**PREDICTING EFFECTIVE**

**ARGUMENTS THROUGH STUDENT RESPONSES**



A Capstone project report

in partial fulfillment of the requirement for the award of the degree

### BACHELOR OF TECHNOLOGY

**in**

### COMPUTER SCIENCE & ENGINEERING

**and**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**By**

19K41A0599 IRUKULLA APOORVA

19K41A05B6 T.PHANI PRIYA

19K41A04G6 M .SNEHA

Under the guidance of

# D. Ramesh Kumar

Assistant Professor, Department of CSE.



# SR ENGINEERING COLLEGE

Ananthasagar, Warangal.



**CERTIFICATE**

This is to certify that this project entitled **“PREDICTING EFFECTIVE ARGUMENTS THROUGH STUDENT RESPONSES”** is the bonafide work carried out by **IRUKULLA APOORVA, T.PHANIPRYA, M.SNEHA,** bearing Roll No(s) 19K41A0599,19K41A05B6,19K41A04G6as a Capstone project for the partial fulfillment to award the degree **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE & ENGINEERING /ELECTRICAL & ELECTRONICS ENGINEERING** during the academic year 2022-2023 under our guidance and Supervision.

### Mr. D. RAMESH Dr.M. Sheshikala

Asst. Professor, Assoc. Prof. & HOD (CSE),

S R Engineering College, S R Engineering College,

Ananthasagar, Warangal Ananthasagar, Warangal

**External Examiner**

# ABSTRACT

The current education system did not put much emphasis in persuasive writing, which may hinder critical thinking development of the students. The task is to build an argument grading system. The purpose of the study is to develop a grading system by predicting effective arguments through student responses which can grade students’ response based on three factors in user writing as effective, adequate, or ineffective. The proposed system is evaluated using datasets from Kaggle. The accuracy of model and obtained results show an agreement with user’ grading. This gives us an indication that the model can be deployed for response of students’ writing, thereby leading to reduction in time, efforts and cost for evaluating an essay.

**Table of contents**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Content** | **Page No** |
| 1 | Introduction | 5 |
| 2 | Literature Review | 8 |
| 3 | Design | 9 |
| 4 | Data set | 10 |
| 5 | Pre-processing | 11 |
| 6 | Methodology | 12 |
| 7 | Results | 19 |
| 8 | Conclusion | 21 |
| 9 | References | 22 |

# INTRODUCTION

* 1. **OVERVIEW**

The assessment plays a significant role in measuring the learning ability of the student. The education system is changing its shift to online-mode, like conducting computer-based exams and automatic evaluation. It is a crucial application related to the education domain, which uses natural language processing (NLP) and Machine Learning techniques. The evaluation of responses is impossible with simple programming languages and simple techniques like pattern matching and language processing. Here the problem is for a single question, we will get more responses from students with a different explanation. So, we need to evaluate all the answers concerning the question. Response scoring is a computer-based assessment system that automatically scores or grades the student responses by considering appropriate features. These systems use natural language processing (NLP) techniques that focus on style and content to obtain the score of an essay. The vast majority of the essay scoring systems in the 1990s followed traditional approaches like pattern matching and a statistical-based approach. Since the last decade, there response grading systems started using regression-based and natural language processing techniques.

# RELATED WORK:

Predicting effective arguments through student responses,

[1] Automated essay scoring is the task of automatically assigning scores to essays as an alternative to human grading. DNN-AES models used for taraing on a large dataset of grading essay, this achieved state-of-the-art accuary.The Dnn-AES framework that integrates IRT models to deal within traing data. [2] Automated essay scoring is one of the most important problem in Natural Language Processing. It has been explored for a number of years, and it remains partially solved. Many works in the past have attempted to solve this problem by using RNNs, LSTMs, etc. This work examines the transformer models like BERT, RoBERTa. [3] Reviewed AES systems on six dimensions like dataset, NLP techniques, model building, grading models, evaluation, and effectiveness of the model. Feature extraction is with NLTK, WordVec, and GloVec NLP libraries; these libraries have many limitations while converting a sentence into vector form. [4] compare two powerful language models, BERT and XLNet, and describe all the layers and network architectures in these models. System lucidate the network architectures of BERT and XLNet using clear notation and compare the

results with more traditional methods, such as bag of words (BOW) and long short term memory (LSTM) networks. [5] in the area of Automated Essay Scoring (AES), pre-trained models such as BERT have not been properly used to outperform other deep learning models such as LSTM. In this paper, we introduce a novel multi-scale essay representation for BERT that can be jointly learned and may be a new and effective choice for long-text tasks. [6] It helps reduce manual workload and speed up learning feedback. The model termed Siamese Bidirectional Long Short-Term Memory Architecture (SBLSTMA) can capture not only the semantic features in the essay but also the rating criteria information behind the essays. Here it use the SBLSTMA model for the task of AES and take the Automated Student Assessment Prize (ASAP) dataset as evaluation. [7] Deep learning algorithms such as Multilayer Perceptron (MLP), Long Short-Term Memory (LSTM), and Gated Recurrent Unit (GRU) were used to learn the model with performance evaluation on metrics such as validation accuracy, training time, loss function, and Quadratic Weighted Kappa. MLP, LSTM, and GRU had average validation accuracy of 0.48, 0.537, and 0.511 respectively. GRU was shown to be the optimal classifier and was used in the development of the essay scoring model. [8] model is a long short-term memory neural network and is trained as a regression method. long short-term memory networks have been used to obtain parse trees by using a sequence-to-sequence model. [9] This paper presents a transformer-based neural network model for improved AES performance using Bi-LSTM and RoBERTa language model based on Kaggle’s ASAP dataset.[10] a RNNs, particularly LSTMs, are good at representing text sequences, essays are longer structured documents and less well suited to an RNN representation. compared performance on three essay scoring tasks with different characteristics, contrasting results with a strong feature-based system.

**1.3PRESENT WORK**

Inthe proposed system we are using LSTM and Bert

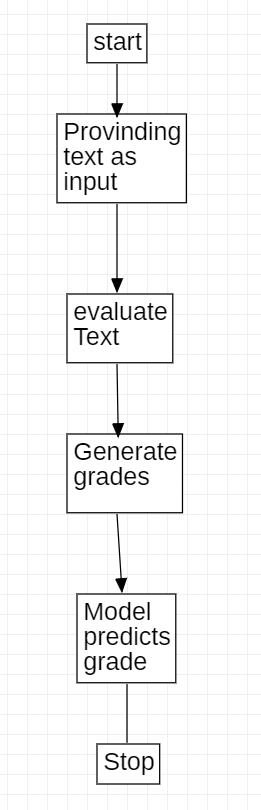


Fig: 1.1 *Overview of proposed system*

**2. LITERATURE SURVEY**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| S.No | Published date | Author | Title | Methodology | Article ID | accuracy | Link |
| 1 | 30-Jun-2020 | Masaki Uto, Masaki Okano | Automated essay scoring using response theory | CNN-LSTM, BERT | 9783030 | 0.88 | <https://link.springer.com/chapter/10.1007/978-3-030-52237-7_44> |
| 2 | Nov-2022 | Kshitij Gupta | Data Augmentation for automated Essay scoring | RNN,LSTM,BERT,RoBERTa | 364689774 |  | <https://www.researchgate.net/publication/364689774_Data_Augmentation_for_Automated_Essay_Scoring_using_Transformer_Models> |
| 3 | 23-Sep-2021 | Suresh kumar Sanampudi | An automated essay scoring system | CNN-LSTM | 10462021 |  | <https://link.springer.com/article/10.1007/s10462-021-10068-2> |
| 4 | 18-Sep-2019 | Christopher Ormerod | Language models and Automated Essay Scoring | BOW&LSTM,  BERT&XLNet | 190909482 |  | <https://arxiv.org/abs/1909.09482> |
| 5 | 8-May-2022 | Yongjie Wang, Chuan Wang | Use of bert for automated essay scoring | BERT | 220503835 |  | <https://arxiv.org/abs/2205.03835> |
| 6 | 10-Dec-2018 | Guoxi Liang, Dongwon Jeong | Essay Scoring using Nural networks | CNN and RNN(LSTM) | 329378585 |  | <https://www.researchgate.net/publication/329378585_Automated_Essay_Scoring_A_Siamese_Bidirectional_LSTM_Neural_Network_Architecture> |
| 7 | 11-April-2022 | Jumoke Eluwa, Shade O Kuyore | Essay scoring Model Based on Gated Recurrent unit Technique | TF(Term frequency),LSTM,GRU(gated recurrent unit) | 360440446 | 0.53 | <https://www.researchgate.net/publication/360440446_Essay_Scoring_Model_Based_on_Gated_Recurrent_Unit_Technique> |
| 8 | 5-May-2016 | Kaveh Taghipour, Hwee Tou Ng | A Neural Approach to Automated Essay Scoring | GRU,LSTM | 305748202 |  | <https://www.researchgate.net/publication/305748202_A_Neural_Approach_to_Automated_Essay_Scoring> |
| 9 | 1-Jun-2021 | Majidi Beseiso, Omar A.Alzubi | A Novel automated essay scoring approach | Bi-LSTM, RoBert | 101007 | 0.87 | <https://link.springer.com/article/10.1007/s12528-021-09283-1> |
| 10 | 2019 | Farah Nadeem, Huy Nguyen | Automated essay scoring using Discourse-Aware Neural Models | RNN,LSTM | W19-4450 | 0.86 | <https://aclanthology.org/W19-4450.pdf> |

**3.Design**

**3.1 REQUIREMENT SPECIFICATION(S/W & H/W)**

**Hardware Requirements**

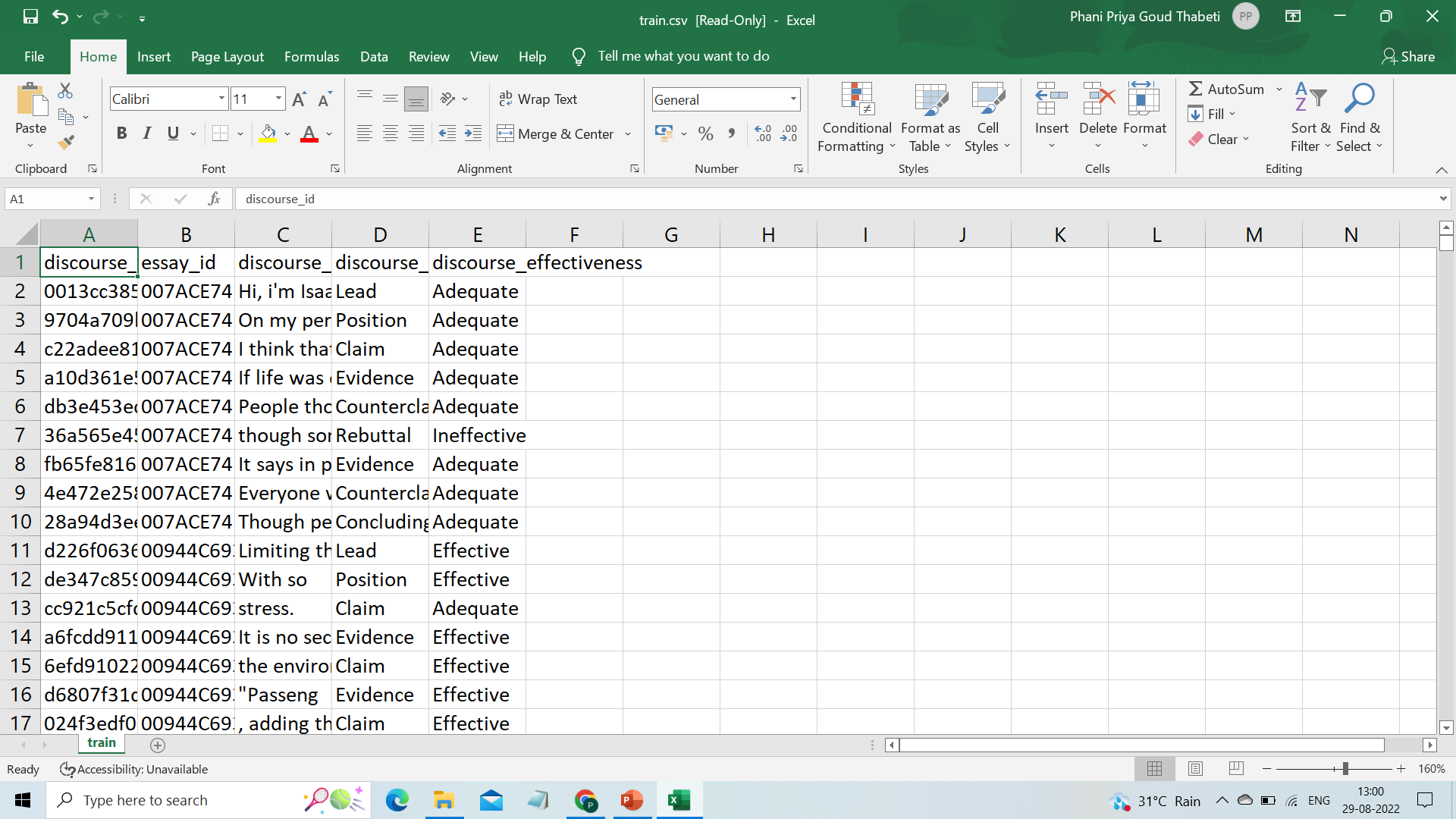
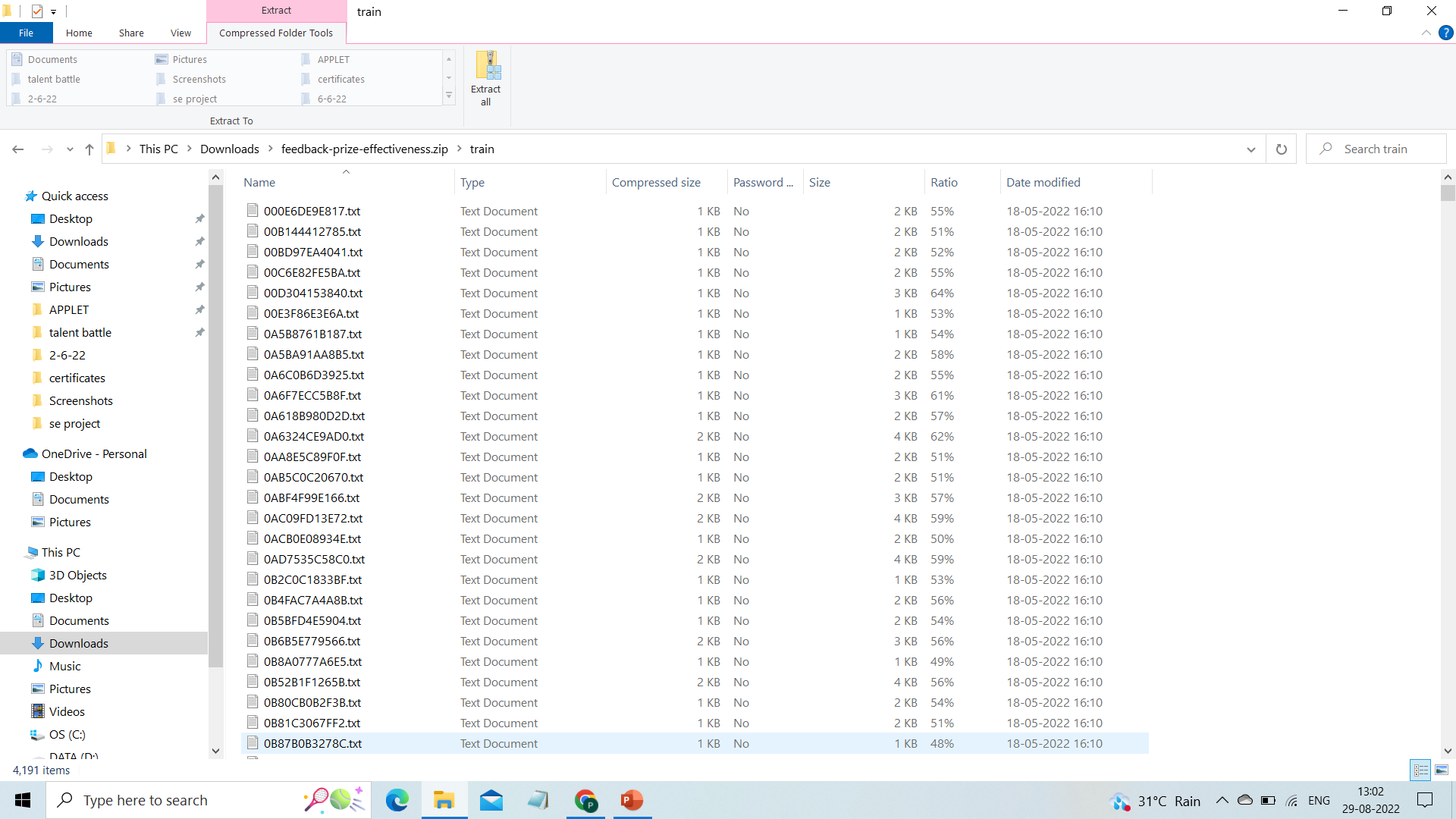
* **System** : Pentium 4, Intel Core i3, i5, i7 and 2GHz Minimum
* **RAM** : 4GB or above
* **Hard Disk** : 10GB or above
* **Input**  : Keyboard and Mouse
* **Output** : Monitor or PC

**Software Requirements**

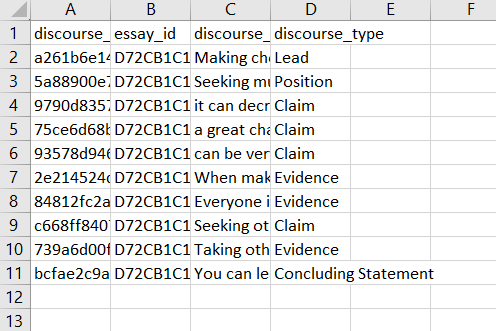
* **OS** : Windows 8 or Higher Versions
* **Platform** : Jupiter Notebook
* **Program Language** : Python

1. **DATA-SET**

Training.csv file Train folder

**** 

Testing.csv



1. **Pre-processing**

**Removing stop words:**

The words which are generally filtered out before processing a natural language are called stop words. These are actually the most common words in any language (like articles, prepositions, pronouns, conjunctions, etc) and does not add much information to the text. Examples of a few stop words in English are “the”, “a”, “an”, “so”, “what”. Stop words are available in abundance in any human language. By removing these words, we remove the low-level information from our text in order to give more focus to the important information. In order words, we can say that the removal of such words does not show any negative consequences on the model we train for our task. Removal of stop words definitely reduces the dataset size and thus reduces the training time due to the fewer number of tokens involved in the training. NLP is one of the most researched areas today and there have been many revolutionary developments in this field. NLP relies on advanced computational skills and developers across the world have created many different tools to handle human language. Out of so many libraries out there, a few are quite popular and help a lot in performing many different NLP tasks.

**Tokenization:**

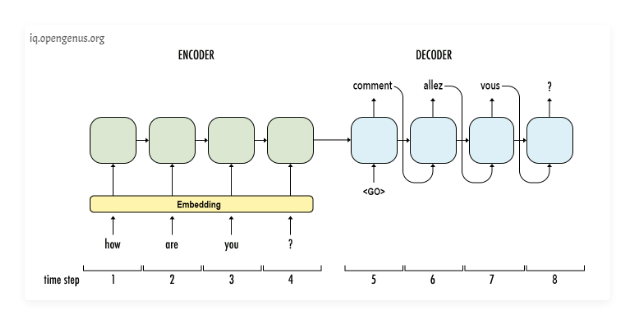
Tokenization is the first step in any NLP pipeline. It has an important effect on the rest of your pipeline. A tokenizer breaks unstructured data and natural language text into chunks of information that can be considered as discrete elements. The token occurrences in a document can be used directly as a vector representing that document. This immediately turns an unstructured string (text document) into a numerical data structure suitable for machine learning. They can also be used directly by a computer to trigger useful actions and responses. Or they might be used in a machine learning pipeline as features that trigger more complex decisions or behavior.

**punctuation removal:**

The punctuation removal process will help to treat each text equally. For example, the word data and data! are treated equally after the process of removal of punctuations. We need to take care of the text while removing the punctuation because the contraction words will not have any meaning after the punctuation removal process. Such as ‘don’t’ will convert to ‘dont’ or ‘don t’ depending upon what you set in the parameter.We also need to be extra careful while choosing the list of punctuations that we want to exclude from the data depending upon the use cases. As string.punctuation in python contains these symbols !"#$%&\'()\*+,-./:;?@[\\]^\_{|}~`

1. **Methodology**

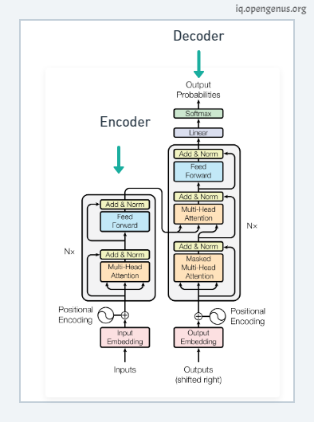
# LSTM Model

  
Initially LSTM networks had been used to solve the Natural Language Translation problem but they had a few problems. LSTM networks are-

* **Slow to train**. Words are passed in sequentially and are generated sequentially it can take a significant number of timesteps for the neural net to learn.
* **It's not really the best of capturing the true meaning of words**, even bi-directional LSTMS are not. Because even here they are technically learning left to right and right to left context separately and then concatenating them so the true context is lost.

# Transformer Architecture

This is the transformer neural network architecture that was initially created to solve the problem of language translation.

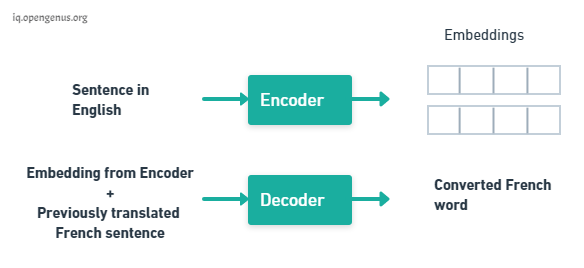


Transformer architecture addresses some of the drawback of LSTM:-

* They are **faster** as words can be processed simultaneously.
* The **context of words is better learned** as they can learn context from both directions simultaneously.

# Transformer Flow

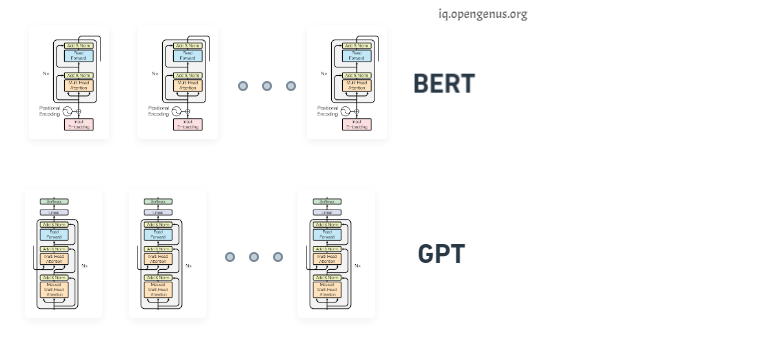
Now let's see the transformer in action. Say we want to train this architecture to convert English to French.



The transformer consists of two key components an **Encoder** and a **Decoder**.  
The Encoder takes the English words simultaneously and it generates embeddings for every word simultaneously these embeddings are vectors that encapsulate the meaning of the word, similar words have closer numbers in their vectors. The Decoder takes these embeddings from the Encoder and the previously

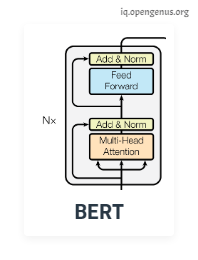
generated words of the translated French sentence and then it uses them to generate the next French word and we keep generating the French translation one word at a time until the end of sentence is reached.

What makes Transformer conceptually stronger than LSTM cell is that we can physically see a separation in tasks. The Encoder learns What is English and its grammar? and What is context? The Decoder learns how do English words relate to French words. Separately they both have some underlying understanding of language and it's because of this understanding that we can pick apart this architecture and build systems that understand language.



We stack the decoders and we get the **GPT** (Generative Pre-training) transformer architecture, conversely if we stack just the encoders we get **BERT** a **bi-directional encoder representation from transformer**.

# 2)BERT



We can use BERT for problems which needs Language understanding:

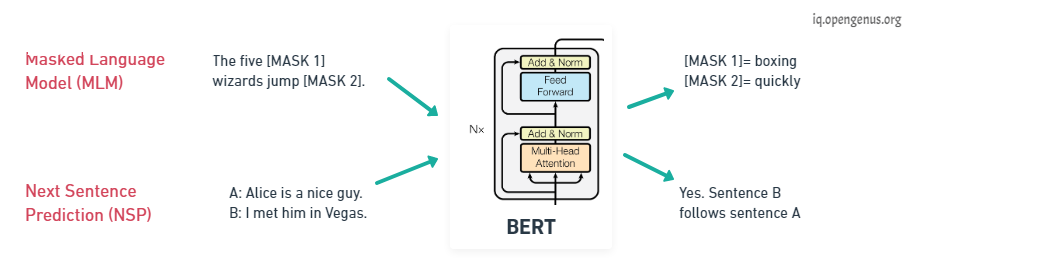
* Neural Machine Translation
* Sentiment Analysis
* Question Answering
* Text summarization

These problems can be solved by **BERT Training phases** which are:

1. **Pretain** BERT to understand language and context.
2. **Fine tune** BERT to learn how to solve a specific task.

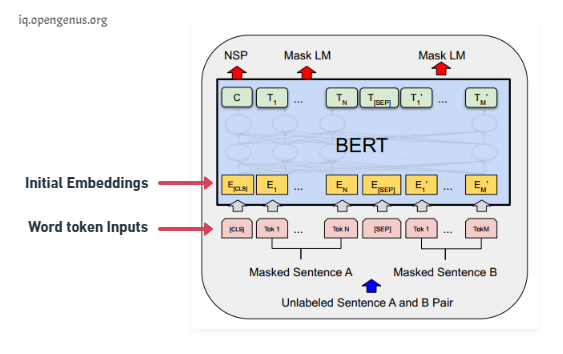
# 1. Pre-training

The goal of pre training is to make BERT learn what is language and what is context? BERT learns language by training on two Unsupervised tasks simultaneously, they are **Mass Language Modeling (MLM)** and **Next Sentence Prediction (NSP)**.



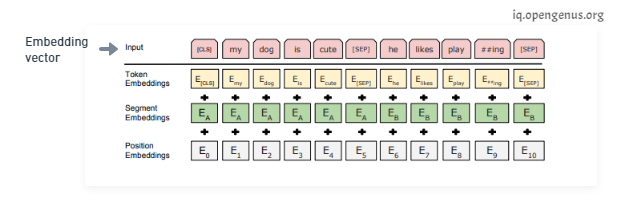
For **Mass Language Modeling**, BERT takes in a sentence with random words filled with masks. The goal is to output these masked tokens and this is kind of like fill in the blanks it helps BERT understand a bi-directional context within a sentence.

In the case of **Next Sentence Prediction**, BERT takes in two sentences and it determines if the second sentence actually follows the first, in kind of like a binary classification problem. This helps BERT understand context across different sentences themselves and using both of these together BERT gets a good understanding of language.

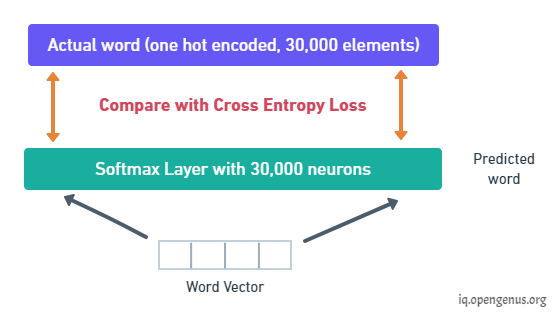


During BERT pre-training the training is done on Mass Language Modeling and Next Sentence Prediction. In practice both of these problems are trained simultaneously, the input is a set of two sentences with some of the words being **masked** (each token is a word) and convert each of these words into **embeddings** using **pre-trained embeddings**. On the output side **C is the binary output for the next sentence prediction** so it would output 1 if sentence B follows sentence A in context and 0 if sentence B doesn't follow sentence A. Each of the **T's here are word vectors that correspond to the outputs for the mass language model problem**, so the number of word vectors that is input is the same as the number of word vectors that we got as output.

On the input side, how are we going to generate embeddings from the word token inputs?



The initial embedding is constructed from three vectors, the **token embeddings** are the pre-trained embeddings; the main paper uses **word-pieces embeddings** that have a vocabulary of 30,000 tokens. The **segment embeddings** is basically the sentence number that is encoded into a vector and the **position embeddings** is the position of a word within that sentence that is encoded into a vector. Adding these three vectors together we get an embedding vector that we use as input to BERT. The **segment and position embeddings are required for temporal ordering** since all these vectors are fed in simultaneously into BERT and language models need this ordering preserved.



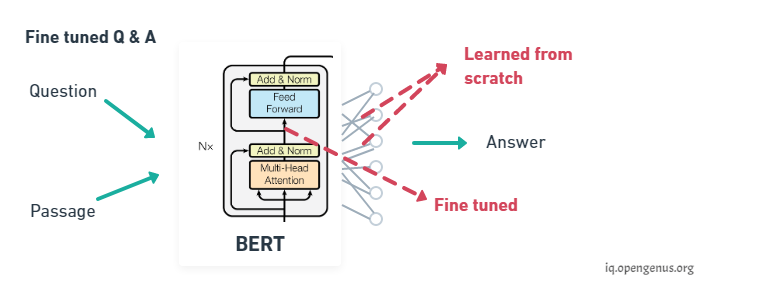
The output is a binary value **C** and a bunch of word vectors but with training we need to minimize a loss. So two key details to note here all of these word vectors have the same size and all of these word vectors are generated simultaneously, we need to take each word vector pass it into a fully connected layered output with the same number of neurons equal to the number of tokens in the vocabulary so that would be an **output layer corresponding to 30,000 neurons in this case and we would apply a softmax activation**. This way we would convert a word vector to a distribution and the **actual label of this distribution would be a one hot encoded vector** for the actual word and so we **compare these two distributions and then train the network using the cross entropy loss**.

But note that the output has all the words even though those inputs weren't masked at all. The loss though only considers the prediction of the masked words and it ignores all the other words that are output by the network this is done to ensure that **more focus is given to predicting [MASK]ed values** so that it gets them correct and it increases context awareness.

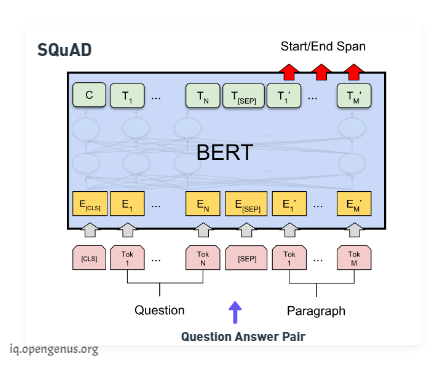
Once training is complete BERT has some notion of language as it's a language model.

# 2. Fine-tuning

We can now further train BERT on very specific NLP tasks for example let's take question answering, all we need to do is replace the fully connected output layers of the network with a fresh set of output layers that can basically output the answer to the question we want.



Then supervised training can be performed using a question answering dataset it won't take long since it's only the output parameters that are learned from scratch, the rest of the model parameters are just slightly fine-tuned and as a result training time is fast. This can be done for any NLP problem that is replace the output layers and then train with a specific dataset.

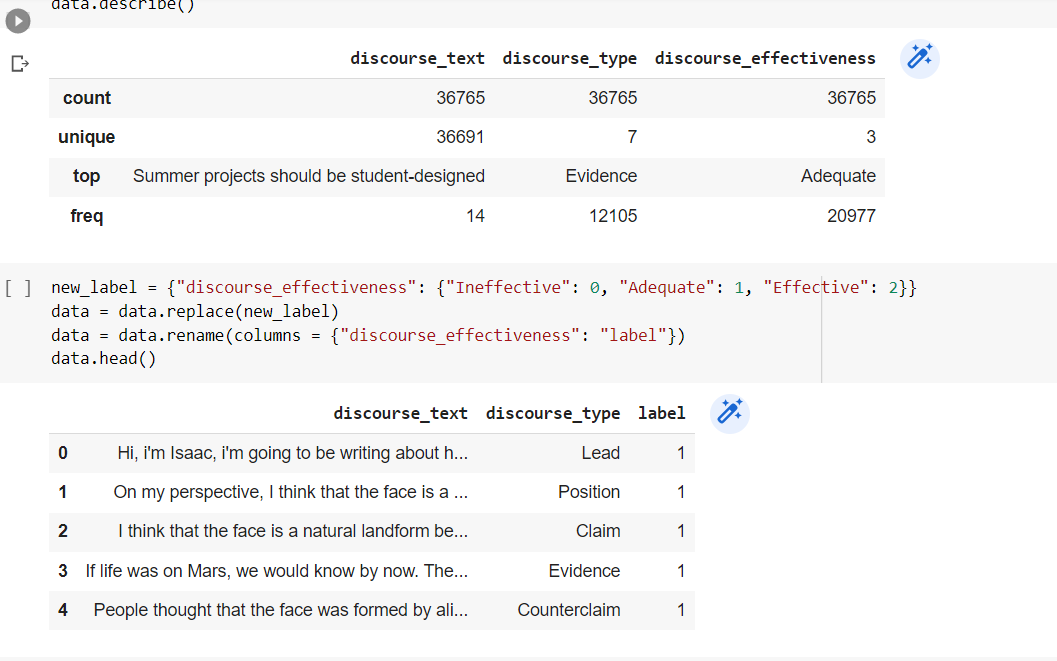


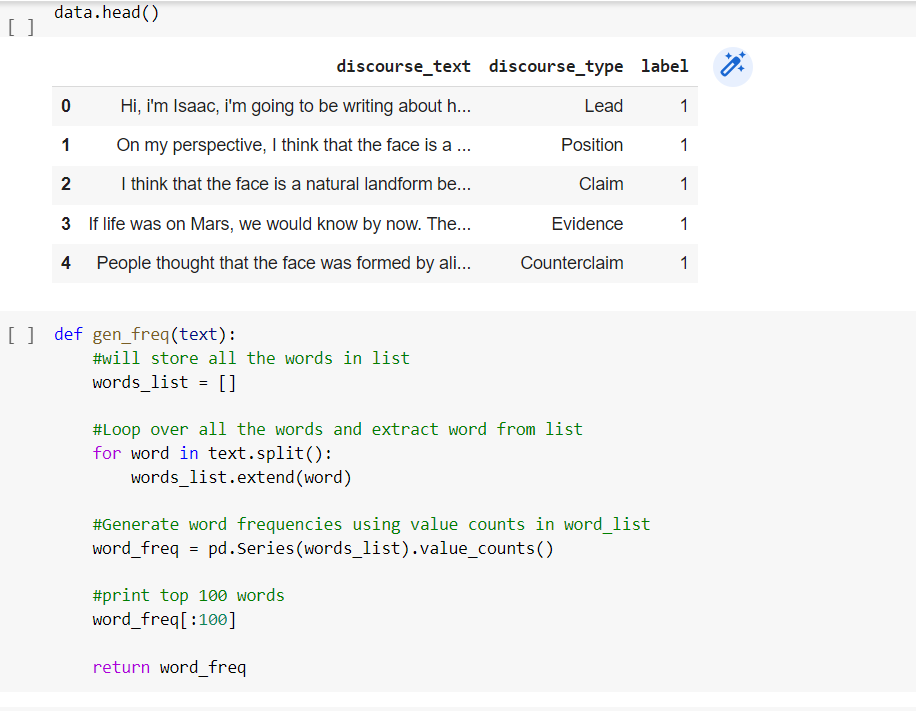
Now on the fine tuning phase, if we wanted to perform question-answering we would **train the model by modifying the inputs and the output layer**. We **pass in the question followed by a passage containing the answer as inputs** and in the **output layer we would output Start and the End words** that encapsulate the answer assuming that the answer is within the same span of text.

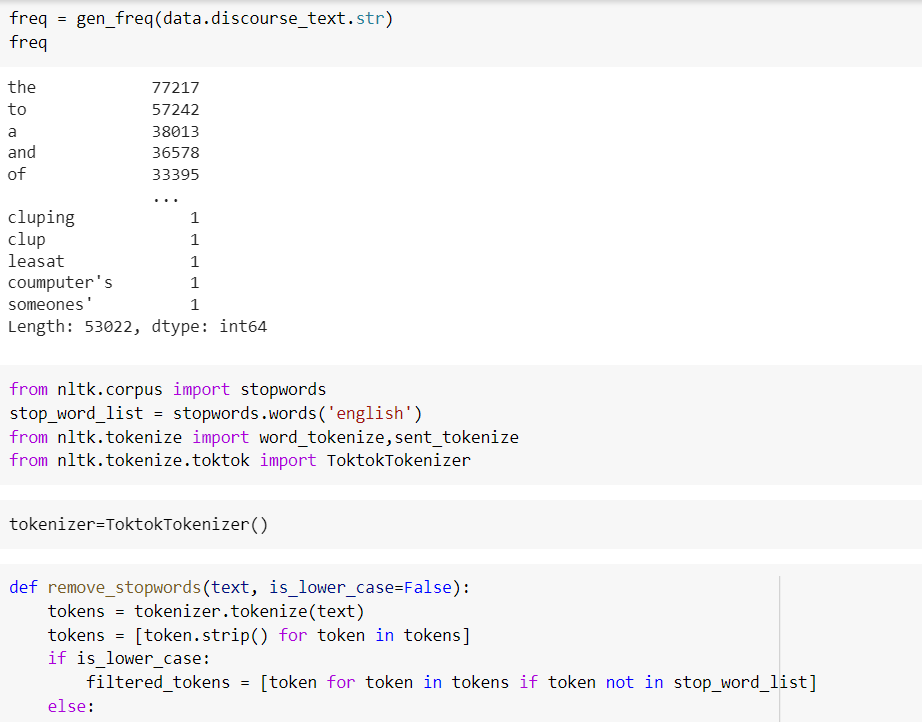
**Code:**

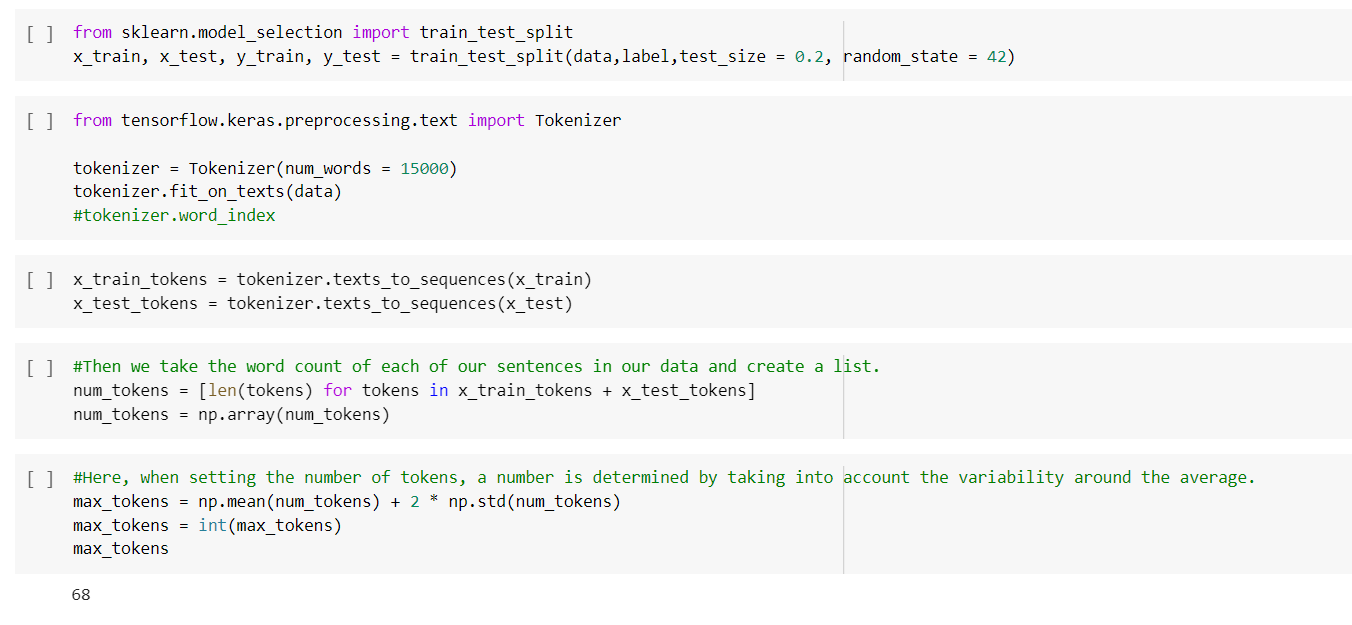


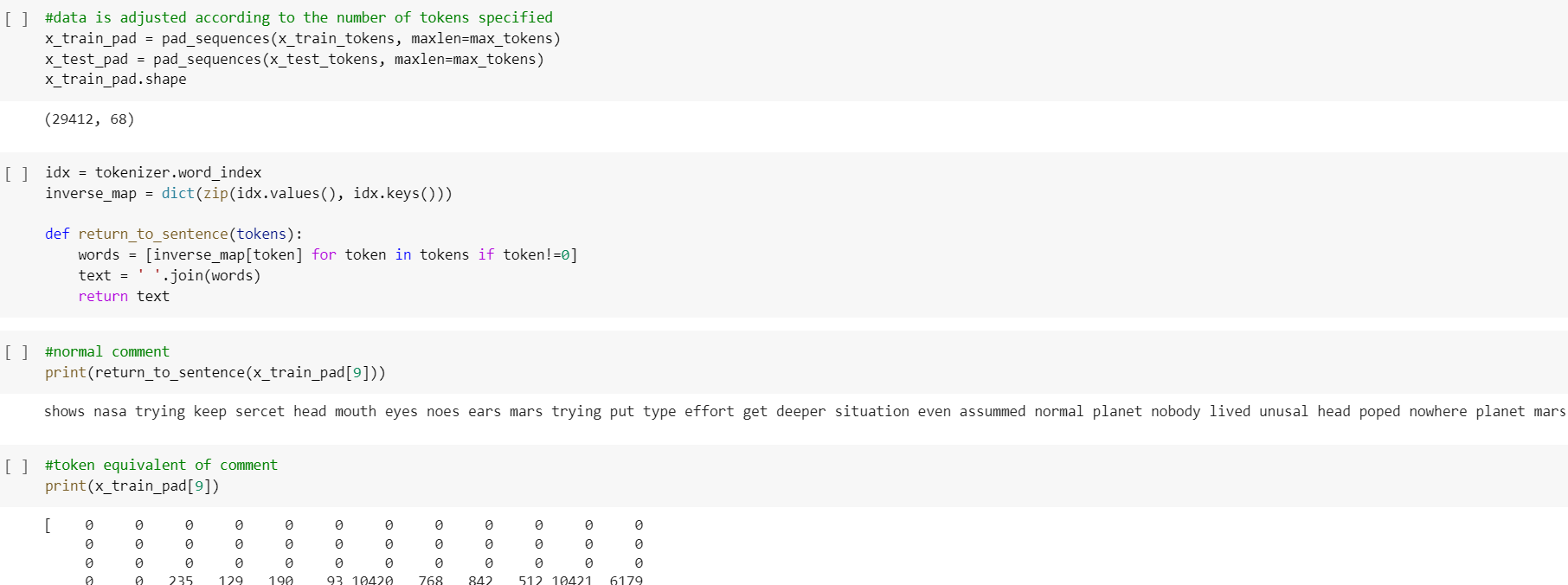




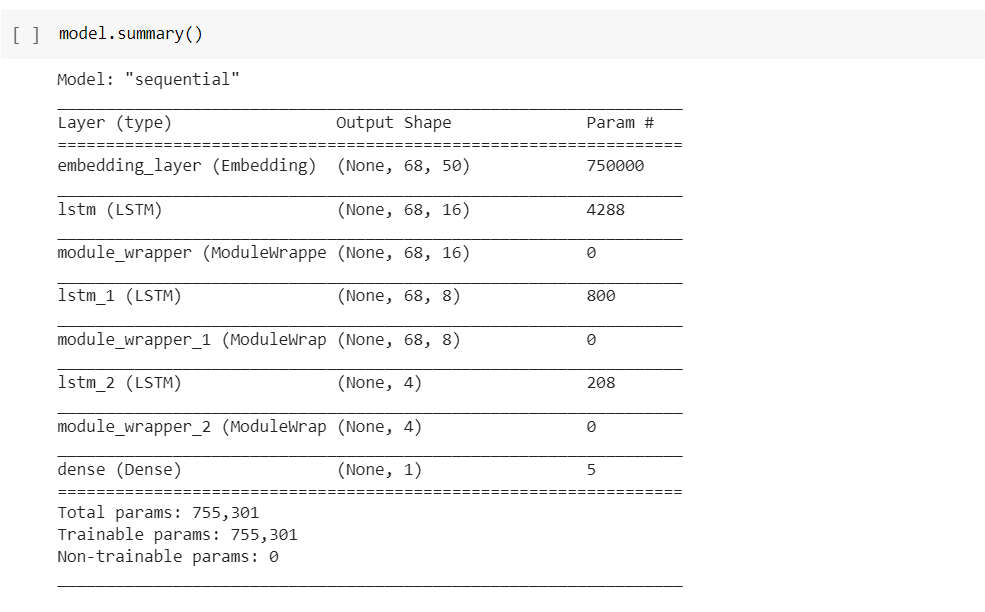




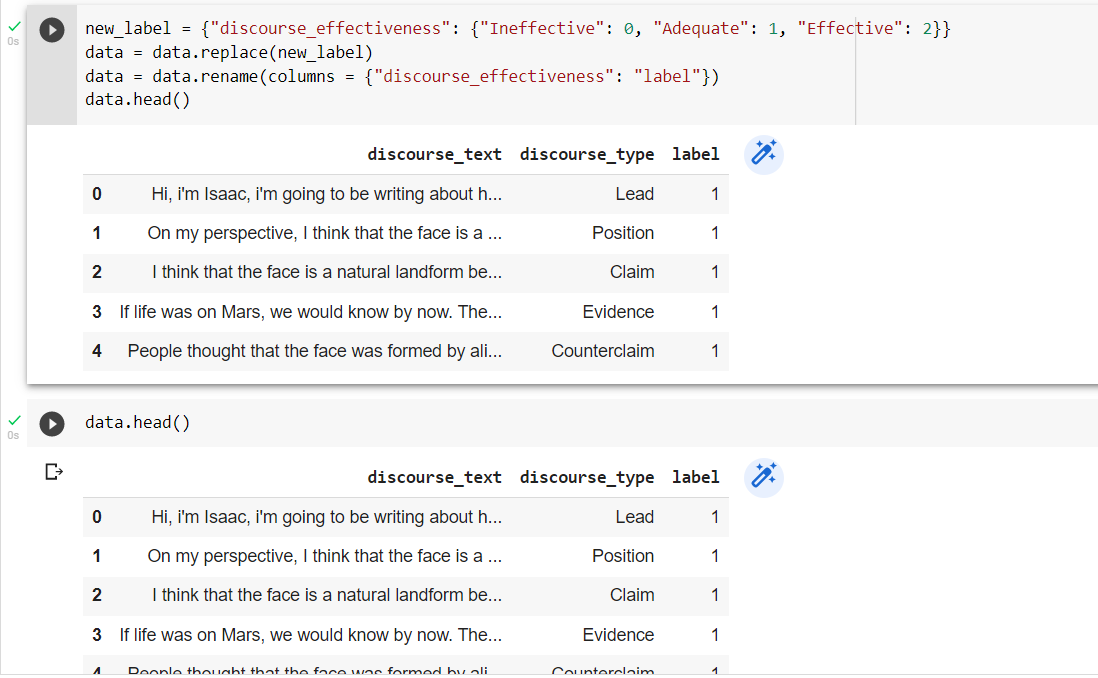


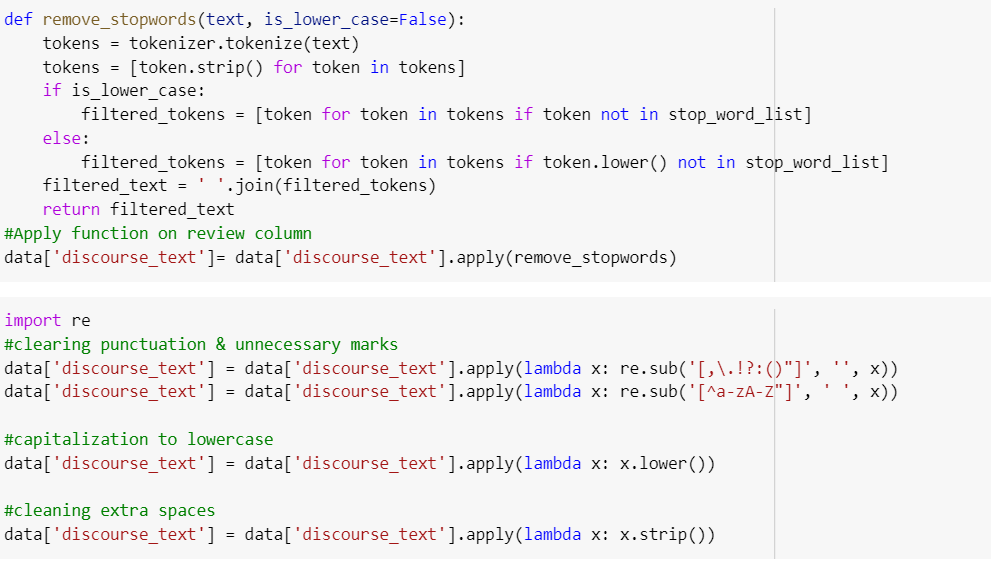


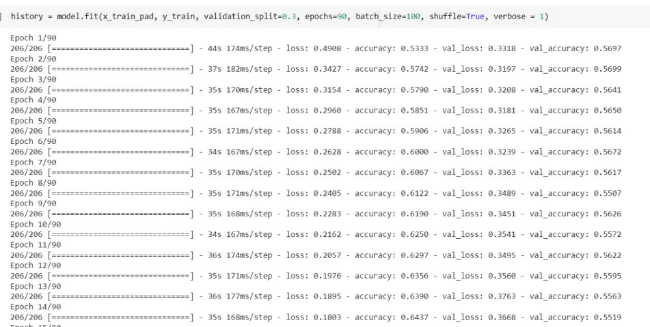


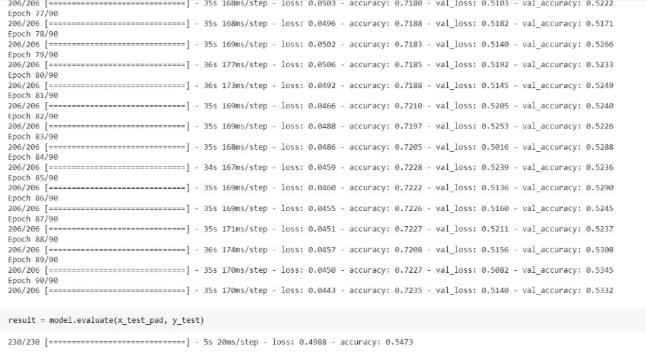


**7.Results**

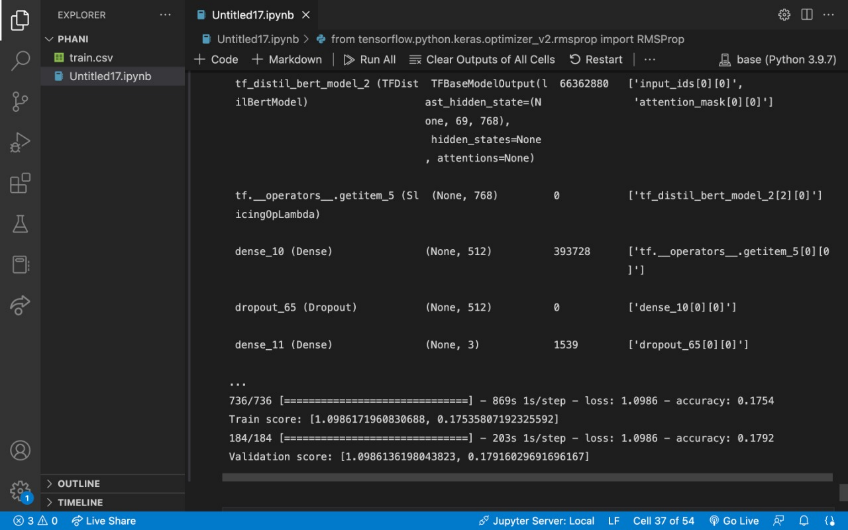


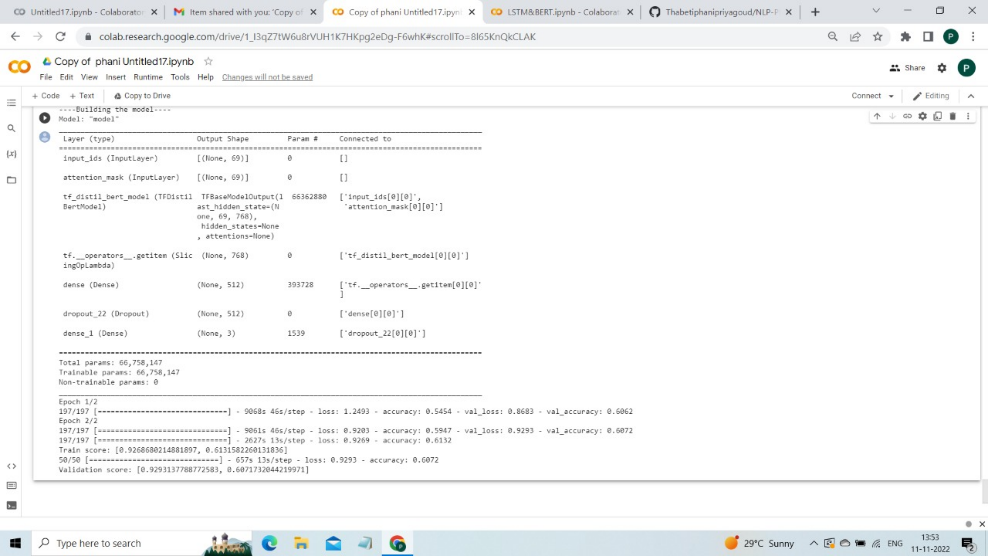






**BERT**





1. **Conclusion**

Student responses became a standard evaluation criterion in several fields like secondary education, academics, software recruitment's etc. As there are huge number of applicants or participants, it’s a hurdle for human evaluators to assess each response and predict it. It will kill huge amount of time and delay the process. Student Responses are collections of sentences and paragraphs that are useful to analyze the “effective”, “adequate”, or “ineffective” based on some parameters. Here used models are LSTM and Bert, between them lstm has good accuary of 0.72%

1. **References**

[1] Uto, M., Okano, M. (2020). Robust Neural Automated Essay Scoring Using Item Response Theory. In: Bittencourt, I., Cukurova, M., Muldner, K., Luckin, R., Millán, E. (eds) Artificial Intelligence in Education. AIED 2020. Lecture Notes in Computer Science(), vol 12163. Springer, Cham. https://doi.org/10.1007/978-3-030-52237-7\_44 <https://link.springer.com/chapter/10.1007/978-3-030-52237-7_44>

[2] Z. Chen and Y. Zhou, "Research on Automatic Essay Scoring of Composition Based on CNN and OR," 2019 2nd International Conference on Artificial Intelligence and Big Data (ICAIBD), 2019, pp. 13-18, doi: 10.1109/ICAIBD.2019.8837007.

[3] Ramesh, D., Sanampudi, S.K. An automated essay scoring systems: a systematic literature review. Artif Intell Rev 55, 2495–2527 (2022). https://doi.org/10.1007/s10462-021-10068-2 <https://link.springer.com/article/10.1007/s10462-021-10068-2>

[4] Christopher Ormerod(Septem,2019), “Language models and Automated Essay Scoring”, **arXiv:1909.09482** (cs). <https://arxiv.org/abs/1909.09482>

[5] Yongjie Wang, Chuan Wang (May,2022). “Use of bert for automated essay scoring”, **arXiv:2205.03835** (cs). <https://arxiv.org/abs/2205.03835>

[6] Guoxi Liang, Dongwon Jeong (2018). “Essay scoring using Neural networks”, DOI:[10.3390/sym10120682](http://dx.doi.org/10.3390/sym10120682). <https://www.researchgate.net/publication/329378585_Automated_Essay_Scoring_A_Siamese_Bidirectional_LSTM_Neural_Network_Architecture>

[7] Jumoke Eluwa, Shade O Kuyore (April,2022) . “Essay scoring Model Based on Gated Recurrent unit Technique”, DOI:[10.32628/IJSRSET229257](http://dx.doi.org/10.32628/IJSRSET229257)

<https://www.researchgate.net/publication/360440446_Essay_Scoring_Model_Based_on_Gated_Recurrent_Unit_Technique>

[8] JOUR,Eluwa, Jumoke,Kuyoro, Shade,O., Awodele,A., Ajayi,2022/04/30,323330“Essay Scoring Model Based on Gated Recurrent Unit Technique”,10.32628/IJSRSET229257,International Journal of Scientific Research in Science, Engineering and Technology, <https://www.researchgate.net/publication/360440446_Essay_Scoring_Model_Based_on_Gated_Recurrent_Unit_Technique/citation/download>

[9] Beseiso, M., Alzubi, O.A. & Rashaideh, H. A novel automated essay scoring approach for reliable higher educational assessments. J Comput High Educ 33, 727–746 (2021). https://doi.org/10.1007/s12528-021-09283-1 <https://link.springer.com/article/10.1007/s12528-021-09283-1>

[10] Farah Nadeem, Huy Nguyen, Yang Liu, and Mari Ostendorf. 2019. [Automated Essay Scoring with Discourse-Aware Neural Models](https://aclanthology.org/W19-4450). In Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications, pages 484–493, Florence, Italy. Association for Computational Linguistics. <https://aclanthology.org/W19-4450/>