Evaluation of Subjective Answer using Semantic Similarity

Project Team

Abdullah Bin Nasser	p17-6111
Muhammad Raza Ali	p17-6108
Muhammad Arslan Ahmad Khan	p17-6089

Session 2017-2021

Supervised by

Ms. Sara Rehmat

Department of Computer Science

National University of Computer and Emerging Sciences
Peshawar, Pakistan

June, 2021

Student's Declaration

We declare that this project titled "*Evaluation of Subjective Answer using Semantic Similarity*", submitted as requirement for the award of degree of Bachelors in Computer Science, does not contain any material previously submitted for a degree in any university; and that to the best of our knowledge, it does not contain any materials previously published or written by another person except where due reference is made in the text.

We understand that the management of Department of Computer Science, National University of Computer and Emerging Sciences, has a zero tolerance policy towards plagiarism. Therefore, We, as authors of the above-mentioned thesis, solemnly declare that no portion of our thesis has been plagiarized and any material used in the thesis from other sources is properly referenced.

We further understand that if we are found guilty of any form of plagiarism in the thesis work even after graduation, the University reserves the right to revoke our BS degree.

Abdullah Bin Nasser Signature: _____

Muhammad Raza Ali Signature: _____

Muhammad Arslan Ahmad Khan Signature: _____

Verified by Plagiarism Cell Officer
Dated:

Certificate of Approval

The Department of Computer Science, National University of Computer and Emerging Sciences, accepts this thesis titled *Evaluation of Subjective Answer using Semantic Similarity*, submitted by Abdullah Bin Nasser (p17-6111), Muhammad Raza Ali (p17-6108), and Muhammad Arslan Ahmad Khan (p17-6089), in its current form, and it is satisfying the dissertation requirements for the award of Bachelors Degree in Computer Science.

Supervisor

Ms. Sara Rehmat Signature: _____

Ms. Mashal Khan

FYP Coordinator
National University of Computer and Emerging Sciences, Peshawar

---

Dr. Hafeez-Ur-Rehman

HoD of Department of Computer Science
National University of Computer and Emerging Sciences

Acknowledgements

We are very thankful to Almighty Allah who is the most Merciful and the most Beneficent for giving us strength and courage to choose such challenging project and for completing this project with great dignity and grace. Lots of people played a significant role in the completion of this project. We firstly say thanks to Ms . Sara Rehmat , who motivated us to select this project. She encouraged and showed us the way the project can be implemented. She has been very kind, supportive and patient to us since the beginning of our project. Her guidance and experience brought us on the right track whenever we faced any difficulty. We are highly thankful to all of our teachers who had been guiding us throughout our project work. Their knowledge, guidance, and training enabled us to carry out this development work more efficiently. We would like to thank all our friends and family for their support, motivation, and suggestions about the project.

Abdullah Bin Nasser

Muhammad Raza Ali

Muhammad Arslan Ahmad Khan

Abstract

Evaluation of subjective answers plays a very important role in examination. Many techniques exist for the evaluation of subjective answers. But checking of subjective answers requires understanding the context and the meaning of the words. Two answers, conveying same idea, can be written quite differently. The checking of subjective answers requires human intelligence and effort. We propose an automatic answer checker application that checks and marks written answers in a way similar to human being. This software application is built to check subjective answers and allocate marks to the user after verifying the answer. Both the answers need not to be exactly the same. The system based on the artificial intelligence will verify the answers and allocate marks accordingly as good as a human being. In the end, the evaluation is done to prove that our approach gives much realistic behavior and can be used in the examination.

Contents

List of Figures

Chapter 1

Preliminaries and Introduction

Analyzing research papers and working on different techniques had helped us to develop such an application which plays an important part in evaluating subjective answers. Many kinds of researches have been made to make an Evaluation of Subjective Answer(ESA)[1] looks more accurate. But besides all the researches that have been done, there have not been a single techniques that have been considered as an ideal technique for the evaluation. There are a lot of issue that have been solved by this kind of software. Also, there are other challenges like developing an algorithm that produces checking good precision between two answers of same question.Most of our evaluation is done by using Bert model technique and Bi-LSTM. There are a lot of techniques like Bag-of-Words[2] and Word2Vec[3] which is uni-directional and does not analyze the context of the answers.

1.1 Motivation

- Evaluation of context of answers is important part of grading process.
- In our education system the one who writes the most,gets the most marks.
- Mostly the automated judgmental tools are based on objectivity.
- The most valuable and time taking activity in education is evaluation.

1.2 Objectives

The primary objective is to develop a method for Evaluation of Subjective Answers. In order to complete all this, a number of primary objectives should be fulfilled.

- Auto evaluate subjective answer.
- Unbiased checking
- To make machine handle and reduce the chances of wrong evaluation.
- Understanding variations in answer.
- Training model on real data.
- Generating different answers.
- Understanding how real-world problems can be solved using computer science.

1.3 Scope

- Auto-Evaluation of answers is a big task so to gain maximum accuracy we have only considered one course for this project i.e. Software engineering.
- In this course only the questions having close-ended answers are considered.
- The reason not to choose open-ended questions is that in the case of an open-ended question there can be 100 different answers of 100 students.

1.4 Limitations

- This model requires a lot of data for training to do well.
- Computationally expensive.
- Accuracy may vary according to data.

- It cannot handle open-ended questions as there can be many possible answers.
- Cannot reach human accuracy.

Chapter 2

Review of Literature

2.1 Techniques

Most important thing to choose was the right technique on which we would train our model. The technique should be semantic based so that meaning of sentence is compared not the actual words are compared. Analyzing many literature we came up with following techniques used in auto-evaluating subjective answers.

- BagOfWords
- Word2Vec
- Bert

2.1.1 BagOfWords

Bag-Of-Words model is one of the most widely used technique for sentiment analysis[4]. But it has some major weaknesses like analyzing sentiments ignoring semantics of words. This can be improve if we assign weight to word instead of TF(Term-Frequency)[5].

2.1.2 Word2Vec

Word2Vec is mostly used for text summarization[6] and in the age of big text data it is very popular[7]. This is used for dimension reduction and also for word embedding. As we all now that computer understand numbers,so it is very easy for machine to understand vector. This technique is very commonly used for text mining. This technique takes the word from the sentence and converts it into Vectors. The similar words have very close cosine distance. Cosine distance is the distance between similar meaning words. Word2Vec make a cluster of similar words into a vector. These vectors are used in pre-processing of ESA. But there is certain disadvantage or you can say defect of it that it cannot understand the context of the sentence.[8].

2.1.3 BERT(Bidirectional Encoder Representation of Transformers)

BERT[9] can give very high accuracy result but requires a lot of training time[10]. BERT has some advantages on other techniques because of its bidirectional approach, this model can understand the meaning of each word based on context both to the right and to the left of the word, this represents a clear advantage in the field of context learning. [11]. Also BERT has been known to be far more less complicated than most of the NLP techniques. BERT uses bidirectional approach that is why it is more likely to understand semantic of sentence[12].

2.2 Grading

2.2.1 Assign weight

Weight value can be assigned to each word according to their importance in sentence.

$$\sum_{k=1}^{k} P_k W_k \tag{2.1}$$

Where P_k is the k^{th} parameter and W_k is the weight value of k^{th} parameter. After assigning the weight value to each parameter, the weight value and the parameter value is multiplied. Then add all value of multiplication which can be the final marks of that answer script.

2.2.2 Accuracy based

Find the semantic accuracy of sentences and assign a threshold to the model[13]. Sentences that reaches specific threshold value will be graded accordingly. Assign label to each such accuracy and assign a grade on every label.

2.3 Similarity measure

2.3.1 Cosine similarity

Cosine measure how two vectors are projected[14]. It can be very useful to measure similarity between two non-zero vectors and how their angles are somehow pointing in same direction.

$$Cosine - similarity(A, B) = \frac{(A, B)}{||A||.||B||} \tag{2.2}$$

2.3.2 Bi-gram similarity

Bigram is simply a sequence of two adjacent vectors. One way is to compute number of bigram in list and then divide those common bigram by the average bigram length of assigned bigram list[15]. Division will produce the bigram similarity.

2.4 Conclusion

We studied several techniques and even implemented most of them. One good option to use these techniques was to first assign some weightage to words according to their importance in sentences but after reviewing more literature and implementing these techniques we end up comparing words and not semantics. To overcome this problem we studied more about Bert and Bi-LSTM. These techniques work bi-directionally, so one sentence is considered from both sides, and weights are assigned to it accordingly. So we considered using Bert and Bi-LSTM.

Chapter 3

Implementation

3.1 Analysis and Design

Use Cases:

1) Check Test

Name: Check Test

Goal: Check student's provided answer and teacher's answer similarity.

Actors: Teacher

Pre-conditions: Students have attempted test and answers are in database.

Basic Flow: Teacher will choose test number according to data and course, students' answers will be there available in database. System will check similarity between teachers answer and students answer and will display results.

Post conditions: Similarity results will be displayed and will be saved in database.

2) Add Answer

Name: Add answer

Goal: Add correct answers to the database

Actors: Teacher

Pre-conditions: Related question is already in database.

Basic Flow: Teacher will add 3 to 4 correct answers to by selecting related course and question so that student's answers can be compared to it and similarity is checked.

Post conditions: Teacher's answers will be saved with respect to its question.

3) Check Results:

Name: Check Results

Goal: Check similarity results

Actors: Teacher

Pre-conditions: Answers have been checked and saved in database.

Basic Flow: Teacher will choose option to check results by searching test result by id or by course or by date of test.

Post conditions: Result will be shown in table with accuracy and answers

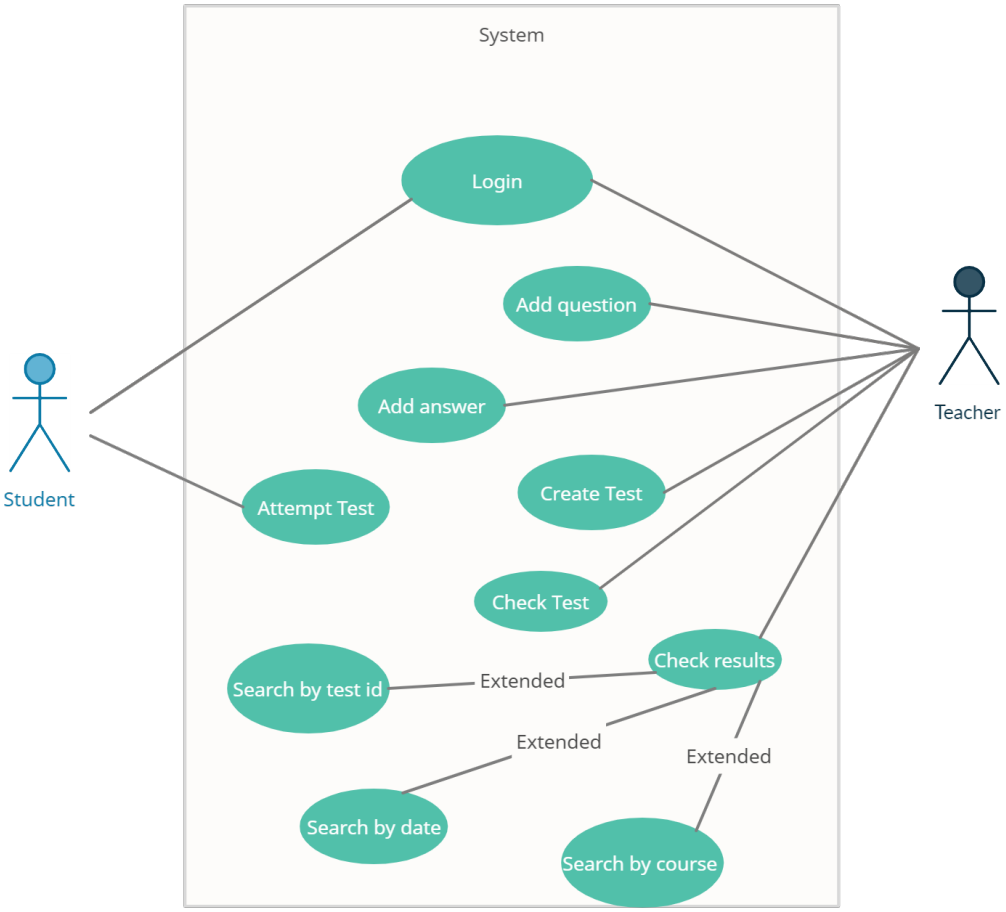

Figure 3.1: Use Case Diagram

- In student interface, once student is logged in, student can start its test which is assigned to him by teacher according to date and course.
- Student given test answer is stored in database and is checked by teacher
- Each answer is compared with all answers teacher has provided to related question and one answer having best similarity is considered.

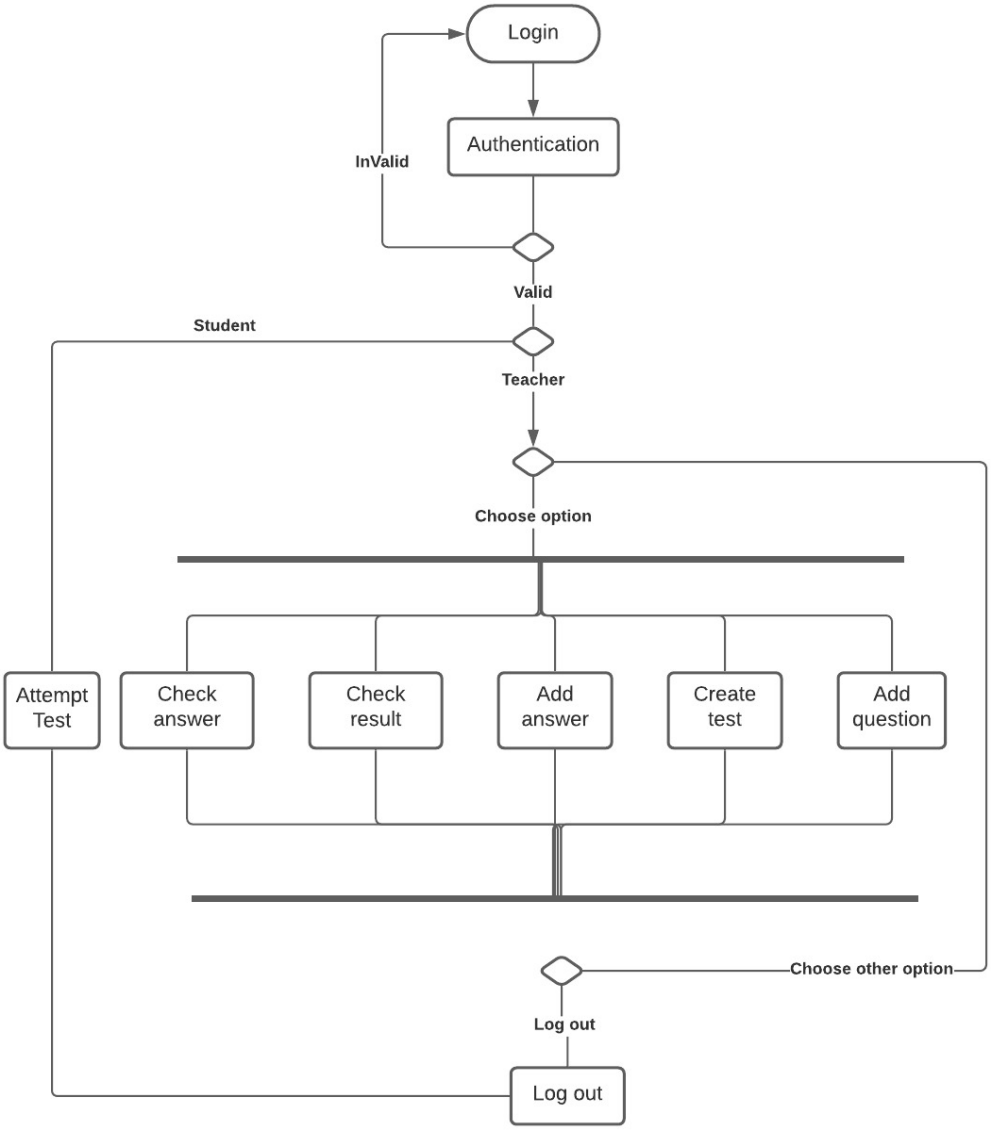

Figure 3.2: Flow Chart

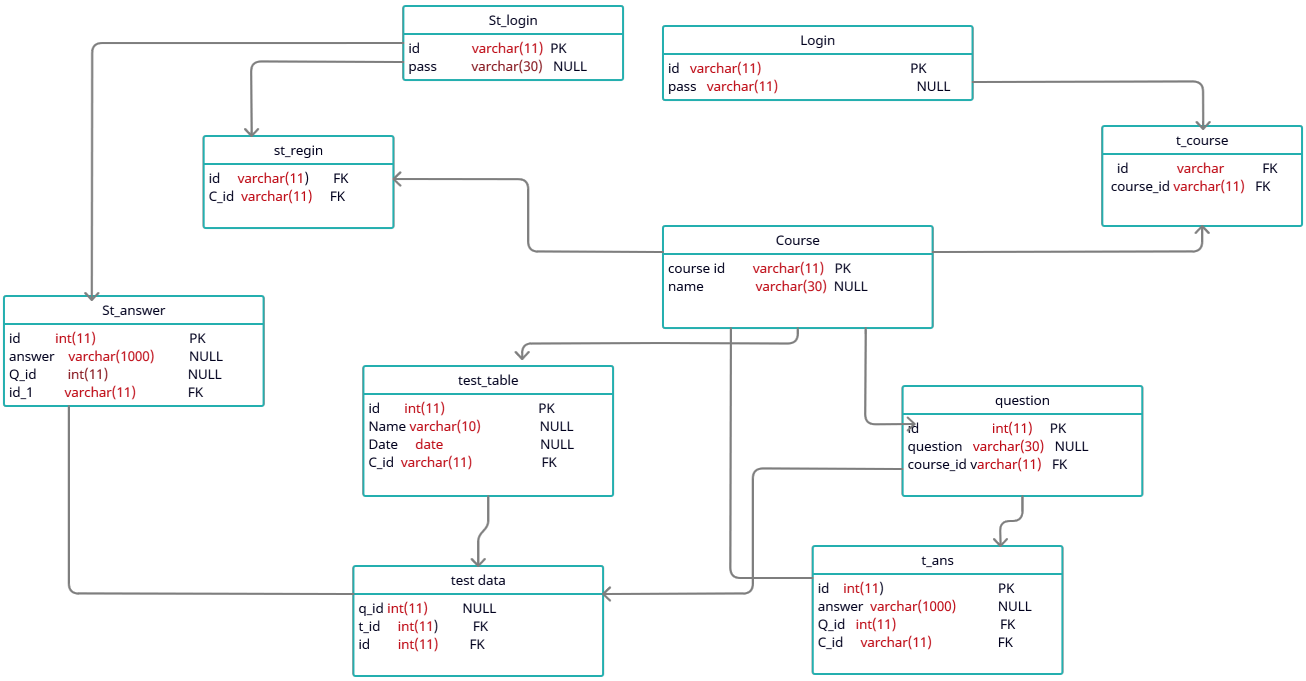

Figure 3.3: Class Diagram

3.2 Data

During the initial stage of the project, we generated artificial data which consisted of around 1000 data for training. This artificial data[16] consists of answers to the most common close-ended questions in software engineering, these answers have been written manually by all group members and also labeled accordingly. After getting the wanted results we gathered actual data of students' answers from FAST NUCES. Currently, the total dataset on which the model has trained consists of around 3700 data.

Our data consists of 3 labels that have been assigned manually according to marks assigned to them by the teacher. Each answer has its own label.

Labels are:

Entailment: Both sentences are strictly similar.
Neutral: Both sentences are fairly similar.
Contradiction: Both sentences are not similar at all.

Labels in our data is divided as:

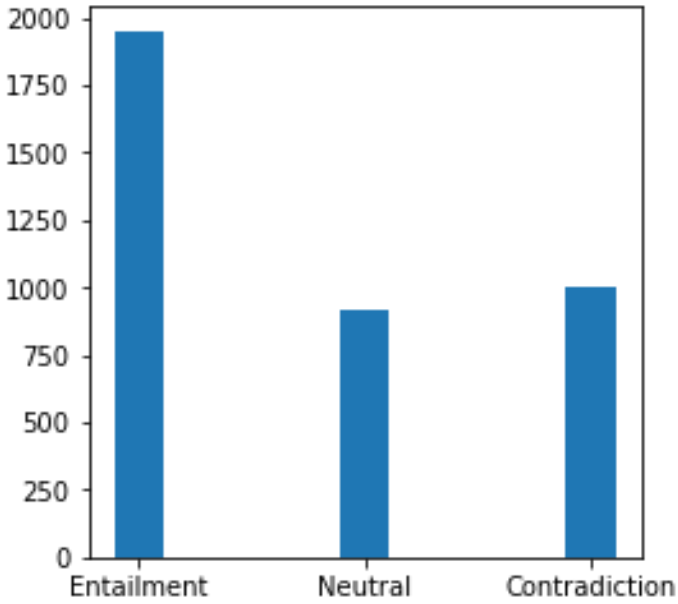

Figure 3.4: Division of labels

3.3 Model training

At first to find similarity between two sentences we implemented some techniques like Word2Vec [17] with skipgram[18] and bag of words [19] along with tf-id [20] but soon we realized that word-based similarity comparison between sentences is not more than just comparing words rather than semantic, if we have the same word in both sentences then it is possible to get higher accuracy. To overcome this we used 'BERT' [21] Bidirectional Encoder Representations Transformers and Bi-lstm [22].

That algorithm is a deep learning algorithm related to natural language processing. It helps a machine to understand, the words in a sentence what they have meant with all ability to express delicate shades of context.

We trained our model on Bert with students' real data and some artificial data.

3.3.1 Transformer

In Bert, we have used transformers which we have two separate mechanisms to convert words to a different number which is easy for the machine to have work done on it, if it has to work on word based strategy, maybe it's not possible for the machine to get anything, so for machine feasibility we have to convert it into numbers [23]. Where we used encoder, which reads from text and a special pattern converts it into numbers, and decoder is used to convert numbers to text, a decoder is also known as a prediction of output task.

3.3.2 Masked

Then we generate token. For this we take out some words from sentence and put tokens instead of those words for e.g. :

$$Refactoring \quad is \quad essential- > \qquad Refactoring \quad is \quad [MASK]$$

The model learns in the following way: Model predicts or tries to get the original word that was masked and that is done on the basis of the context provided by the rest of the sentence which in this case is: "Refactoring is". Which the following code represents.

```
# Encoded token ids from BERT tokenizer.
input_ids = tf.keras.layers.Input(
    shape=(max_length,), dtype=tf.int32, name="input_ids"
)

# Attention masks indicates to the model which tokens should be attended to.
attention_masks = tf.keras.layers.Input(
    shape=(max_length,), dtype=tf.int32, name="attention_masks"
)

# Token type ids are binary masks identifying different sequences in the model.
token_type_ids = tf.keras.layers.Input(
    shape=(max_length,), dtype=tf.int32, name="token_type_ids"
)

# Loading pretrained BERT model.
bert_model = transformers.TFBertModel.from_pretrained("bert-base-uncased")

sequence_output, pooled_output = bert_model(
    input_ids, attention_mask=attention_masks, token_type_ids=token_type_ids
)
```

Figure 3.5: Bert Hidden Layer

3.3.3 Semantic Data Generator

It is used to get 42 randomly examples or data from the dataset and shuffle it and each example's sentence is combined with the output separately and all of this is converted into an array where the argument is class taking an array of two sentences as combined, its labels, and batch size, shuffle data (True or False).

As Output of in Tuple's array with ids, attention mask, and type. Labels are required in the training model only.

3.3.4 Bi-LSTM:

Using bidirectional[24] that will run inputs in two ways, one from past to future and one from future to past, runs backward and forward, which preserve information from the future and backward and using two hidden states combined we are able in any point in time to preserve information from both past and future. This makes mask each word in a sentence forward and backward differently, the output of the last hidden layer is combined with the forward hidden layer and backward hidden

layer. And each layer contains 256 neurons backward and forward so the combined neurons are 512.

Artificial neuron's methodology makes the program think almost like a human brain. In this, the previous layer's single neuron is connected with the next layer's all neurons. This helps us understand the behavior of data.

Equations:

```
# Add trainable layers on top of frozen layers to adapt the pretrained features on the new data.
bi_lstm = tf.keras.layers.Bidirectional(
    tf.keras.layers.LSTM(256, return_sequences=True)
)(sequence_output)
```

Figure 3.6: Bi-LSTM

BiLSTM neural network[25] will produce a hidden vector h_t. Then h_t is used as input to multi-layer Perception to get a new hidden representation u_t which is represents in 3.1 equation where b_w tell the weight of the backward layer and tanh is used to scale the wait near around zero.

In equation 3.2 u_t is tell the importance of the word given by h_t and u_w tell the weight of the word in whole context of the sentence. In last equation 3.3 you compare the weight of previous hidden layer u_t weight with theu_w new layer weight which means similarity between both of them.

$$u_t = tanh(W_w h_t + b_w) \tag{3.1}$$

$$\partial_t = \frac{exp(u_t^T u_w)}{\sum_t exp(u_t^T u_w)} \tag{3.2}$$

$$s = \sum_t \partial_t h_t \tag{3.3}$$

3.3.5 Model training:

We use all layers in a sequence where the first layer gets a sequence of output from BERT pre-trained language model function that sequence of output directly input as Bi-LSTM neural network consists of 256 neurons and return sequence to average pooling and max pooling where get Average and max marks values from a whole pole or batch then both sequences concatenate and last use dense to providing one output to the next. For chosen optimal solution ADAM optimizers are used which always go for the best and optimal answers [26].

Training model while BERT Model freeze because of Feature Extraction on the top layer and it will allow the model to use the representations of the pre-trained model when trained then unfreeze BERT model trainable and trained on the previous model which will make change effectively and will increase accuracy.

Summary of First Trained model:

Model: "model"

Layer (type)	Output Shape	Param #	Connected to
input_ids (InputLayer)	[(None, 512)]	0	
attention_masks (InputLayer)	[(None, 512)]	0	
token_type_ids (InputLayer)	[(None, 512)]	0	
tf_bert_model (TFBertModel)	((None, 512, 768), (	109482240	input_ids[0][0] attention_masks[0][0] token_type_ids[0][0]
bidirectional (Bidirectional)	(None, 512, 512)	2099200	tf_bert_model[0][0]
global_average_pooling1d (Globa	(None, 512)	0	bidirectional[0][0]
global_max_pooling1d (GlobalMax	(None, 512)	0	bidirectional[0][0]
concatenate (Concatenate)	(None, 1024)	0	global_average_pooling1d[0][0] global_max_pooling1d[0][0]
dropout_37 (Dropout)	(None, 1024)	0	concatenate[0][0]
dense (Dense)	(None, 3)	3075	dropout_37[0][0]

Total params: 111,584,515
Trainable params: 2,102,275
Non-trainable params: 109,482,240

Figure 3.7: First Trained model

Summary of Second Trained model :

bidirectional (Bidirectional)	(None, 512, 512)	2099200	tf_bert_model[0][0]
global_average_pooling1d (Globa	(None, 512)	0	bidirectional[0][0]
global_max_pooling1d (GlobalMax	(None, 512)	0	bidirectional[0][0]
concatenate (Concatenate)	(None, 1024)	0	global_average_pooling1d[0][0] global_max_pooling1d[0][0]
dropout_37 (Dropout)	(None, 1024)	0	concatenate[0][0]
dense (Dense)	(None, 3)	3075	dropout_37[0][0]

Total params: 111,584,515
Trainable params: 111,584,515
Non-trainable params: 0

Figure 3.8: Second Trained model

* params = neurons

3.3.6 Evaluation:

After model training, we can evaluate the model by giving two sentences and it will return us a label with accuracy. This accuracy shows us how similar both sentences are. The more the accuracy, the more the model is certain about the label.

Entailment: Both sentences are strictly similar.

Neutral: Both sentences are fairly similar.

Contradiction: Both sentences are not similar at all.

3.3.7 Bert limitations:

- It cannot handle long sequences.
- It supports up to 512 token [27].
- It is Compute-Intensive.
- If we train model on more than 512 token, then it will come up with more complex or high computations and hard to improve accuracy.

Proposed solution: We increase the word limit of answer from 500 to 500+. Bert has a limitation of 512 tokens which affects our sentence length. In this case, we can't increase the length of the sentence so we have handled this limitation by breaking the long sentence into 512 lengths of words after 512 words they will store into a different sentence and will compare as well and get the similarity from the authentic teacher provided answer and those marks will add up and will be considered as one answer's marks. This task will be done parallelly with help of threads to lessen the time consumption

Consider the following scenario:

Teacher Sentence (TS): 400 words

Student Sentence (SS) at 800 words

SS: ss1= 500 and ss2= 300

Check Similarity (TS, ss1) = (For e.g. if it returns entailment with 80%)

Check Similarity (TS, ss2) = (For e.g. if neutral with 20%)

.

entailment	entailment	%+%=%
entailment	neutral	%+%=%
entailment	contradiction	%+%=%
neutral	neutral	%+%=%
neutral	contradiction	%+%=%
contradiction	contradiction	%+%=%

Table 3.1: Grading

3.4 GUI

For GUI we have used Tkinter which is a python GUI library.

- Select teacher or student

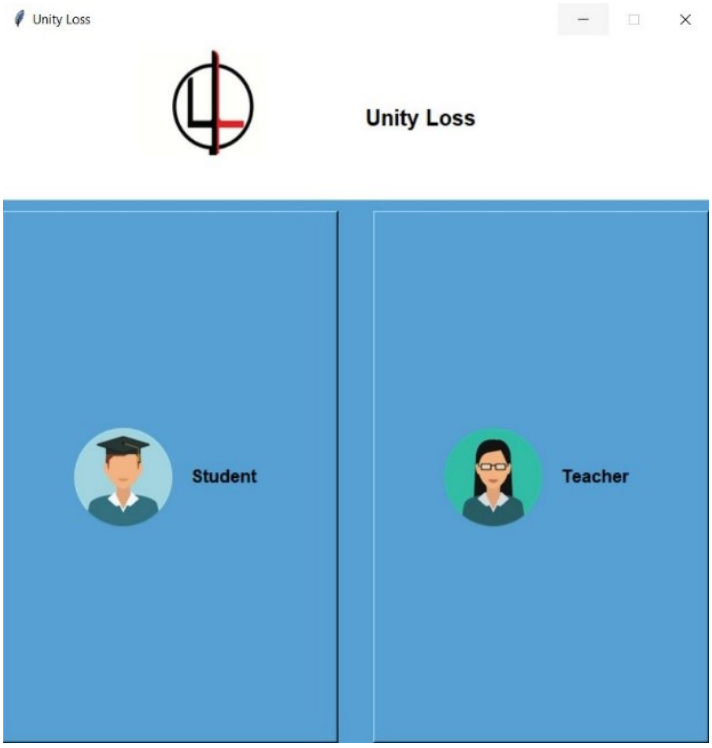

Figure 3.9: Interface index page

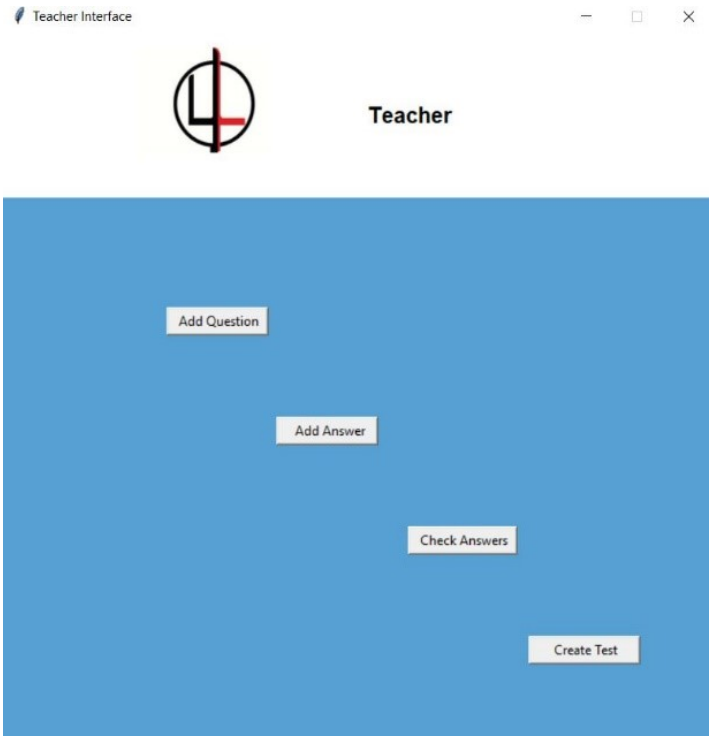

Figure 3.10: Teacher Interface

- Teacher has Following Options:
  - Teacher can add question
  - Teacher can add answers to those questions
  - Teacher can create test with those questions which have answers
  - Teacher can check test which have been attempted by students

- Student after logging can attempt test assigned to him by teacher.

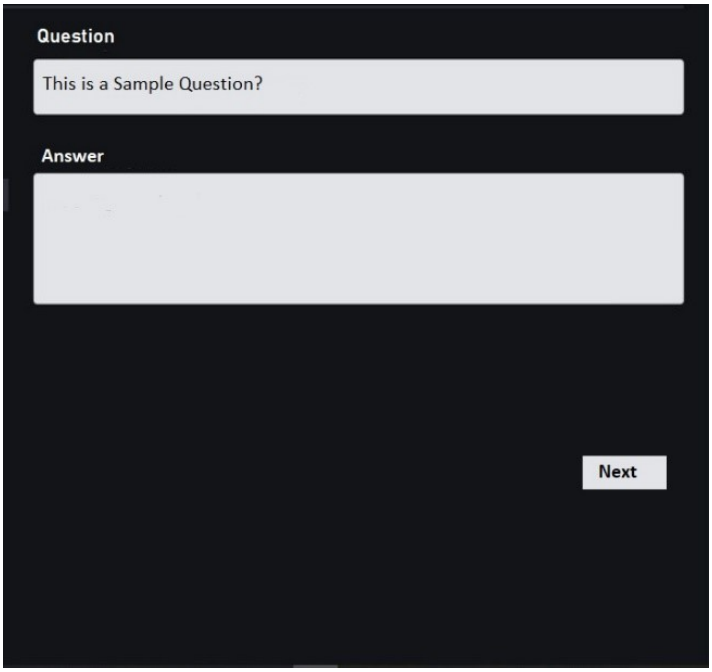

Figure 3.11: Student Test Interface

3.5 Grading

For grading, we have assigned 80-100 marks to entailment label, 60-80 marks to neutral, and 60-40 marks to contradiction depending upon how much contradiction is there in the sentence. More the contradiction less the marks are given. If the similarity of any label is below 60 code, the system will auto-grade at 0.0 and assign it to the exceptional cases.

Code for Grading:

```
if(label=="entailment"):
  if(Similarity>=60):
    ans=Similarity-60
    ans=(ans/2)+80

if(label=="neutral"):
  if(Similarity>=60):
    ans=Similarity-60
    ans=(ans/2)+60

if(label=="contradiction"):
  if(Similarity>=60):
    ans=Similarity-40
    ans=60-ans
```

Chapter 4

Results

4.1 Results on Artificial Data

4.1.1 Sample 1:

Question 1:
What is Grey box –testing?

Teacher's Answer:
It is only a strategy for software debugging in which the tester has only limited or some knowledge of the internal details of the program or software.

#	**Student Answers**	**Label**	**Accuracy**
1	A technique where tester software has a certain limit to test a program which is based on knowledge of internal details.	Entailment	0.76
2	It is a way for software to debugging the program where tester has boundaries of internal details of the program is known as Gray Box	Entailment	0.95
3	Testing in which there is limited knowledge of program.	Entailment	0.96
4	Process which includes only some information of internal structure	Contradiction	0.55
5	In which there is limited knowledge of software	Entailment	0.97
6	It can be said as type of testing which has restricted information of software	Entailment	0.58
7	SDebugging with limited work	Contradiction	0.90

Table 4.1: First Model Result Trained on Artificial Data

.

4.1.2 Sample 2:

Question 2:
What is a stake holder in software?

Teacher's Answer:
Anyone who has some interest or has invested in working of team or having interest in released products.

Student Answers	Label	Accuracy
A stakeholder is anyone outside of the team with vested interests in the working of the team and the products they release	Entailment	0.83
stakeholders are the people or groups that are or can be affected by a software development project	Entailment	0.91
A stakeholder outside of office with some interest in the working of the team and the products they release.	Entailment	0.76
A stakeholder not belong from the team but and they are not interested in work of the team and releasing the products	Contradiction	0.53
Stakeholder is a person, group or company that is directly or indirectly involved in the project or software and gets effected by the outcome of project	Entailment	0.89
Any person who is directly or indirectly affected by the project	Entailment	0.95
Any person with no interest in project	Contradiction	0.68
Any individual with no direct or indirect interest	Entailment	0.63
A secondary user of project is also known as stake holder	Neutral	0.58

Table 4.2: Sample 2 on Artifical Data

4.2 Results on student's actual answers

4.2.1 Sample 3:

Question 3:
What will happen to the speed of development process if we add new members to the team during project?

NO.	Teacher's Answer
1	At first it will decrease the speed because of new member but for long term it will increase the speed of development
2	If the staff is experienced it will increase the speed else it will only decrease the speed of development.
3	Considering that this is a large-sized software system, increasing the development team size might not speed up the process as the new people, who are unaware of the whole operation
4	Increasing speed will not be good idea, there will be mismanagement and poor team work

NO.	Student's Answer
A	If member is qualified well then it can develop it quickly
B	Yes, we can divide task into many modules and give group to each module
C	It can or cannot increase the speed of development depends on the developer
D	If member is bad in development in software, then it will not increase
E	If you have specialized staff, it may increase speed else not.
F	Adding new member is very efficient project will be done in no time
G	Process become very fast if new member is added to the team

#	**1**	**2**	**3**	**4**
A	Neutral 0.99%	contradiction 0.75%	contradiction 0.71%	contradiction 0.67%
B	contradiction 0.79%	contradiction 0.92%	contradiction 0.51%	contradiction 0.94%
C	Neutral 0.98%	Neutral 0.99%	Neutral 1.00%	Neutral 0.99%
D	contradiction 0.92%	contradiction 1.00%	contradiction 1.00%	contradiction 1.00%
E	contradiction 0.83%	Neutral 0.69%	Neutral 0.70%	contradiction 0.78%
F	contradiction 1.00%	contradiction 1.00%	contradiction 1.00%	contradiction 1.00%
G	contradiction 1.00%	contradiction 1.00%	contradiction 1.00%	contradiction 1.00%

4.2.2 Sample 4:

Question 4:
Old System having high business value and quality having maintenance issue. What should we do?

#	**Teacher's Answer**
1	High business value and quality, it should be maintained using normal system maintenance.
2	As system has High business value system should be re-engineered it is expensive to maintain the system.
3	It should be re-engineered or replaced if suitable system is available, this is so because of its high business value it contributes a lot to the business.

#	**Student's Answer**
A	As the system is of a lot work to business, and running for a long time gone through many changes and may go to even more changes, option for this system should be the managed normally and constantly.
B	COTs can be used, we will do it by giving interface for access. System can update using legacy wrapping.
C	Only option is the replacement of the system.
D	7 We can solve it by simple providing interface as source of access.

Chapter 5

Discussion

From the beginning our main goal was to achieve enough accuracy that model can find accurate similarity between both of the answers (teacher's and students') according to data. The results shown in 4th chapter is fairly according to our goal. In the results entailment with higher accuracy or accuracy close to 1.0 shows both answers are very similar. In case of neutral, both sentences are sufficiently similar and if it neutral label has higher accuracy or accuracy closer to 1.0 it means it means sentences are more similar to each other. But in case of higher accuracy in contradiction, higher accuracy means both sentences are more contradictory in semantics.

Chapter 6

Conclusions and Future Work

In future, we can see a lot of technology and a lot less papers. Today or tomorrow, we have to move from paper exams to computer exams. Starting from this project we can achieve this .Only problem in our project was lack of data but we somehow managed this problem by generating data artificially and by inserting some past papers data. Some future goals and modification are described below:

1. If for some year any university or any educational institute take exam on computer and teacher check those answer on computer and assign them some number, we can retrieve those answers and numbers and assign them label according to those numbers for e.g. we assign entailment label to those answer which have scored between 80-100 marks and assign neutral to one who score between 60-80 and contradiction to those who have less than 60.

2. This data can help us achieve a lot of accuracy when data variation and data amount is large enough. We will train our machine on this data and then officially introduce our model for professional use.

3. In future if pandemic happen again and universities have to move to online examination this project will be helpful if we create web interface for the project.

Bibliography

[1] Piyush Patil, Sachin Patil, Vaibhav Miniyar, and Amol Bandal. Subjective answer evaluation using machine learning. *International Journal of Pure and Applied Mathematics*, 118(24):1–13, 2018.

[2] M Syamala Devi and Himani Mittal. Machine learning techniques with ontology for subjective answer evaluation. *arXiv preprint arXiv:1605.02442*, 2016.

[3] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *arXiv preprint arXiv:1310.4546*, 2013.

[4] Doaa Mohey El-Din. Enhancement bag-of-words model for solving the challenges of sentiment analysis. *International Journal of Advanced Computer Science and Applications*, 7(1), 2016.

[5] Juan Peng. Research on the text similarity algorithms in automatic scoring of subjective questions. In *Journal of Physics: Conference Series*, volume 1952, page 042039. IOP Publishing, 2021.

[6] Nabil Alami, Mohammed Meknassi, and Noureddine En-nahnahi. Enhancing unsupervised neural networks based text summarization with word embedding and ensemble learning. *Expert systems with applications*, 123:195–211, 2019.

[7] Boris Galitsky. Distributional semantics for crm: Making word2vec models robust by structurizing them. In *Artificial Intelligence for Customer Relationship Management*, pages 25–56. Springer, 2020.

[8] Joseph Lilleberg, Yun Zhu, and Yanqing Zhang. Support vector machines and word2vec for text classification with semantic features. In *2015 IEEE 14th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, pages 136–140. IEEE, 2015.

[9] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer, 2019.

[10] Cristóbal Colón-Ruiz and Isabel Segura-Bedmar. Comparing deep learning architectures for sentiment analysis on drug reviews. *Journal of Biomedical Informatics*, 110:103539, 2020.

[11] Santiago González-Carvajal and Eduardo C Garrido-Merchán. Comparing bert against traditional machine learning text classification. *arXiv preprint arXiv:2005.13012*, 2020.

[12] MV Koroteev. Bert: A review of applications in natural language processing and understanding. *arXiv preprint arXiv:2103.11943*, 2021.

[13] Himani Mittal. Review based on techniques for subjective answer evaluation using natural language processing techniques, 08 2016.

[14] Ming Tan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*, 2015.

[15] M Rahman and Fazlul Hasan Siddiqui. Nlp-based automatic answer script evaluation. *vol*, 4:35–42, 2018.

[16] Johannes Bjerva, Nikita Bhutani, Behzad Golshan, Wang-Chiew Tan, and Isabelle Augenstein. Subjqa: A dataset for subjectivity and review comprehension. *arXiv preprint arXiv:2004.14283*, 2020.

[17] Haixia Liu. Sentiment analysis of citations using word2vec. *arXiv preprint arXiv:1704.00177*, 2017.

[18] Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*, 2014.

[19] Suzan Verberne, Lou Boves, Nelleke Oostdijk, and Peter-Arno Coppen. What is not in the bag of words for why-qa? *Computational Linguistics*, 36(2):229–245, 2010.

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

[21] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*, 2019.

[22] Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 221–231, 2017.

[23] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. Pretrained transformers for text ranking: Bert and beyond. *arXiv preprint arXiv:2010.06467*, 2020.

[24] Yonghua Zhu, Xun Gao, Weilin Zhang, Shenkai Liu, and Yuanyuan Zhang. A bi-directional lstm-cnn model with attention for aspect-level text classification. *Future Internet*, 10(12):116, 2018.

[25] Gang Liu and Jiabao Guo. Bidirectional lstm with attention mechanism and convolutional layer for text classification. *Neurocomputing*, 337:325–338, 2019.

[26] Imran Khan Mohd Jais, Amelia Ritahani Ismail, and Syed Qamrun Nisa. Adam optimization algorithm for wide and deep neural network. *Knowl. Eng. Data Sci*, 2(1):41–46, 2019.

[27] Chris Alberti, Kenton Lee, and Michael Collins. A bert baseline for the natural questions. *arXiv preprint arXiv:1901.08634*, 2019.