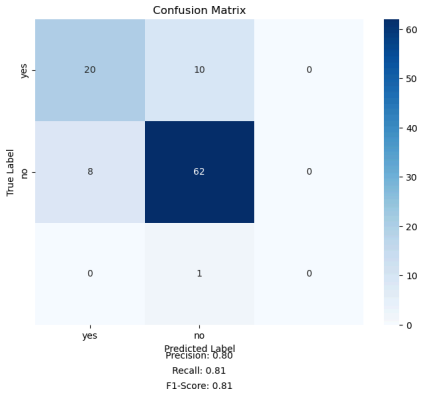
Result Based on LM-V2 model:

Table 5

Question	Accuracy	F1 score	Precision
1. What is a Computer?	82.0	0.80583	0.80079
2. What is a Calculator?	66.33	0.67296	0.69509
3. Why do we eat food ?	74.25	0.74041	0.73886
4. What is Pollution?	73.26	0.73568	0.74568
5. What is the Internet?	69.30	0.69158	0.69134
6. What is Photosynthesis?	75.22	0.79522	0.88248
7. What is a Thermometer?	73.26	0.74404	0.76218
8. Why blood is red in colour?	46.53	0.54154	0.80794
9. Why do we need a scale?	62.37	0.61222	0.60209
10. What are the poles a magnet has?	83.33	0.83950	0.85128

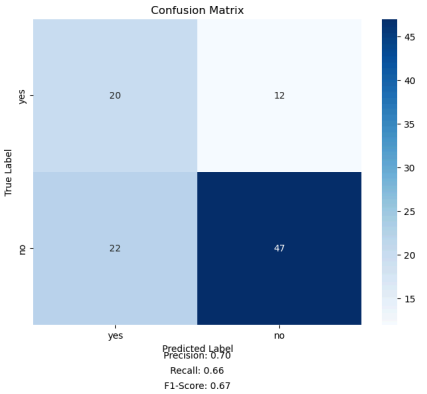
Table 6

Q1



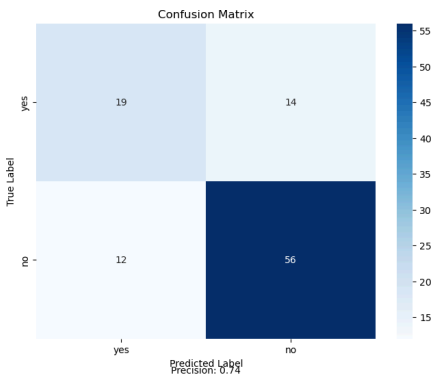
Precision: 0.80
Recall: 0.81
F1-Score: 0.81

Q2



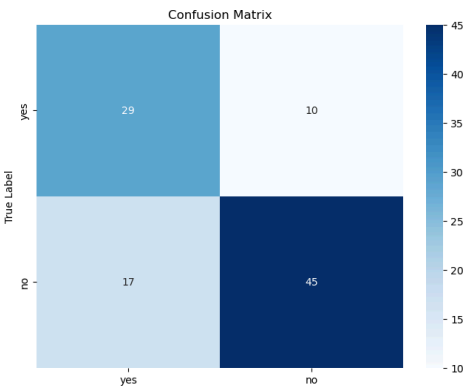
Precision: 0.70
Recall: 0.66
F1-Score: 0.67

Q3



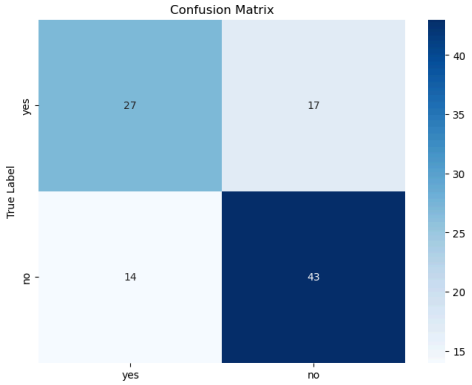
Precision: 0.74
Recall: 0.74
F1-Score: 0.74

Q4



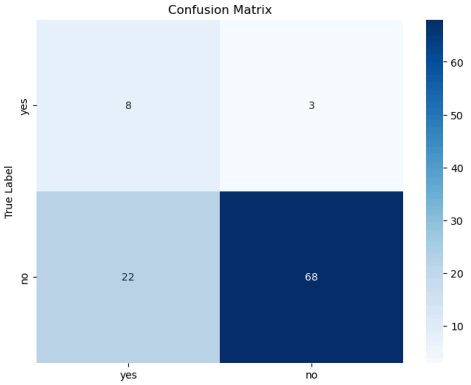
Precision: 0.75
Recall: 0.73
F1-Score: 0.74

Q5



Precision: 0.69
Recall: 0.69
F1-Score: 0.69

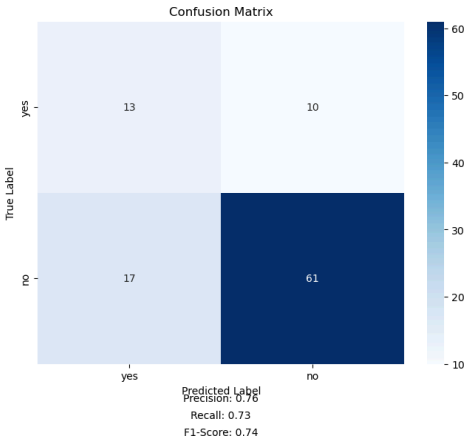
Q6

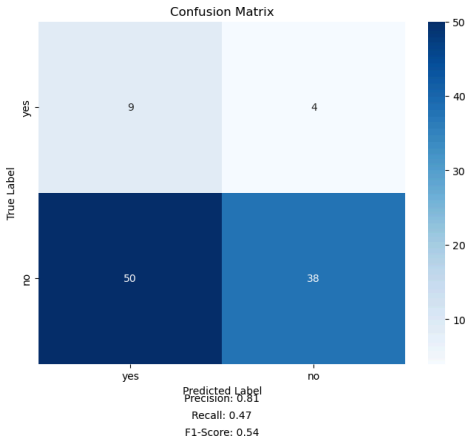


Precision: 0.88
Recall: 0.75
F1-Score: 0.80

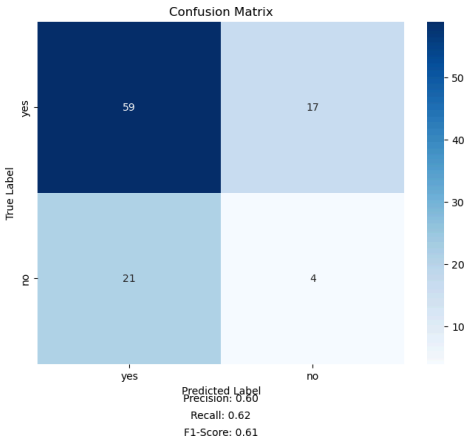
Q7

Q8

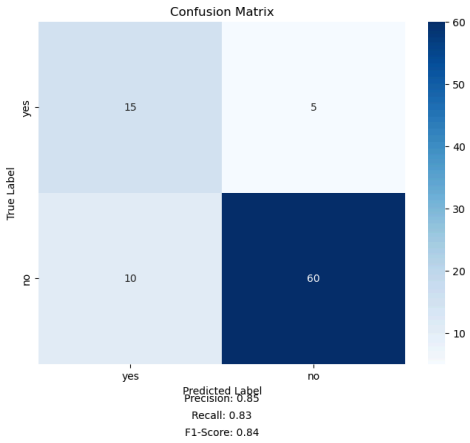

Confusion Matrix
13
10
17
61
yes
no
True Label
Predicted Label
Precision: 0.76
Recall: 0.73
F1-Score: 0.74


Confusion Matrix
9
4
50
38
yes
no
True Label
Predicted Label
Precision: 0.81
Recall: 0.47
F1-Score: 0.54

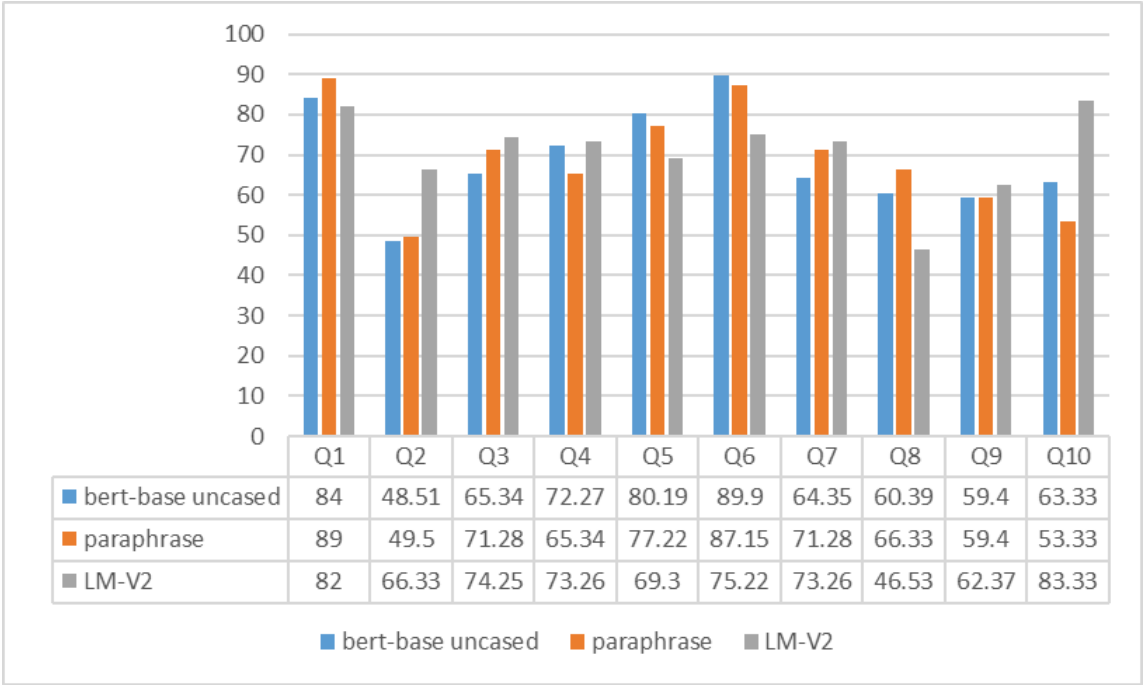
Q9


Confusion Matrix
59
17
21
4
yes
no
True Label
Predicted Label
Precision: 0.60
Recall: 0.62
F1-Score: 0.61

Q10


Confusion Matrix
15
5
10
60
yes
no
True Label
Predicted Label
Precision: 0.85
Recall: 0.83
F1-Score: 0.84

Comparison of Accuracy among the Models:



	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
bert-base uncased	84	48.51	65.34	72.27	80.19	89.9	64.35	60.39	59.4	63.33
paraphrase	89	49.5	71.28	65.34	77.22	87.15	71.28	66.33	59.4	53.33
LM-V2	82	66.33	74.25	73.26	69.3	75.22	73.26	46.53	62.37	83.33

Conclusion:

1. **Quality and Dependability:** There is a potential for confusion-induced mislabeling, stemming from human errors that may occur during the labelling process. It is crucial to exercise heightened attention to detail during manual labelling, as oversights or mistakes could compromise the accuracy and dependability of the labelled dataset.

2. **Inconsistencies:** The presence of variations in identical responses introduces confusion in the labelling process, diminishing the standard of assessment, particularly in the case of descriptive answers. This lack of uniformity may result in subjective grading, introducing elements of unfairness or bias in the overall evaluation.

3. **Challenges in Scalability:** Effectively extending manual evaluation to manage a sizable dataset poses a formidable challenge. With an increasing volume of answers, maintaining consistent and precise assessment across all responses becomes progressively more difficult.

4. **Fatigue and Oversight:** The process of labelling is not immune to instances of fatigue or cognitive biases, especially when confronted with a substantial number of descriptive answers. This fatigue can lead to a reduction in attention to detail, elevating the likelihood of errors or oversights during the evaluation phase.

Future Scope:

1. **Diversification of Dataset:** Broadening the dataset to encompass a wide array of answer scripts holds the potential to elevate the system's efficacy and precision. Enriching the dataset with diverse student responses allows the system to adeptly handle variances in writing styles, content, and linguistic proficiency, thereby fortifying its capacity for comprehensive evaluations.

2. **Dynamic Optimization of Word Embedding Models:** The automatic fine-tuning of word2vec models stands as a key avenue for refining the vectorization process. Through the dynamic adjustment of model parameters and hyperparameters tailored to the nuances of answer script evaluation, this approach fosters improved vector representations. This refinement captures pertinent semantic information more adeptly, fostering heightened accuracy in evaluations and bolstering overall system performance.

3. **Subject-specific Evaluation Frameworks:** Crafting future systems with the capacity for subject-specific adaptations ensures a nuanced evaluation process. Tailoring evaluation criteria and language models to specific subjects enables more precise and contextually relevant assessments of descriptive answers.

4. **Progress in Natural Language Processing (NLP):** Ongoing advancements in NLP methodologies continue to shape the landscape of accurate understanding and evaluation of descriptive responses. Evolving language models facilitate a more nuanced grasp of meaning, coherence, and relevance in student answers.

5. **Incorporation of Language Transformers:** The integration of advanced language transformers, exemplified by technologies like GPT-3, holds promise for refining the evaluation paradigm. These transformers contribute to more accurate, context-aware assessments and can generate insightful feedback, enhancing the overall quality of evaluations.

6. **Embedding Plagiarism Detection Features:** Augmenting automatic evaluation systems with robust plagiarism detection capabilities serves as a pivotal step in upholding the authenticity and integrity of assessments. This integration safeguards against instances of copied content or academic misconduct, ensuring the credibility of the evaluation process.

Bibliography:

[1]. *Jha, S., & Hämäläinen, W. (2016). Automated Short Answer Grading: A Survey of the State-of-the-Art. IEEE Transactions on Learning Technologies, 9(3), 197-205.*

[2]. *Gupta, M., & Agarwal, P. (2019). Automated Evaluation of Student Answers Using Natural Language Processing. In 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 400-405). IEEE.*

[3]. *Bhattacharya, A., Mitra, R., & Dutta, A. (2017). Automated Evaluation of English Answers for Educational Purposes: A Review. In Proceedings of the 10th International Conference on Theory and Practice of Electronic Governance (pp. 342-351). ACM.*

[4]. *Panda, B. K., & Kankar, P. K. (2018). Automated Assessment of Descriptive Answers for MOOCs Using Natural Language Processing. In 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI) (pp. 1263-1267). IEEE.*

[5]. *Yau, Y. H., & Joy, M. (2019). Automated Short Answer Grading: A Review. In Proceedings of the 2019 3rd International Conference on Cloud and Big Data Computing (pp. 167-171). ACM.*

[6]. *NLP-based Automatic Answer Script Evaluation, Md. Motiur Rahman1 and Fazlul Hasan Siddiqui2*

[7]*. Automatic Exam Correction Framework (AECF) for the MCQs, Essays, and Equations MatchingnHOSSAM MAGDY BALAHA AND MAHMOUD M. SAAFAN Computers and Systems Engineering Department, Faculty of Engineering, Mansoura University, Mansoura 35516, Egypt Corresponding author: Hossam Magdy Balaha (hossam.m.balaha@mans.edu.eg)*

[8]. *Design and Development of a Framework for an Automatic Answer Evaluation System Based on Similarity Measures Madhumitha Ramamurthy ORCID logo EMAIL logo and Ilango Krishnamurthi From the journal Journal of Intelligent System https://doi.org/10.1515/jisys-2015-0031*

[9]. *FEW-SHOT TEXT CLASSIFICATION WITH DISTRIBUTIONAL SIGNATURES Yujia Bao, Menghua Wu, Shiyu Chang, Regina Barzilay Computer Science and Artificial Intelligence Lab, MIT, MIT-IBM Watson AI LAB, IBM Research*