Inclusion of Deep Learning in the Question-Answering Chatbot Development as a Tool for Digital/Smart Learning

A dissertation submitted in partial fulfilment of the requirements for the

degree of Master of Science

by Yash Mishra

Master's Degree in Data Science

Cardiff School of Technology

Cardiff Metropolitan University, Cardiff

May, 2022

Abstract:

Education teaches an individual how to grow as a person and a professional in their future and it all starts right from the beginning. Even being the most important factor, people tend to leave education at a very young age. According to research, majority of these cases occurred due to lack of attention or less interactive lectures. Besides these, another cause for this issue are time constraints for teaching professionals. Technology can reduce these numbers drastically using digital/smart learning. Lack of attention which includes less time provided to process the concepts and clearing questions related to the concepts. This study aims to create a question-answering chatbot model which can be used to answer students' queries individually. The question-answering chatbot model is based on XLNet model which is a type of transformer model used specifically for Natural Language Programming (NLP) related tasks.

Keywords: Question-Answering Chatbot, Chatbots, Smart Learning, Digital Learning, XLNet model, NLP.

Table of Contents

CHAPTER 1 INTRODUCTION: 1

1.1 Aims & Objectives: 2

CHAPTER 2 RELATED WORK: 3

2.1. Overview of Machine Learning: 4

2.1.1. Supervised Learning 4

2.1.2. Semi-supervised Learning 5

2.1.3. Unsupervised Learning 5

2.2. Transformer Model: 6

2.2.1. Encoder-Decoder 6

2.2.2. Training 7

2.2.3. Application 7

2.2.4. Limitations of Transformer: 8

2.3 Literature Survey: 8

2.4 Overview of Literature Survey: 14

CHAPTER 3 METHODOLOGY: 19

3.1. Software Architecture 19

3.1.1 Chatbot Environment 19

3.1.2 Class Diagram 20

3.1.3 System Modules 20

3.2. Software Implementation 21

3.2.1 Programming Languages and Libraries 21

3.2.2 Dataset Used for Training 22

3.3 Data Structure Used 23

3.4 Model Used 25

3.4.1 XLNet Model 25

3.4.2 Permutation Language Modelling 26

CHAPTER 4 RESULT: 28

4.1. Fine-Tuning: 28

4.2. Main Script: 32

CHAPTER 5 CONCLUSION AND FUTURE DIRECTIONS 36

5.1 Conclusion 36

5.2 Future Directions 37

REFERENCES 38

APPENDIX A: FINE TUNE CODE (FINE_TUNE.PY) 42

APPENDIX B: MAIN SCRIPT CODE (MAIN_SCRIPT.PY) 44

List of Tables:

Table 3.1 Literature Survey Overview..15

List of Figures:

Figure 2.1 Chatbot Architecture (E.Kasthuri, 2021)....................9
Figure 2.2 Process of Judgement (Liang, et al., 2020) 11
Figure 3.1 Chatbot Environment....................20
Figure 3.2 Class Diagram for Chatbot 20
Figure 3.3 Data structure of Squad v2.0 with first sample 23
Figure 3.4 Layer 1 of Data structure used....................23
Figure 3.5 Nested Layer of Data structure used 24
Figure 3.6 Another Nested Layer of Data structure used 25
Figure 4.1 Training Result Metrics of jkgrad/xlnet-base-squadv2 29
Figure 4.2 Loading Dataset....................29
Figure 4.3 Loading the tokenizer of selected model....................30
Figure 4.4 Loading the model....................30
Figure 4.5 Tokenizing the data with other parameters 30
Figure 4.6: Fine tuning the model with specific params....................31
Figure 4.7: XLNet Training Loss and Validation Loss 32
Figure 4.8: List of subjects available 32
Figure 4.9 get_subjects method 33
Figure 4.10 print_topics method....................33
Figure 4.11: List of Topics available 33
Figure 4.12 get_description method 34
Figure 4.13 Topic name with its context 34
Figure 4.14 1st Question asked on topic Punic Wars 34
Figure 4.15 2nd Question asked on topic Punic Wars 35
Figure 4.16 3rd Question asked on topic Greek Mythology....................35

CHAPTER 1 INTRODUCTION:

Society is becoming more "mobile-first" as a result of digitization (Oracle, n.d.). As messaging apps grow increasingly common, chatbots are becoming more important in this mobility-driven revolution (Oracle, n.d.). Intelligent conversational chatbots are increasingly being utilised as interfaces for mobile applications, and they're changing the way businesses and customers communicate (Oracle, n.d.). Chatbots enable businesses to communicate with consumers on a personal level without incurring the costs of hiring human representatives (Oracle, n.d.). Chatbots are nothing more than a programme that engages in conversation with the user and responds to their questions (Oracle, n.d.). Chatbots, often known as "bots," can understand natural language and respond to users' inquiries (Chat Compose, n.d.). These replies, on the other hand, might take the shape of a job, such as visiting a website, exhibiting an image, or watching a movie (Chat Compose, n.d.). Almost everyone has had a conversation with a chatbot at some point in their lives (Chat Compose, n.d.). Chatbots, in most circumstances, save a lot of time and human resources while also providing a superior answer without human mistake or emotion (Chat Compose, n.d.). This research is centred on the role of chatbots in digital and smart learning, as well as their development utilising deep learning.

Many prominent brands are already utilising AI chatbots to improve customer service and engage a growing number of consumers in order to stay relevant and visible (Botsify, n.d.). Other industries, such as educational institutes and instructors, are using chatbots in addition to corporations (Botsify, n.d.). Bots that can talk with students in all topics at the elementary, secondary, high school, and university levels are created using artificial intelligence and the newest conversational design (Botsify, n.d.). After determining the user's purpose, the chatbot must respond with the most appropriate response to the user's request, which might be (Chat Compose, n.d.):

1. a prepared and generic text answer (Chat Compose, n.d.)
2. a context based on data provided by the user (Chat Compose, n.d.)
3. data kept in databases (Chat Compose, n.d.)
4. a specified action (Chat Compose, n.d.)

The following are the primary benefits of utilising chatbots for businesses or other purposes:

1. Lowering the cost of staff who offered customer service (Chat Compose, n.d.)

2. Users receive immediate assistance and replies. (Chat Compose, n.d.)
3. An improved user experience (Chat Compose, n.d.)
4. When employing artificial intelligence systems, it is possible to learn about the user's preferences. (Chat Compose, n.d.)
5. Integration into many digital platforms is simple, with no need to download software or other tools. (Chat Compose, n.d.)

Chatbots are utilised in a number of industries, but their usage in education is particularly important (uPlanner, 2017). The majority of kids drop out of high school due to a lack of meaningful time with professors and counsellors (uPlanner, 2017). Chatbots can be utilised as virtual advisers to instructors, assisting them in focusing on each individual pupil (Botsify, n.d.). It will assist the school's principal in keeping track of each pupil and maintaining records (Botsify, n.d.). The most cost-effective and practical solution to this problem is chatbots (Botsify, n.d.). It can collect data in the process of assisting instructors, which will be utilised to analyse in the near future (Botsify, n.d.).

Because of the education industry's delayed adaptability and adoption of newly presented technologies, the introduction of AI to classrooms was eclipsed by other companies (Botsify, n.d.). However, more and more administrators and instructors are discovering this low-cost, high-value method of keeping pupils engaged and streamlining operations (Botsify, n.d.). Because children are growing up with tablets in their hands, teaching them about new technology and apps is only going to help them prepare for the future (Botsify, n.d.). Furthermore, technology may make the job of a teacher much easier (Botsify, n.d.).

1.1 Aims & Objectives:

The objectives of this dissertation are as follows:

a. the effectiveness of NLP integrated chatbot in digital learning
b. role of chatbots in evolving the traditional teaching methods
c. challenges pertaining adoption of chatbots in education system

CHAPTER 2 RELATED WORK:

A chatbot is a piece of computer program that mimics and processes human interaction, allowing people to converse with digital gadgets as if they were conversing with real people (Oracle, n.d.). Chatbots may be as basic as one-line programmes that react to a single inquiry, or they can be as complex as digital assistants that learn and adapt as they gather and analyse data to provide increasing levels of customisation (Oracle, n.d.).

The six primary categories of chatbots are as follows (Engati Team, n.d):

1. Menu/button based (Engati Team, n.d.)
2. Linguistic based (Engati Team, n.d.)
3. Keyword recognition-based (Engati Team, n.d.)
4. Machine Learning (Engati Team, n.d.)
5. The hybrid models (Engati Team, n.d.)
6. Voice bots (Engati Team, n.d.)

   A chatbot that is based on a menu or a button is the most basic kind currently in use (Engati Team, n.d.). Most often, these bots are glorified decision tree hierarchies with buttons that the user may interact with (Engati Team, n.d.). If you can anticipate the questions your clients will ask, a multilingual chatbot might be the answer (Engati Team, n.d.). If/then logic is used by linguistic or rules-based chatbots to build conversational flows (Engati Team, n.d.).

Keyword recognition-based chatbots, unlike menu-based chatbots, can listen to what users input and answer accordingly (Engati Team, n.d.). To determine how to offer a suitable answer to the user, these chatbots use customisable keywords and an AI application called Natural Language Processing (NLP) (Engati Team, n.d.). A contextual chatbot is significantly more sophisticated than the previous three bots (Engati Team, n.d.). Machine Learning is used in these chatbots (ML) Businesses are starting to adopt voice-based chatbots or voice bots to make conversational interfaces even more natural (Engati Team, n.d.).

We will focus on Machine Learning Chatbots utilising Deep Neural Networks in this dissertation. The artificial neural network (ANN) is a node-based network inspired by the simplicity of neurons in the brain (Kelleher, 2019). A deep neural network (DNN) is an artificial neural network that is capable of learning representations (Kelleher, 2019). Deep architecture frequently contains several layers between the input and output levels (Kelleher, 2019). It's a feedforward network, which means data is sent directly from the input to the output layer without looping back (Kelleher, 2019). Deep architectures may not necessitate the extraction of features; instead, the features themselves may compute them (Kelleher, 2019). Deep neural networks are the most effective for constituency parsing, information retrieval, sentiment analysis, machine translation, intent classification, spoken language understanding, Text categorization, with other tasks (Kelleher, 2019).

In computer programming, in order to grasp the message in natural language is the most critical part of constructing a chatbot (Kelleher, 2019). There are two types of chatbots that process rules: rule-based chatbots and artificial intelligence-based chatbots (Kelleher, 2019).

2.1. Overview of Machine Learning:

AI is the study of utilising data to predict and arrange what will happen in the future (Wikipedia, n.d.). We will undoubtedly conduct design acknowledgement and influence the forecast once we get additional facts (Wikipedia, n.d.). It is also known as Predictive Analysis or Predictive Modelling and is a crucial component of Artificial Intelligence (Wikipedia, n.d.). When these computations are updated with fresh data, they accumulate and modify their cycles for better execution, resulting in consistent data generation (Wikipedia, n.d.). Medical services, teaching, finance, train lines, and examination are all examples of cultural applications that use these computations (Wikipedia, n.d.). It entails PCs finding out how to do tasks without being explicitly programmed to do so (Wikipedia, n.d.). The data will be used to build the AI model, which will then be used to generate predictions (Wikipedia, n.d.).

2.1.1. Supervised Learning

Man-made reasoning and AI discipline is supervised learning, often known as managed AI (Wikipedia, n.d.). It is distinguished by its use of labelled datasets to produce calculations that correctly categorise data or predict outcomes (Wikipedia, n.d.). When input information is taken into the model during the cross-approval stage until the model is well fitted, the loads are adjusted (Wikipedia, n.d.). Organizations may use managed learning to deal with a wide range of large-scale certifiable issues, such as spam collection in an identifiable envelope from email (Wikipedia, n.d.).

The types of supervised learning are as follows:

1. Classification

In the classification arrangement, the AI model should make a decision based on the preparation data and determine which classification fresh perceptions belong to (Wikipedia, n.d.). Consider the case where we want to know whether a specific consumer buys Eggs or not (Wikipedia, n.d.). The model should be capable of learning from past data and anticipating future data (Wikipedia, n.d.).

2. Regression

The AI framework should evaluate and understand the relationships between components in relapse assignments (Wikipedia, n.d.). Relapse inquiry is extremely significant for anticipating and estimating since it focuses on one ward variable and a progression of other emerging components, such as the influence of compost and water on rural creation (Wikipedia, n.d.).

2.1.2. Semi-supervised Learning

Semi-supervised learning is a type of learning with a set number of labelled examples and a large number of unlabelled models (Wikipedia, n.d.). This type of learning problem is difficult to solve since neither administered nor unassisted learning calculations can deal with a mix of marked and untellable data (Wikipedia, n.d.). As a result, specialised semi-regulated learning methods are necessary (Wikipedia, n.d.).

Semi-supervised learning includes the following types (Wikipedia, n.d.):

1. Graph-based semi-supervised learning
2. XLNet model
3. Self-training
4. Low-density separation

2.1.3. Unsupervised Learning

Unsupervised learning is the application of artificial intelligence (AI) frameworks to discover patterns in informational indexes that include no organised or labelled data items (Wikipedia, n.d.). As a consequence, the computations may sort, identify, or perhaps cluster the valuable data in the informative collections (Wikipedia, n.d.). Overall, solo learning allows the framework to recognise patterns in informational indexes on its own (Wikipedia, n.d.). In unassisted learning, an AI framework will categorise unsorted data based on similarities and differences, even if no categories are supplied (Wikipedia, n.d.). In the case of confused managing errands, unaided learning computations outperform managed learning frameworks (Wikipedia, n.d.). Unaided learning is another method for examining AI (Wikipedia, n.d.).

2.2. Transformer Model:

A transformer is a deep learning model that employs the self-attention process to assign different weights to each element of the input data (Wikipedia, n.d.). Natural language processing and computer vision are two areas where it is widely used (Wikipedia, n.d.). Transformers, like recurrent neural networks (RNNs), are designed to analyse sequential input data, such as natural language, and are used for things like translation and text summarization (Wikipedia, n.d.). In contrast to RNNs, transformers handle the entire input at once (Wikipedia, n.d.). The attention mechanism provides context at any point in the input stream (Wikipedia, n.d.). The transformer does not have to handle one word at a time if the incoming input is a natural language phrase, for example (Wikipedia, n.d.). In comparison to RNNs, this allows for more parallelization, resulting in quicker training times (Wikipedia, n.d.).

2.2.1. Encoder-Decoder

The encoder-decoder architecture of the initial Transformer model, like prior seq2seq versions, was similar to that of preceding seq2seq versions (Wikipedia, n.d.). The encoder is made up of encoding layers that handle the input one by one, and the decoder is made up of decoding layers that handle the encoder's output in the similar way (Wikipedia, n.d.).

The role of each encoder layer is to construct encodings that include information about which portions of the inputs are related to one another (Wikipedia, n.d.). It delivers its encodings to the next encoder layer as inputs (Wikipedia, n.d.). Each decoder layer reverses the process, taking all of the encodings and combining them with the contextual data they've provided to create an output sequence (Wikipedia, n.d.). Each encoder and decoder layer uses an attention method to do this (Wikipedia, n.d.).

Attention weighs the importance of each input and draws on it to create the output for each one (Wikipedia, n.d.). Before collecting information from the encodings, each decoder layer has an extra attention mechanism that accumulates information from prior decoder outputs (Wikipedia, n.d.).

2.2.2. Training

Transformers frequently learn on their own, with unsupervised pretraining and supervised fine-tuning (Wikipedia, n.d.). Pretraining is often done on a bigger dataset than fine-tuning since labelled training data is limited (Wikipedia, n.d.). Tasks for pre-training and fine-tuning are frequently included (Wikipedia, n.d.):

- Reading comprehension
- Paraphrasing
- Sentiment analysis
- Next-sentence prediction
- Language modelling
- Question answering

2.2.3. Application

Natural language processing (NLP) applications such as machine translation and time series prediction have showed great potential for the transformer (Wikipedia, n.d.). GPT-3, XLNet, RoBERTa, and GPT-2 are among the pretrained models that show transformers' ability to perform a wide range of NLP-related tasks and have the potential to find real-world applications (Wikipedia, n.d.). These may consist of:

- Video understanding
- Named entity recognition (NER)
- Document summarization
- Biological sequence analysis
- Document generation
- Machine translation

2.2.4. Limitations of Transformer:

Transformer is unquestionably superior to RNN-based seq2seq models (Joshi, 2019). However, it has its own number of weaknesses (Joshi, 2019):

- Consideration can now only handle text strings with specified lengths. Before being fed into the system as input, the text must be divided into a specified number of segments or chunks (Joshi, 2019).
- Text chunking produces context fragmentation. When a phrase is split down the middle, for example, a considerable amount of context is lost. To put it another way, the text is separated without regard for the phrase or any other semantic barrier (Joshi, 2019).

In order to resolve these issues, the Transformer-XL was born (Joshi, 2019). XLNet is the extension of Transformer-XL (Joshi, 2019). We will learn more about it in this paper ahead.

2.3 Literature Survey:

A chatbot is a computer software that mimics natural language conversation between a user and the machine. LINE, Facebook Messenger, and other chat networks have APIs that have been made available to developers. Mathawan Jaiwai and his colleagues presented a theoretical concept for constructing a chatbot structure as a question-answer system using LINE services, a Deep Neural Network Model with text classification is used to create answers. You may use the Messaging API to set up a two-way communication channel between your app and LINE users (Mathawan, et al., 2021).

Massive Open Online Courses, or MOOCs, have increased need for individualised student support. Question and Answer answers can help students study more and drop out less. Task-oriented chatbot models are more beneficial in the educational context, according to a study of chatbot models. As a solution to this challenge, Rachid Karra and Abdelali Lasfar suggested an integrated chatbot on the Open edX platform in 2021. The model was created to complete a question-answering assignment. TensorFlow and NTLK were used to do the training (Motahhir, S. & Bossoufi, B, 2021).

As we all know, English is widely spoken around the world. Some countries are still having difficulty learning the universal language. Teachers who teach language courses to a large class can benefit greatly from chatbots. It might be difficult to focus on each student and resolve questions by offering personalised comments, especially in underdeveloped nations and countries with a big youthful population. Kee Man Chuah and his colleague Muhammad Kamarul Kabilan demonstrated how instructors might benefit from employing chatbots for learning in a mobile setting. They also concluded that, from a methodological standpoint, this study demonstrated that chatbot dialogues may be analysed using the Community of Inquiry framework to identify degrees of social, cognitive, and instructional presence. The findings suggest that chatbots might play an important role in improving social and educational presence (Kee-Man Chuah & Muhammad Kamarul Kabilan, 2021).

In 2021, E. Kasthuri, a student at Francis Xavier Engineering College, and her teacher, S. Balaji, attempted to create an instructional chatbot using a deep learning method. Their model was quite successful and widely investigated. During pandemics, when student-teacher interaction was at an all-time low, that method was designed for academic objectives. It can serve as a virtual tutoring assistance for individuals (E. Kasthuri, 2021).

A chatbot can be an active participant in virtual spoken language, understanding users' needs and attempting to meet them through casual interactions. Chatbots communicate with users in a similar way, but individuals talk. It's ideal for selling, customer service, and searching, as well as e-learning. Figure 2.1 depicts the overall architecture for creating a highly accurate chatbot. Chatbots may also respond to sophisticated requests using this framework (E.Kasthuri, 2021).

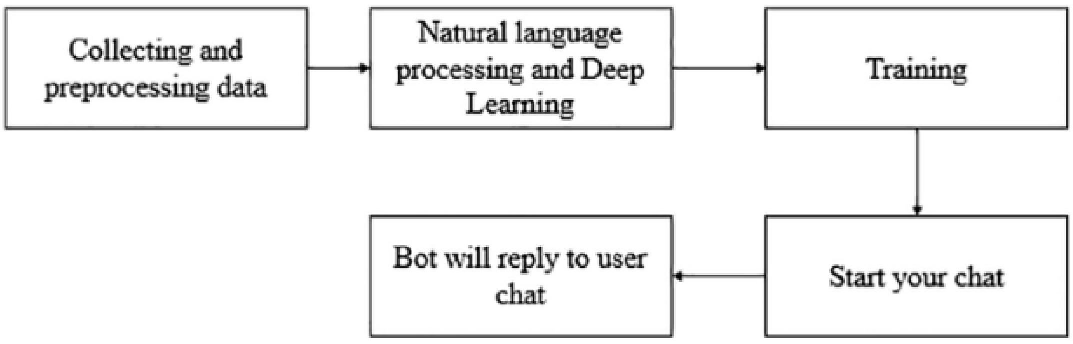

Collecting and preprocessing data
Natural language processing and Deep Learning
Training
Start your chat
Bot will reply to user chat

Figure 2.1 Chatbot Architecture (E.Kasthuri, 2021)

Conversational AI has become a part of our daily lives in the modern world, whether we're shopping, banking, or contacting customer support. Chatbots are commonly found in online or mobile apps such as Siri, Google Assistant, Alexa, and others. It may handle a variety of use cases depending on the needs of the company. A multi-level model for chatbots can include a machine learning classifier, such as SVM, phrase similarities, such as cosine similarity, and the deep learning seq2seq model (Ali, 2021).

The retrieval-based chatbot is similar to a search engine in that it first saves the dialogue database and creates an index. Following the user's input, search and match the relevant response content in the database, and then output. The Encoder-Decoder framework is now employed in most deep learning-based chatbot solutions. The Encoder-Decoder framework makes it simple to build a chatbot and enhance it. The use of chatbots has drastically decreased the requirement for human assistance (EleniAdamopoulou, 2020).

Chatbots are separated into two groups based on their implementation technology: retrieval chat robots and generative chat robots. After the user inputs the statement, the retrieval chatbot searches and finds the matching response material in the database and outputs it, whereas the generative chatbot generates a sentence answer using particular technological methods after the user provides the stat. The researchers intended to explore if chatbots might trick clients into thinking they were real humans, as seen in Figure 2.2. With the introduction of the ELIZA chatbot in 1966, a great deal of effort was put into creating a chatbot that could pass the Turing test. Many strategies for generating chatbots and various technologies for producing chatbots have arisen as a consequence of their efforts. A chatbot should study the inquiry posed by a customer appropriately and build a suitable answer to flawlessly resemble a human response. A chatbot's purpose is to create a user-friendly interface with natural language communication. User's input may be used to determine whether the user's problem should be handled by a deep learning module or a retrieval module (Liang, et al., 2020).

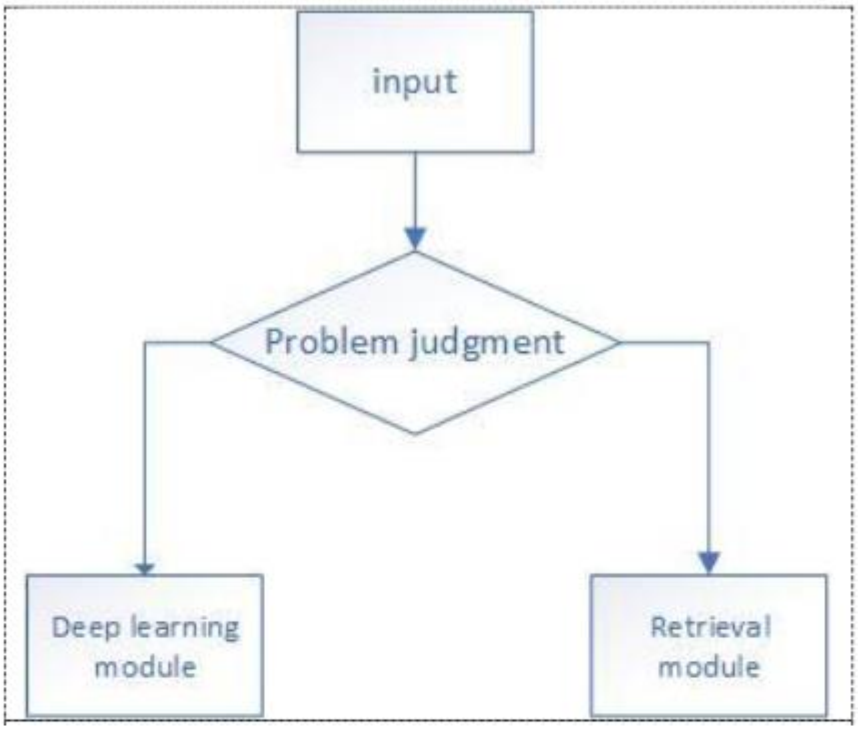


Figure 2.2 Process of Judgement (Liang, et al., 2020)

The development and training of a clever conversational agent has long been a goal of humanity and a significant task for scientists and programmers. Smart chatbots can learn via dialogue much like people thanks to deep learning and neural networks. On a learner's own mobile device, chatbots are available 24 hours a day, seven days a week. They give learning at a suitable time and location, as well as in little chunks or short bits, which is ideal for current students' fast-paced lives. They also offer instant feedback, check for comprehension, and patiently repeat the knowledge that must be grasped, remembered, and put into practise several times. They create welcoming surroundings free of expectations or judgement (Irina Dokukina, 2020).

It is impossible to see chatbots completely replacing human teachers. Some researchers have attempted to compare the knowledge imparted by a chatbot against a human teacher, highlighting the concerns with the novelty effect. Chatbots have been deployed as assistants, with their primary purpose being to assist with certain activities. Teachers' participation in the application of this technology is critical to its success. Chatbots provide users a false feeling of anonymity, and users are nastier to them than they would be to a real interlocutor. It underlines the importance of the chatbot's personality in establishing an empathetic interaction with the user (José Quiroga Pérez, 2020).

Chatbots are being used for learning not just in high school and junior college, but also in other professions, such as nursing students undergoing nursing training. According to study published in 2021 by Ching-Yi Chang, Meei-Ling Gau, and Gwo-Jen Hwang, this technique improved nursing students' learning outcomes in terms of obstetric vaccination knowledge, self-efficacy, and learning experience. Developing nurses' knowledge and competence in vaccination delivery is a critical problem in prenatal education for protecting pregnant women and babies from illness. As a result, vaccination knowledge is a basic and required training programme for nursing students (Ching-Yi Chang, 2020).

Conversational AI bots are one of the many huge promises of information technology. They were established as a new interface to replace applications and improve website visits by using chat to streamline interactions between services and end-users. Natural Language Processing (NLP) and user inquiries are frequently handled by these bots. These emotions, on the other hand, do not always take the form of writing; they might also take the form of actions, such as purchasing a vacation or checking emails. As a result, various industries, like customer service and banking, are adopting this technology to make it easier for customers to use their systems. Another area where chatbots might have a significant impact is education. Each kid in a classroom has various learning requirements and interests. As a result, everyone needs the assistance of a professional instructor. Unfortunately, even at the most advanced institutions throughout the world, this sort of service is not offered. Students nowadays frequently utilise messaging services to connect with one another and, on occasion, with their professors, which are typical features in platforms such as Google Classrooms and other class management systems. This type of feature is designed to allow students to ask questions and receive responses that will help them gain a better understanding and assist their study outside of the classroom. Research on highly adaptable edu-chatbots was published by Tarek AIT BAHA and his colleagues. They developed a system that receives user communications through an instant messaging platform. A plain text or audio message can be received. To convert audio to text, a speech recognizer module is utilised. The recognised speech is delivered to an in-house NLP Engine for Intent Recognition tasks through Xatkit's runtime engine, which is built on the CamemBERT architecture. When the runtime gets the recognised intent, it does a lookup to obtain a list of actions and events connected with it. Finally, the Speech Synthesizer module will translate the text response to speech, while a contents projector module will display non-messaging activities. Based on the input, an animated 3D avatar will respond with exact gestures, mimics, and carry on the conversation in synchrony with the audio (Tarek AITBAHA, 2022).

Between the input and output levels, deep architectures usually include numerous layers. In feedforward networks, data is sent from the input layer to the output layer without looping back. Deep architectures do not require a human to extract features and can calculate them independently. Deep neural networks perform well in constituency parsing, sentiment analysis, information retrieval, machine translation, spoken language interpretation, intent categorization, text classification, and other tasks (Kelleher, 2019).

According to the findings, the instructional chatbot for the Facebook Messenger platform is effective. A method for determining discoverability and features such as language, topic content, and developer platform (Pavel Smutny, 2020).

The system creates an academic chatbot using NLP and ML that may be utilised by a variety of educational institutions. There are two options available: audio and text. Instead of being placed on the inquiry disk's waiting list, consumers can communicate with the bot. The same question is asked in several ways to ensure accuracy (S. Kumari, 2020).

For the patients, ML algorithms were used to create a framework that works as a virtual assistant. By just communicating with them, it can forecast symptoms, recommend doctors, and assess the patient's therapies. Effective in looking after the health of the patient, with optimistic outcomes (M. Polignano, 2020).

A framework for intensive study to improve the virtual patient's communication abilities. The discovery of domain-specific word embeddings was first made possible by deep neural networks. With an accuracy of 81%, empirical data on a domestic corpus indicated that our strategy beat one-of-a-kind algorithms (J. E. Zini, 2019).

Users may utilise Query Response Creation to import and execute major objectives on relevant files, quickly assemble solutions, and generate queries. Many cleaning documents and pre-processing techniques use the imported file as a starting point. The query-generation module determines the query-solution pairings, while the questions replying module uses a ranking algorithm and to extract the optimal response from the knowledge store using neural networks. The quality of the Quiz period was improved by identifying statements that are most likely to result in appropriate quiz questions. This was accomplished by removing non-declarative lines and employing content awareness across the content of the file, and using a series of scoring algorithms to score sentences, resulting in a set of query-answer pairings that were more germane to the issue being counted (Sreelakshmi, 2019).

Through the scientific evaluation of chatbots that are cloud-based, an ECHO approach was built and shown. The impact of a similar evaluation on local cloud chatbots will be investigated and archived by ECHO. To confirm ECHO's efficacy, a comparative study with two separate questions of three common cloud-based chatbots of three difficulty levels should be employed (A. R. Mohammad Forkan, 2020).

To address the present machine's lack of availability, a chatbot device for college enquiries is being developed. It makes it simpler to avoid putting the responsibility of recognising academic information on the operational group of workers. Because it's linked to the NLP module, this chatbot may mimic human discussions and act as a personal assistant, guiding users through the university's material. This chatbot machine has an advantage since it can respond not only with sentences, but also with images and links. Any query that does not have a response is then emailed to the administrator, who receives the message at the bottom of the page. The administrator then updates the query on the device. The chatbot is simple to use since the administrator may watch and edit dialogues by supplying the right identity and password. As the administrator checks the interaction, there is no possibility of extraneous information being presented (Inamdar & , 2019).

Chatbots have been extensively researched, and new approaches have been devised to demonstrate how students may most effectively learn and practise a new language in the target language. However, despite their widespread usage as a useful tool for connecting with corporate clients, chatbots have yet to be completely researched and realised for the language learning community (Fryer & Nakao, 2019).

2.4 Overview of Literature Survey:

This section shows an overview on the literature survey performed for this research. For each research paper referred, the name of the article, goal of that research paper, method which is used in that research paper and main findings presented by that particular researched paper is mentioned in a tabular format.

Table 2.1 Literature Survey Overview

Article	Goal	Method	Main Findings
(Mathawan, et al., 2021)	An approach for building a question-answering system via chatbot technology	Quantitative analysis	Successfully created APP converting language from Thai to English using LINE integration using Deep Neural Network with text classification.
(Motahhir, S. & Bossoufi, B, 2021)	To create Q&A chatbot using Open edX Platform	Quantitative, between-subjects ex	Proposed Integration of chat bot in open edx platform carried out using deep learning. Also focusing on knowledge transfer by adding blocks of knowledge by using pre trained models also enabling the chatbot to recommend chapters and study material based on gap between users' knowledge
(Kee-Man Chuah & Muhammad Kamarul Kabilan, 2021)	To find the teachers views on use of chatbot in teaching English language to the students	Qualitative, Exploratory Method	Willingness of Teachers to adopt the use of chatbots to enhance the learning experience along with justifying the potential of chatbots by increasing social and teaching presence. Chatbots have acted as teaching assistants and guide students by stimulating human like interactions, also identifying chatbots minimal capacity to provide feedback
(E. Kasthuri, 2021)	Design chatbot for MATLAB dataset to answer student's queries given to bot	Quantitative Approach of Analysis	The authors aimed at developing a chatbot that can take instruction through voice and give relevant answers, thereby clearing doubts of students, using LSTM ML algorithm which has best user rate when compared to other algorithms like Naive Bayes, Canopy and J48

(E.Kasthuri, 2021)	The purpose of the study is supporting students and answering their queries in live chats and also guide users by series of assessments and helping them with course content	Quantitative	The developed system is a smart and clever answer generator that answers complex queries using Long short term memory algorithm and determines its performance on the basis of measures like precision, accuracy etc.
(Liang, et al., 2020)	This research was based on adoption of digital campus robots consisting of three modules i.e., chat module, campus information module and virtual teaching assistant module aimed at	Quantitative approach	The prepared bot consisting of three modules makes use of encoder decoder framework, thereby allowing new students to get answers to questions related to their campus. The bot gives timely and accurate reply to the questions also reducing the need of additional human resource

	improving student's user experience		
(Ali, 2021)	The paper aims at designing a generalized chatbot that can be used in any domain depending on business requirements	Qualitative and Quantitative both	The multilayer HMLM architecture consists of a machine learning classifying model like Support Vector Machine to check sentence similarity using cosine similarity and further a deep learning sequence to sequence model consisting of Matching Layer and Generative layer. Each level after being trained on multiple and different datasets which helped achieve better classifier score and blue score. The model also consists of embedded entity recognition module which helps in finding entities from selective sentences
(Irina Dokukina, 2020)	The smart chatbots helps ESL learners by creating multiple scenarios and allows the trainers to multitask and making their work more effective	Qualitative	Chatbots have many advantages like 24/7 availability and can be used in any place, thereby suiting the modern fast paced lives and also provides feedback immediately and also providing safe environment for learners free from biases.
(José Quiroga Pérez, 2020)	The aim of research is to develop the chatbot that	Qualitative PRISMA approach	The research discovered two distinct classifications among the chatbots Service Oriented and Teaching Oriented which have been tested among various age groups. The

	can act as educational agents	studying 80 researches	teaching oriented chatbot could be applied at different levels where it can assume the role of either an assistant or a complete educator. Such chatbots also help to identify the mood of the learner and improve the affectivity and also help with disabled children
(Kumar, 2020)	The aim of the study is to study and compare various technologies for developing chatbots and then propose the best system	Qualitative	The study compared several NLP algorithms and also emphasized the importance of dataset and algorithm both while designing an algorithm and ultimately developed a basic Q&A answering chatbot that could fetch answers from the database using Seq2seq model.
(Tarek AITBAHA, 2022)	The aim of study is to use and test the chatbot framework Xatkit to develop a chatbot in education industry	Quantitative	The built chatbot using Xatkit framework first accepts a message from user inform of text, speech -which then get converted into text using speech recognizer model, which when forwarded to NLP engine based on CamemBERT architecture performs lookup to find associated actions and responses and further generates a response through speech synthesizer and performs actions on the projected module, thereby making the digital avatar act accordingly.

CHAPTER 3 METHODOLOGY:

3.1. Software Architecture

The software architecture section is divided into 3 parts naming “Chatbot Environment”, explaining the environment of chatbot in this research, “Class Diagram” mentioning the structure of the question-answering chatbot to be followed, and “System Modules”, describing the modules used in the class diagram.

3.1.1 Chatbot Environment

The chatbot environment shown in Figure 3.1 is made up of several components. User communications are one of them. They're a dynamic input that our model can get at any moment. They are made up of a string that represents the user's real text, along with some metadata structured to have additional information that can describe the text more understandingly. Metadata may contain any images or the links sent by the user, the timestamp of the message sent, the platform from which it is sent, etc. The chatbot only has read-only rights and it cannot change the text sent by the user.

Another part of the chatbot environment in the Figure 3.1 is the trained model. It is the model that we trained using the Transformer XLNet model and the dataset collected. It is capable of handling the user and giving all the answers without human intervention. It can give suggestions to users on the basis of their queries in order to help them in every possible way.

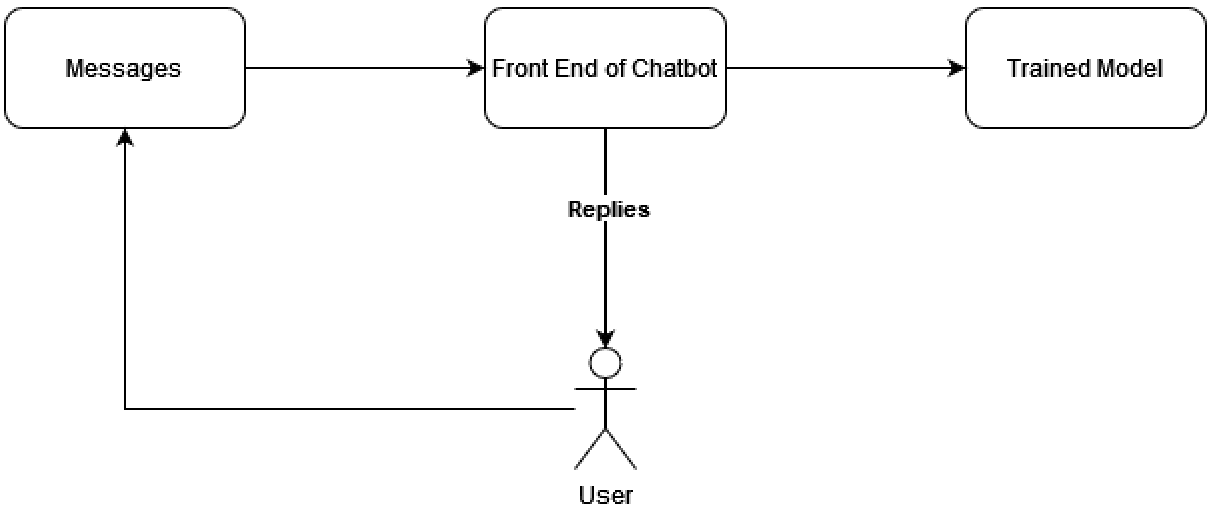


Figure 3.1 Chatbot Environment

3.1.2 Class Diagram

The chatbot, as can be seen in this Figure 3.2, is made up of various components, each of which solves a specific problem. The component at the top of the Figure 3.2 labelled User is essentially a generic user application that makes use of the chatbot. It might be as simple as a terminal client or as sophisticated as a dynamic online listener. The chatbot has 4 main components: Intent Classifier, Action Planner, Context Manager, and Action Executor.

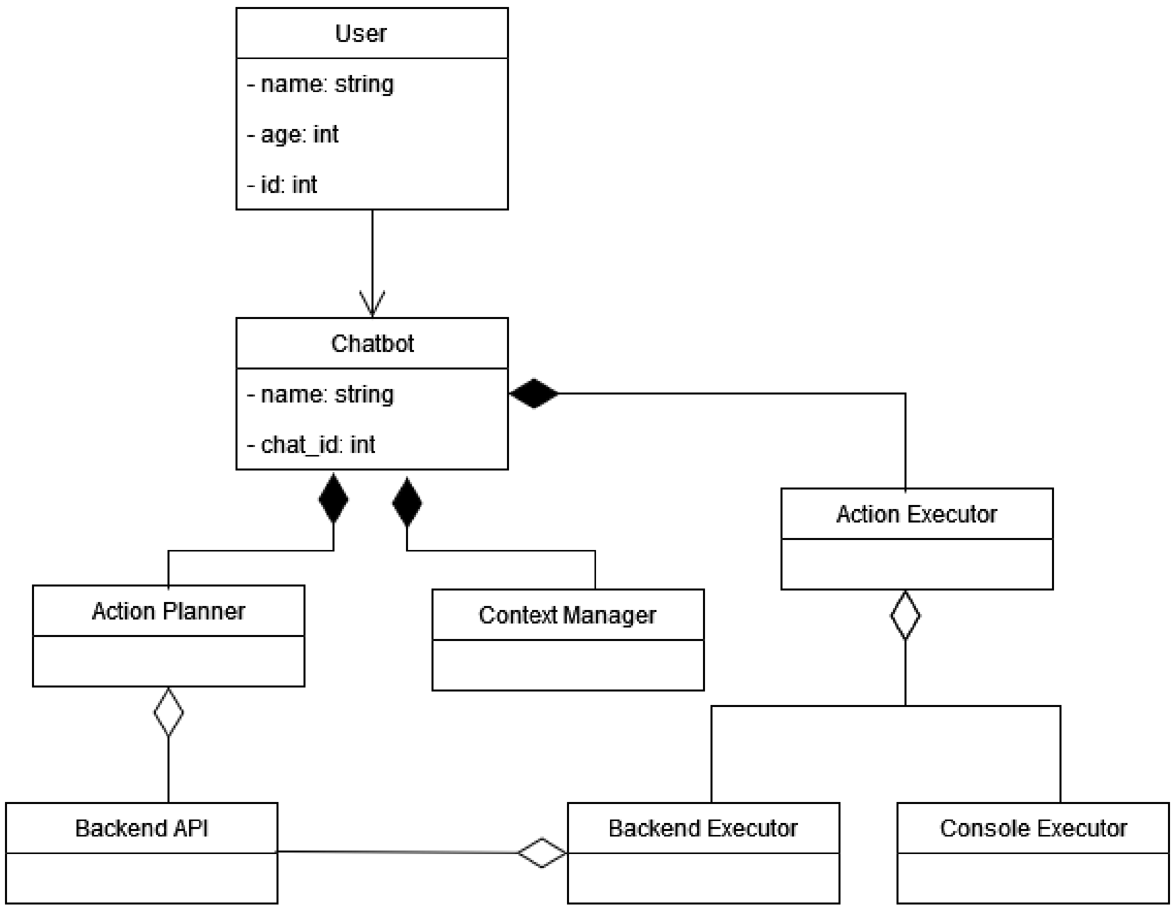


Figure 3.2 Class Diagram for Chatbot

3.1.3 System Modules

1. Context Manager:

   The module is in charge of the framework that organizes contexts. In this scenario, a context is described as a series of messages, as well as the information associated with them, that are relevant for assisting a user with a specific problem. This closely relates to our intuitive understanding of a single discussion in human words.

2. Action Planner:

   The purpose of the action planner is to determine which actions the agent should do given a probability vector. It is in charge of observing the status of the backend and creating a payload containing the actions to be performed and in what order. The payload's format will be a JSON object in this scenario. It's also in charge of producing the actual response that will be sent to the user.
3. Action Executor:

   This module is in charge of executing sequences of instructions, as the name indicates. It works exclusively just on the output of the Action planner and, on the backend, delivers the messages and executes the scheduled instructions.

3.2. Software Implementation

This section will explain the various libraries and datasets that were chosen for the project and go into the specific implementation details of the finished system.

3.2.1 Programming Languages and Libraries

For this dissertation, we have used Python as our main language in order to implement the chatbot. Python is the best choice for machine learning and AI applications because to its simplicity and consistency, as well as access to good AI and machine learning tools and frameworks, flexibility, platform independence, and a big community (Beklemysheva, n.d.). Although, this implementation can also be done with other languages such as C# as well.

Some of the major libraries used in this implementation are:

a. PyTorch: PyTorch is an open-source machine learning framework based on the Torch library. It makes the learning for models much easier, with less code (Wikipedia, n.d.).
b. Tensorflow: We can use the TensorFlow data API to create sophisticated input pipelines out of basic, reusable components.

c. Pandas: Pandas is a data transformation and analysis software package for the Python programming language. Data structures and functions for manipulating numerical tables are available in Pandas (pandas, n.d.).

d. Numpy: NumPy is a Python library that adds support for massive, multi-dimensional arrays and matrices, as well as a vast variety of high-level mathematical functions, to the Python programming language (Numpy, n.d.).

e. Huggingface: Huggingface provides open-source NLP technologies that are used in this implementation.

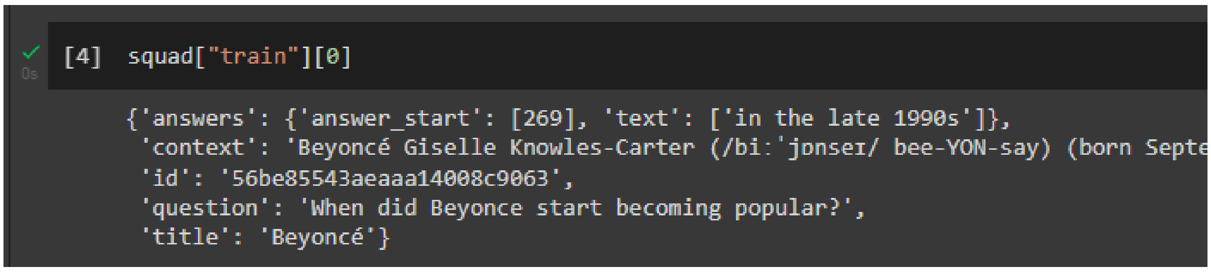
f. nltk: The Natural Language Toolkit is a collection of libraries and applications for processing natural language in English symbolically and statistically.

Many more minor libraries are also used like json, time, etc.

3.2.2 Dataset Used for Training

For the implemented chatbot training, the Stanford Question Answering Dataset has been used. The Stanford Question Answering Dataset (SQuAD) is a reading comprehension dataset comprised of questions submitted by crowd workers on a collection of Wikipedia articles, with each question's response consisting of a text segment, or span, from the relevant reading passage, or the question being unanswerable. The SQuAD has 2 versions where the v1.1 overall dataset size is 94.06 MiB and has been split into ‘train’ and ‘validation’ whereas ‘train’ has 87599 examples and ‘validation’ has 10570 examples. Similarly, v2.0 has an overall dataset size is 148.54 MiB and has been split into ‘train’ and ‘validation’ whereas ‘train’ has 130319 examples and ‘validation’ has 11873 examples (TensorFlow, n.d.).

Figure 3.3 shows squad v2.0 structure of the data in which it is present and first sample of the dataset. There are columns of title, which has the title of the context, context, from which the answer is provided, question, id and answers with the text and its starting point in that particular context.



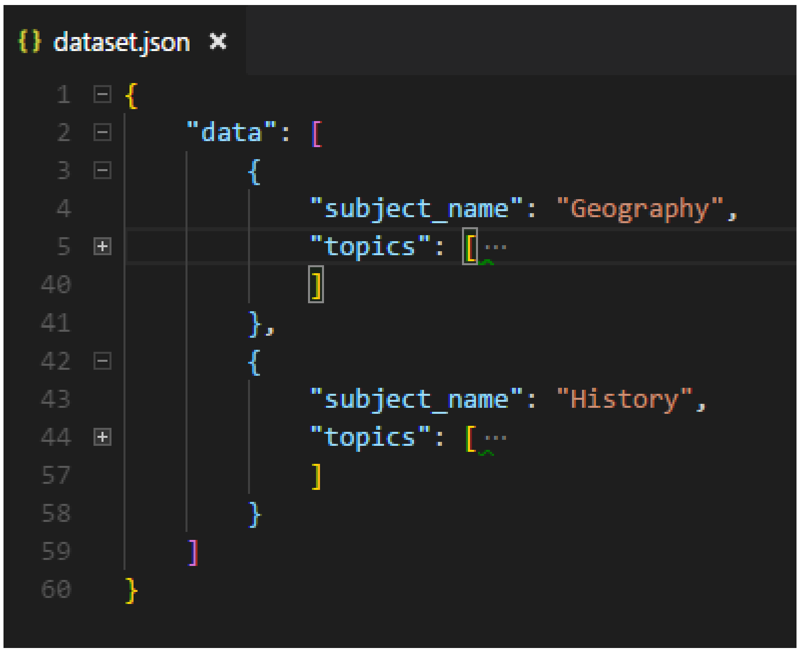
```
[4]  squad["train"][0]

     {'answers': {'answer_start': [269], 'text': ['in the late 1990s']},
      'context': 'Beyoncé Giselle Knowles-Carter (/biːˈjɒnseɪ/ bee-YON-say) (born Septe
      'id': '56be85543aeaaa14008c9063',
      'question': 'When did Beyonce start becoming popular?',
      'title': 'Beyoncé'}
```

Figure 3.3 Data structure of Squad v2.0 with first sample

3.3 Data Structure Used

In order to access the data easily, a simple data structure is used. Usually, a recommended database for the Question Answering chatbot is SQL. For the shorter version, instead, we have used a JSON file with a demo dataset in it.

As shown in the Figure 3.4, the layer 1 of the JSON is comprised of two keys, namely, “subject_name” and “topics”. “subject_name” represents the name of the subject as shown in the demo dataset as “Geography”, “History”, etc whereas “topics” represents the topics of which the content is present in that particular subject.

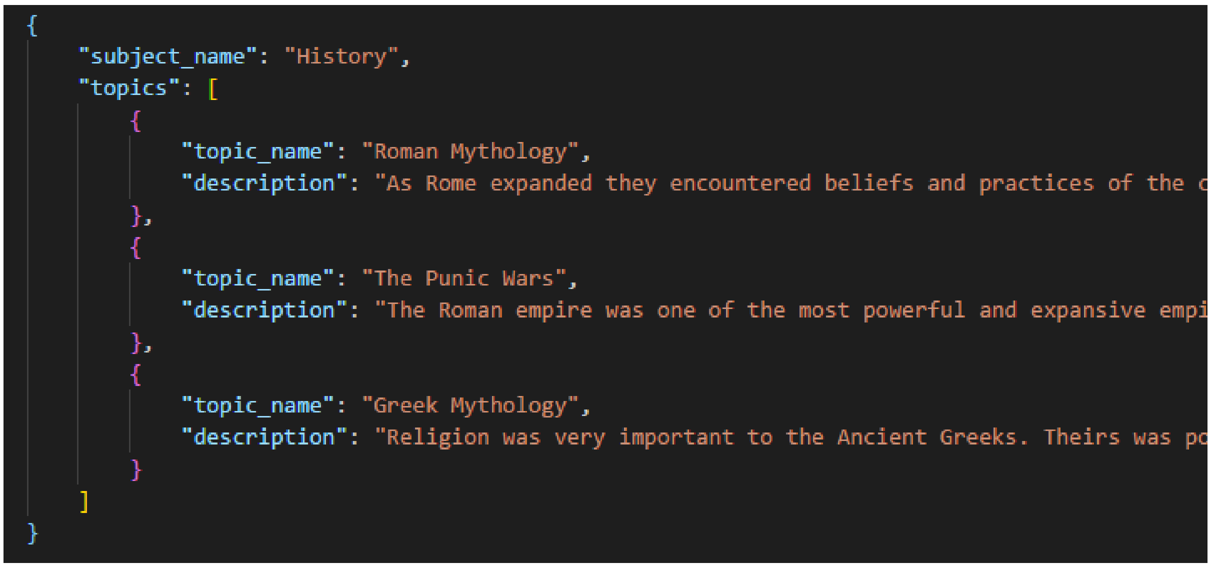


```
{} dataset.json ×

{
    "data": [
        {
            "subject_name": "Geography",
            "topics": [...
            ]
        },
        {
            "subject_name": "History",
            "topics": [...
            ]
        }
    ]
}
```

Figure 3.4 Layer 1 of Data structure used

The Figure 3.5 shows the nested layer or Layer 2 of the demo dataset. It comprises 2 keys i.e., “topic_name” and “description”. Here, “topic_name” represents the topic in that particular subject, and “description” represents the content related to that particular topic of that subject.

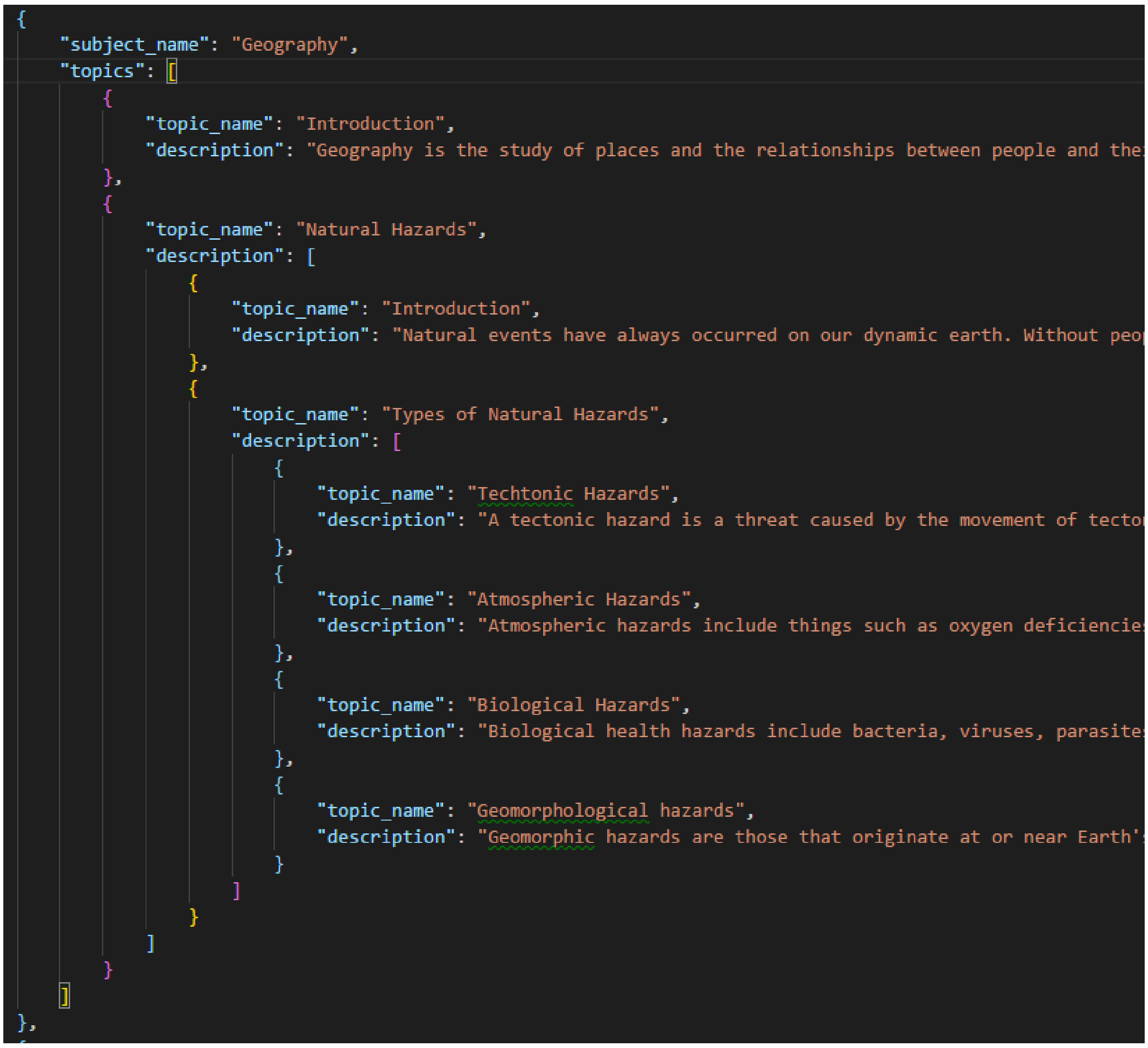
In the Figure 3.5, the description is only string data, that can be directly used as context. But in many cases, there can be nested topics and they can be used as a list format in the description itself. The script running can take care of the list of dictionaries as well.



```
{
    "subject_name": "History",
    "topics": [
        {
            "topic_name": "Roman Mythology",
            "description": "As Rome expanded they encountered beliefs and practices of the c
        },
        {
            "topic_name": "The Punic Wars",
            "description": "The Roman empire was one of the most powerful and expansive empi
        },
        {
            "topic_name": "Greek Mythology",
            "description": "Religion was very important to the Ancient Greeks. Theirs was po
        }
    ]
}
```

Figure 3.5 Nested Layer of Data structure used

The Figure 3.6 shows a complex version of the script which is being used in the demo dataset as well.



```json
{
    "subject_name": "Geography",
    "topics": [
        {
            "topic_name": "Introduction",
            "description": "Geography is the study of places and the relationships between people and the
        },
        {
            "topic_name": "Natural Hazards",
            "description": [
                {
                    "topic_name": "Introduction",
                    "description": "Natural events have always occurred on our dynamic earth. Without peo
                },
                {
                    "topic_name": "Types of Natural Hazards",
                    "description": [
                        {
                            "topic_name": "Techtonic Hazards",
                            "description": "A tectonic hazard is a threat caused by the movement of tecto
                        },
                        {
                            "topic_name": "Atmospheric Hazards",
                            "description": "Atmospheric hazards include things such as oxygen deficiencie
                        },
                        {
                            "topic_name": "Biological Hazards",
                            "description": "Biological health hazards include bacteria, viruses, parasite
                        },
                        {
                            "topic_name": "Geomorphological hazards",
                            "description": "Geomorphic hazards are those that originate at or near Earth'
                        }
                    ]
                }
            ]
        }
    ]
},
```

Figure 3.6 Another Nested Layer of Data structure used

3.4 Model Used

This section comprises of the explanation in two sections of the model used i.e., XLNet model.

3.4.1 XLNet Model

XLNet is a language model that learns unsupervised representations of text sequences using a generalised autoregressive language model. While avoiding the restrictions of AE, this model blends modelling approaches from Autoencoder (AE) models (BERT) into AR models. XLnet is a model extension for the Transformer-XL. It learns bidirectional contexts using an autoregressive method (Kandru, 2021). In 20 separate tasks, XLNet outperforms the state-of-the-art BERT algorithm. In tasks like question answering, natural language inference, sentiment analysis, and document rating, they generally outperform by a wide margin (Srinivasan, 2019).

The state-of-the-art autoregressive model which is Transformer-XL is used in XLNET's pretraining. The Transformer model from Google is used for language translation. It essentially focuses on the concept of "attention." It's an encoder-decoder model in which one sequence is mapped to another (Srinivasan, 2019).

We remove tokens from the input data at random and try to guess what will happen next. This job assists the model in comprehending data patterns, and the model uses the learned parameters to perform effectively in downstream jobs. The Masked Language Modelling aim of BERT may be expressed quantitatively as (Srinivasan, 2019):

- During training and testing, the perm mask input may be utilised to change the precise attention pattern (Srinivasan, 2019).
- Because training a totally auto-regressive model across multiple factorization orders is difficult, XLNet is pre-trained using only a portion of the output tokens as target, which is determined using the target mapping option (Srinivasan, 2019).
- Utilize the perm mask and target mapping inputs to manage the attention span and outputs to use XLNet for sequential decoding (rather than completely bidirectional) (Srinivasan, 2019).
- XLNet is one of the few models that does not limit the length of a sequence (Srinivasan, 2019).

XLNet model can be used in Pytorch as well as in Tensorflow as well. It provides methods and functionalities for both the features equally.

3.4.2 Permutation Language Modelling

PLM is the notion of using an autoregressive model to capture bidirectional context by training it on every conceivable permutation of words in a phrase. XLNET optimises expected log likelihood over all potential permutations of the sequence, rather than using fixed left-right or right-left models. Each position should learn to use contextual information from all other positions, capturing bidirectional context (Srinivasan, 2019).

While doing permutation language modelling on the current segment, XLNet employs Transformer-relative XL's positional embedding and recurrence method to maintain hidden state information from previous segments (Srinivasan, 2019).

XLNet is a game-changing advancement in natural language processing and language modelling. Language modelling is the process of creating models that can extract long-range contextual information and important statistical qualities from a corpus or body of text sequences. Language models that have been trained may predict the next token in a sequence, and the weights can be used to input into other NLP models. The more semantic information and features that a model can extract from a corpus, the better it will perform on subsequent challenges (Srinivasan, 2019).

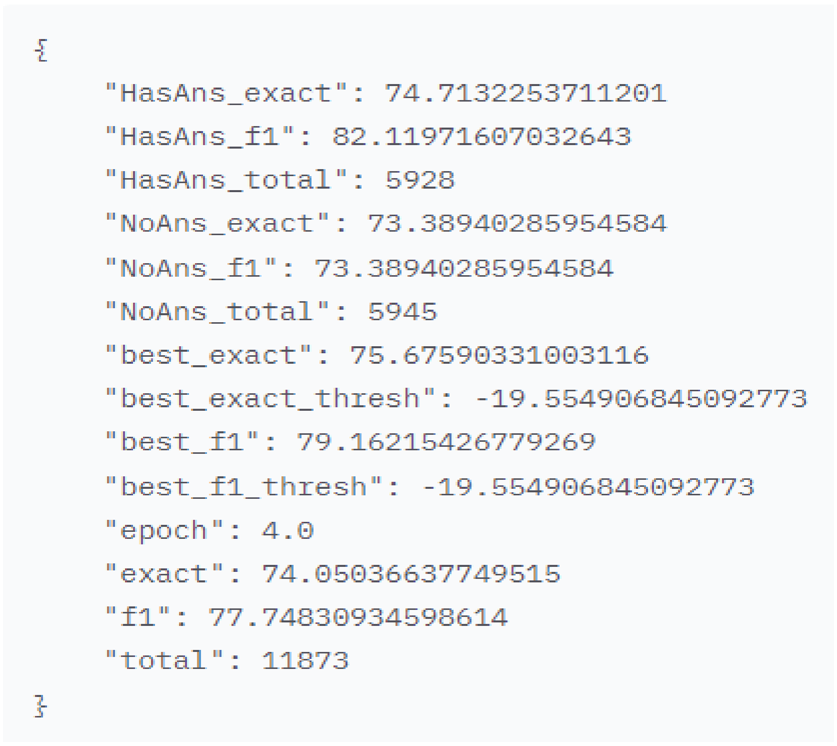
CHAPTER 4 RESULT:

The result section is divided into two parts i.e., Fine-Tuning/training the model and main script using which the trained model is used.

4.1. Fine-Tuning:

As mentioned before, the huggingface is a platform where various NLP pretrained algorithm models can be found. The algorithm used in this research is the XLnet algorithm. XLnet is a pre-trained version of the Transformer-XL model that maximizes the expected likelihood across all permutations of the input sequence factorization in order to learn bidirectional contexts using an autoregressive technique.

For our use, the algorithm was used XLnet but the pretrained model used was “jkgrad/xlnet-base-squadv2”. XLnet was jointly developed by Google and CMU and fine-tuned on SQuAD2.0 for question answering downstream tasks. The training result metrics of the pretrained model were as follows:



```
{
    "HasAns_exact": 74.7132253711201
    "HasAns_f1": 82.11971607032643
    "HasAns_total": 5928
    "NoAns_exact": 73.38940285954584
    "NoAns_f1": 73.38940285954584
    "NoAns_total": 5945
    "best_exact": 75.67590331003116
    "best_exact_thresh": -19.554906845092773
    "best_f1": 79.16215426779269
    "best_f1_thresh": -19.554906845092773
    "epoch": 4.0
    "exact": 74.05036637749515
    "f1": 77.74830934598614
    "total": 11873
}
```

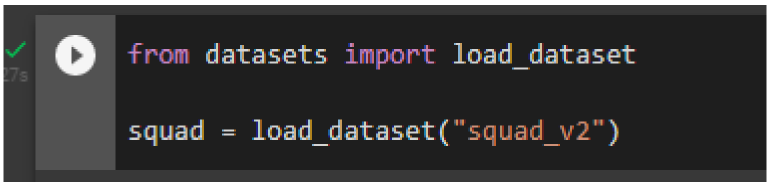
Figure 4.1 Training Result Metrics of jkgrad/xlnet-base-squadv2

As shown in Figure 4.1, the base model of XLNet was fine-tuned with an epoch value of "4". The epoch value is the number which represents the number of times we want to train the model with same dataset. In order to get a significant result and value, the model needs to be trained with a greater number of epochs. Hence, we fine-tuned the model with an epoch value of "16".

The steps used in order to fine-tune the model are:

1. Load the dataset:

   The dataset "squad_v2" is loaded using the package "load_dataset" from datasets library. As shown in Figure 4.2, just the name of the dataset is required and the whole dataset will be downloaded and stored in the variable "squad".

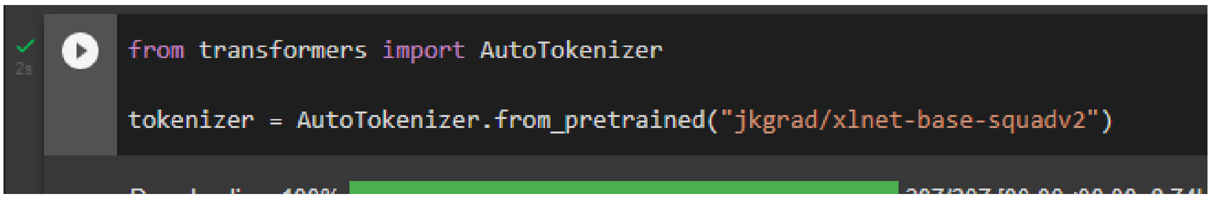


```
from datasets import load_dataset

squad = load_dataset("squad_v2")
```

Figure 4.2 Loading Dataset

2. Load the tokenizer:

   The data from the dataset need to be encoded before feeding it to the model. In order to encode or decode the data from the dataset, Figure 4.3 shows the tokenizers used from the huggingface library. The name of the huggingface library is "transformers". We imported the "Autotokenizer" from transformers and downloaded the tokenizer suitable for the model "jkgrad/xlnet-base-ssquadv2". This tokenizer is stored in the variable "tokenizer".

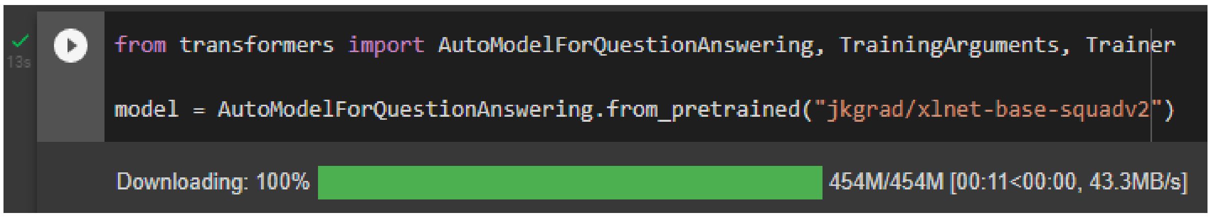


```
from transformers import AutoTokenizer

tokenizer = AutoTokenizer.from_pretrained("jkgrad/xlnet-base-squadv2")
```

Figure 4.3 Loading the tokenizer of selected model

3. Load the model:

The pre-trained model has been downloaded using the "AutoModelForQuestionAnswering". Figure 4.4 shows the model downloaded and stored in the variable "model"

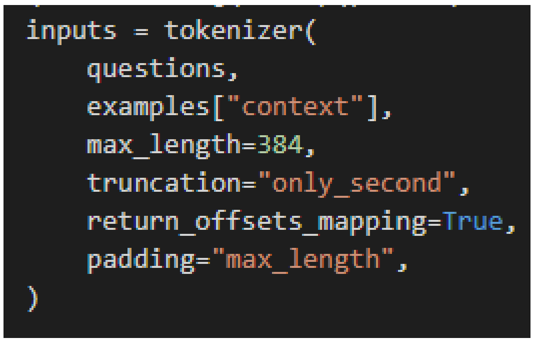


```
from transformers import AutoModelForQuestionAnswering, TrainingArguments, Trainer

model = AutoModelForQuestionAnswering.from_pretrained("jkgrad/xlnet-base-squadv2")

Downloading: 100%  454M/454M [00:11<00:00, 43.3MB/s]
```

Figure 4.4 Loading the model

4. Tokenize (encode) the dataset:

The Figure 4.5 shows the tokenizer downloaded earlier and creating the inputs which will be fed to the model later. The question and context are provided with other arguments like truncation and padding. The outputs are stored in variable "inputs".

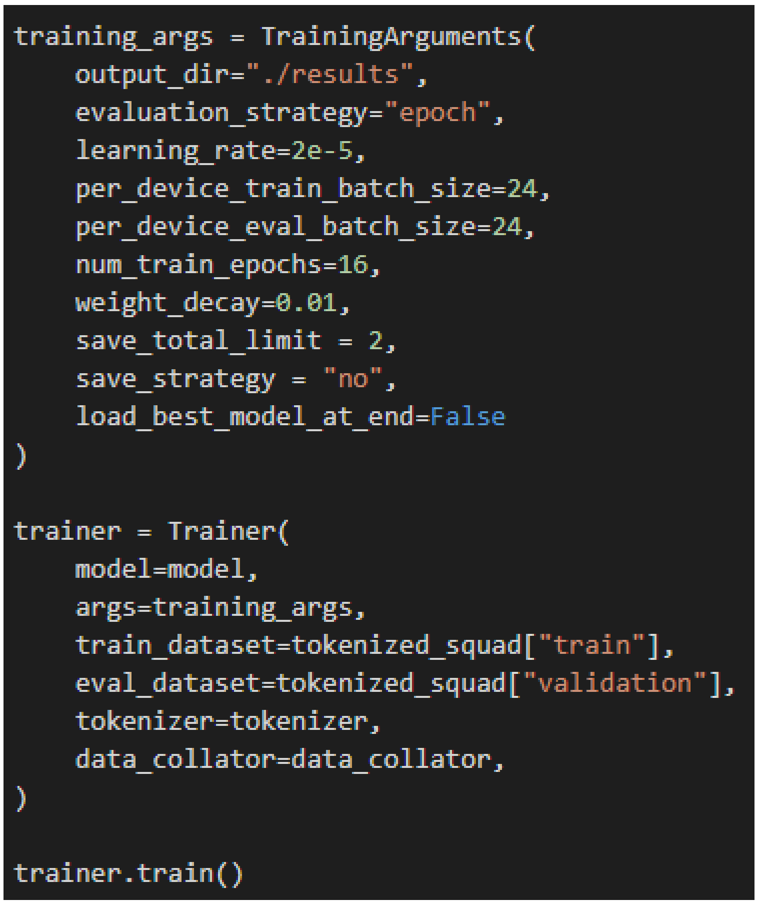


```
inputs = tokenizer(
    questions,
    examples["context"],
    max_length=384,
    truncation="only_second",
    return_offsets_mapping=True,
    padding="max_length",
)
```

Figure 4.5 Tokenizing the data with other parameters

5. Fine-tune the model using encoded value:

Figure 4.6 shows the training commands used. The “TrainingArguements” and “Trainer” are the methods available from huggingface library named “transformers”. There are many arguments that can be used to train the model with multiple epochs and validating the data with different metrics at the same time.

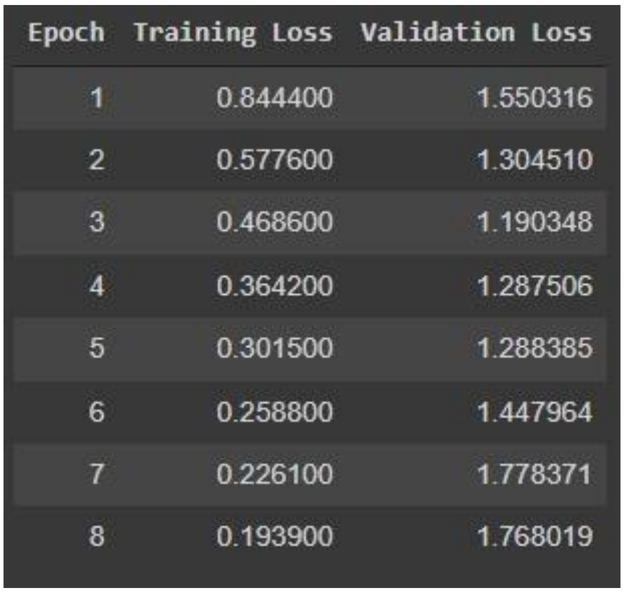


```
training_args = TrainingArguments(
    output_dir="./results",
    evaluation_strategy="epoch",
    learning_rate=2e-5,
    per_device_train_batch_size=24,
    per_device_eval_batch_size=24,
    num_train_epochs=16,
    weight_decay=0.01,
    save_total_limit = 2,
    save_strategy = "no",
    load_best_model_at_end=False
)

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_squad["train"],
    eval_dataset=tokenized_squad["validation"],
    tokenizer=tokenizer,
    data_collator=data_collator,
)

trainer.train()
```

Figure 4.6: Fine tuning the model with specific params

The overall code will be attached in the Appendix A used for fine-tuning. Here, the script demands high-end RAM and GPU availability in order to run the fine tune code. Due to the unavailability of the high-end devices, the XLNet mode is trained with 8 epochs twice. Which will complete the target epoch, instead of running it for 16 epochs at once.

Figure 4.7 shows the last 8-epoch model results with training loss and validation loss. As shown in Figure 4.7, the training loss is 0.193900 which is very good. It might reduce mode if trained with varieties of dataset and more epochs.



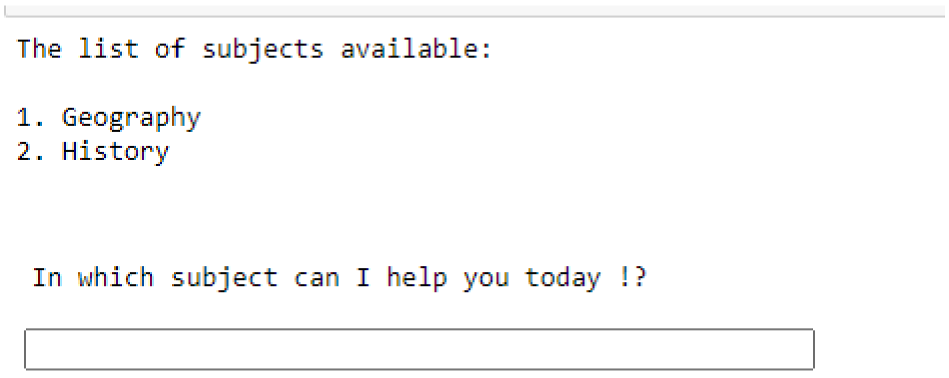
Epoch	Training Loss	Validation Loss
1	0.844400	1.550316
2	0.577600	1.304510
3	0.468600	1.190348
4	0.364200	1.287506
5	0.301500	1.288385
6	0.258800	1.447964
7	0.226100	1.778371
8	0.193900	1.768019

Figure 4.7: XLNet Training Loss and Validation Loss

4.2. Main Script:

For executing the trained model, we need the ask the student about the subject in which he has the question/doubt. For displaying all the subjects that are available in the dataset, we have to access the data and list of subjects first.

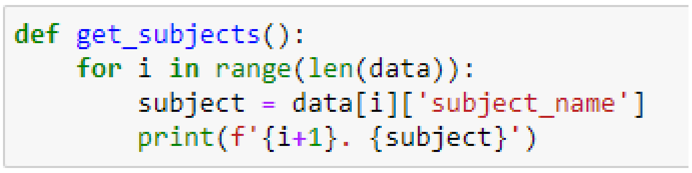
The bot will ask the question with the list of subjects as shown in Figure 4.8.



```
The list of subjects available:

1. Geography
2. History


 In which subject can I help you today !?
```

Figure 4.8: List of subjects available

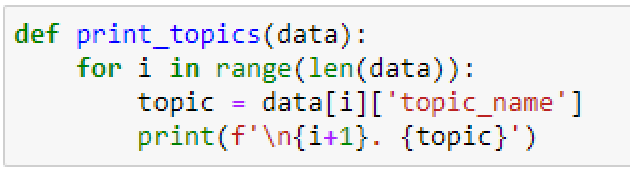
The student has to reply in an integer number. The list of subjects is obtained by a method called “get_subjects” shown in Figure 4.9.



```python
def get_subjects():
    for i in range(len(data)):
        subject = data[i]['subject_name']
        print(f'{i+1}. {subject}')
```

Figure 4.9 get_subjects method

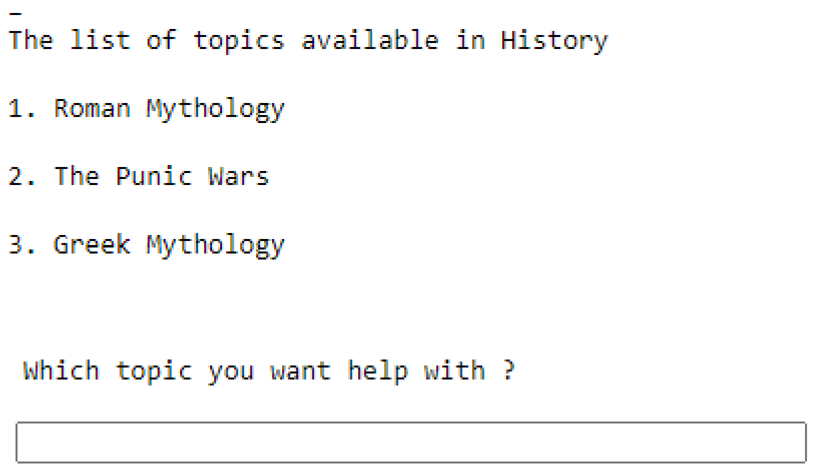
After selecting a subject, the script will then show a list of topics available in that subject. This will happen with the help of "print_topics" method as shown in Figure 4.10. The method will be provided by a list of dictionaries with the structure of "topic_name" and "description" with it and it will print all the list of topics available.



```python
def print_topics(data):
    for i in range(len(data)):
        topic = data[i]['topic_name']
        print(f'\n{i+1}. {topic}')
```

Figure 4.10 print_topics method

The Figure 4.11 shows the list of topics available in the subject "History". The bot will then again ask the student the specific topic on which he/she wants help.

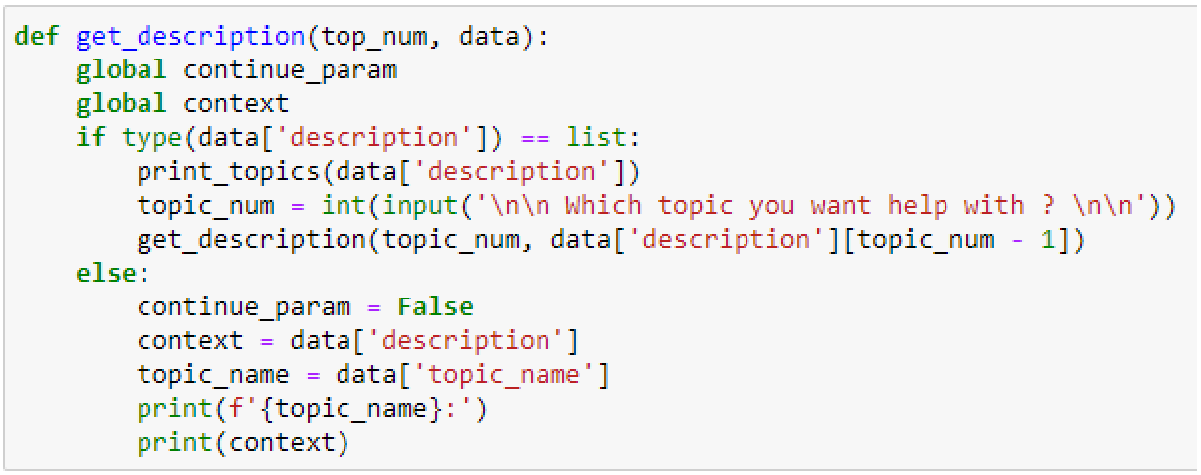


```
-
The list of topics available in History

1. Roman Mythology

2. The Punic Wars

3. Greek Mythology


 Which topic you want help with ?
```

Figure 4.11: List of Topics available

After selecting the specific topic, the script will check whether the dataset has any more subtopics on the selected topic. If yes, then the subtopics will be printed, else, the description of the selected topic will be printed. This is done with the help of the method "get_description".

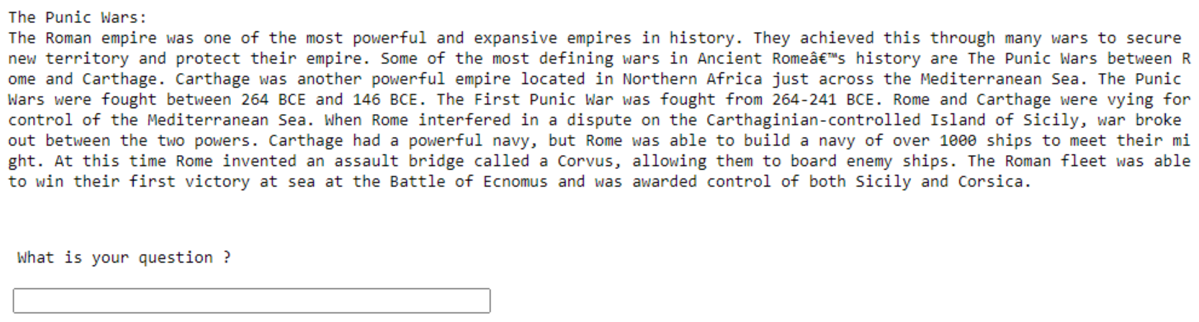
The Figure 4.12 shows the "get_description" method used to get the subtopics and all the nested topics within the selected topic. The "get_description" method takes 2 arguments viz, "top_num" which is topic_number, and "data" in which description or the subtopics of the selected topic will be available.



```python
def get_description(top_num, data):
    global continue_param
    global context
    if type(data['description']) == list:
        print_topics(data['description'])
        topic_num = int(input('\n\n Which topic you want help with ? \n\n'))
        get_description(topic_num, data['description'][topic_num - 1])
    else:
        continue_param = False
        context = data['description']
        topic_name = data['topic_name']
        print(f'{topic_name}:')
        print(context)
```

Figure 4.12 get_description method

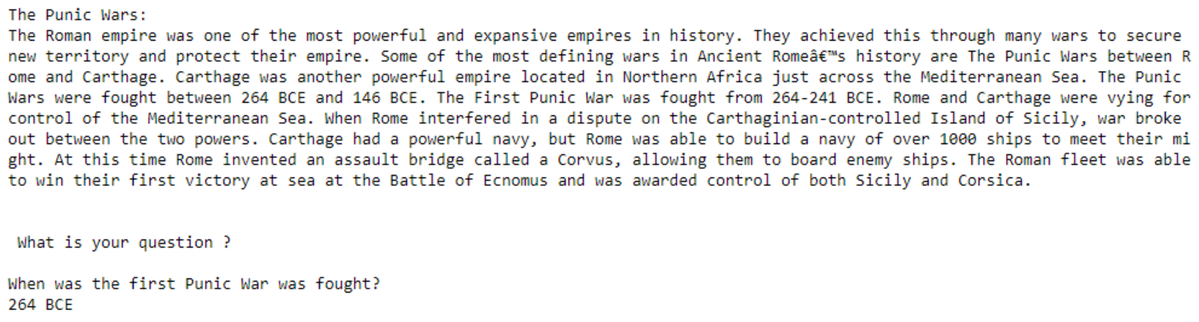
The Figure 4.13 shows the description of the topic "Punic Wars" and the script is asking for a question to be asked on the displayed description.



```
The Punic Wars:
The Roman empire was one of the most powerful and expansive empires in history. They achieved this through many wars to secure
new territory and protect their empire. Some of the most defining wars in Ancient Romeâ€™s history are The Punic Wars between R
ome and Carthage. Carthage was another powerful empire located in Northern Africa just across the Mediterranean Sea. The Punic
Wars were fought between 264 BCE and 146 BCE. The First Punic War was fought from 264-241 BCE. Rome and Carthage were vying for
control of the Mediterranean Sea. When Rome interfered in a dispute on the Carthaginian-controlled Island of Sicily, war broke
out between the two powers. Carthage had a powerful navy, but Rome was able to build a navy of over 1000 ships to meet their mi
ght. At this time Rome invented an assault bridge called a Corvus, allowing them to board enemy ships. The Roman fleet was able
to win their first victory at sea at the Battle of Ecnomus and was awarded control of both Sicily and Corsica.


 What is your question ?
```

Figure 4.13 Topic name with its context

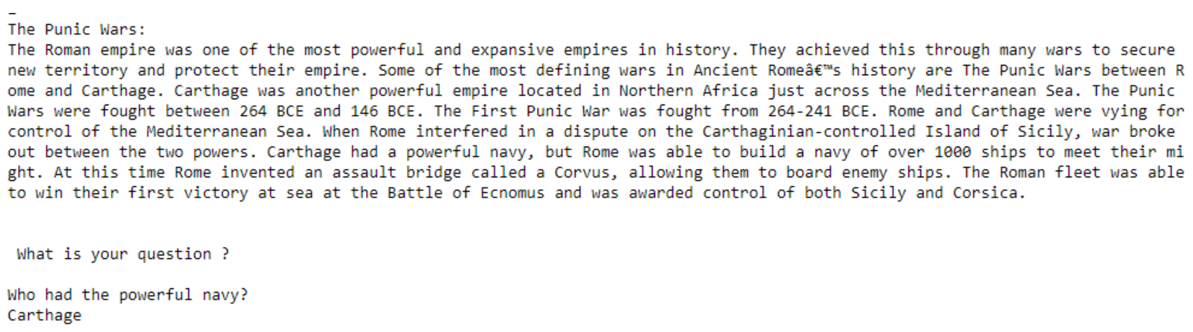
Figure 4.14 shows that upon asking the 1st question "When was the First Punic War was fought?", the algorithm replies with the correct answer as "264 BCE".



```
The Punic Wars:
The Roman empire was one of the most powerful and expansive empires in history. They achieved this through many wars to secure
new territory and protect their empire. Some of the most defining wars in Ancient Romeâ€™s history are The Punic Wars between R
ome and Carthage. Carthage was another powerful empire located in Northern Africa just across the Mediterranean Sea. The Punic
Wars were fought between 264 BCE and 146 BCE. The First Punic War was fought from 264-241 BCE. Rome and Carthage were vying for
control of the Mediterranean Sea. When Rome interfered in a dispute on the Carthaginian-controlled Island of Sicily, war broke
out between the two powers. Carthage had a powerful navy, but Rome was able to build a navy of over 1000 ships to meet their mi
ght. At this time Rome invented an assault bridge called a Corvus, allowing them to board enemy ships. The Roman fleet was able
to win their first victory at sea at the Battle of Ecnomus and was awarded control of both Sicily and Corsica.


 What is your question ?

When was the first Punic War was fought?
264 BCE
```

Figure 4.14 1st Question asked on topic Punic Wars

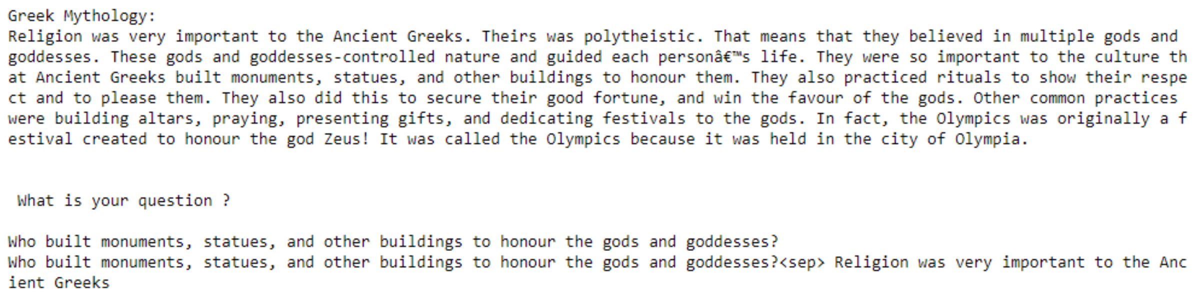
Similarly, as the second question was asked, “who had the powerful navy?”, the model replied with the correct answer i.e., “Carthage” which can be seen in Figure 4.15.



```
-
The Punic Wars:
The Roman empire was one of the most powerful and expansive empires in history. They achieved this through many wars to secure
new territory and protect their empire. Some of the most defining wars in Ancient Romeâ€™s history are The Punic Wars between R
ome and Carthage. Carthage was another powerful empire located in Northern Africa just across the Mediterranean Sea. The Punic
Wars were fought between 264 BCE and 146 BCE. The First Punic War was fought from 264-241 BCE. Rome and Carthage were vying for
control of the Mediterranean Sea. When Rome interfered in a dispute on the Carthaginian-controlled Island of Sicily, war broke
out between the two powers. Carthage had a powerful navy, but Rome was able to build a navy of over 1000 ships to meet their mi
ght. At this time Rome invented an assault bridge called a Corvus, allowing them to board enemy ships. The Roman fleet was able
to win their first victory at sea at the Battle of Ecnomus and was awarded control of both Sicily and Corsica.


 What is your question ?

Who had the powerful navy?
Carthage
```

Figure 4.15 2nd Question asked on topic Punic Wars

As shown in Figure 4.16, after trying a few times, the model replied with a few wrong answers as well. The model is very excellent and had a great success rate and if trained more and properly, its accuracy can be increased drastically. The overall code will be attached in the Appendix B used for the main script.



```
Greek Mythology:
Religion was very important to the Ancient Greeks. Theirs was polytheistic. That means that they believed in multiple gods and
goddesses. These gods and goddesses-controlled nature and guided each personâ€™s life. They were so important to the culture th
at Ancient Greeks built monuments, statues, and other buildings to honour them. They also practiced rituals to show their respe
ct and to please them. They also did this to secure their good fortune, and win the favour of the gods. Other common practices
were building altars, praying, presenting gifts, and dedicating festivals to the gods. In fact, the Olympics was originally a f
estival created to honour the god Zeus! It was called the Olympics because it was held in the city of Olympia.


 What is your question ?

Who built monuments, statues, and other buildings to honour the gods and goddesses?
Who built monuments, statues, and other buildings to honour the gods and goddesses?<sep> Religion was very important to the Anc
ient Greeks
```

Figure 4.16 3rd Question asked on topic Greek Mythology

Here, we have successfully created the question-answering chatbot using XLNet model in working condition.

CHAPTER 5 CONCLUSION AND FUTURE DIRECTIONS

5.1 Conclusion

The effectiveness of NLP integrated question-answering chatbot in digital learning can be noticed easily. The customized XLnet-based model created here has a higher accuracy rate as compared to any other model. In the beginning, the created chatbot can be used in high schools. Chatbots may be smoothly integrated with educational apps to provide rapid answers to frequent inquiries or even easy resolutions. Apart from training the model, which took around a few days, everything else in this research took less time to complete. If made properly, this chatbot can be created more interactive and with a pleasant user interface.

The role of chatbots in evolving the traditional teaching methods will be very crucial. In this manner, the top frequent topics on which the questions were asked, can be determined easily by the teacher and they can revise the topic again for better understanding. It can also be used to grade students or arrange the results for every student and create the profile of a mark automatically. The data can be viewed by authorized people. Especially if the crowd of students is high, this will help in creating a sorted profile with the number of questions asked repeatedly so that it can be cross-checked and re-taught again. The language barrier can also be overcome by this question-answering chatbot.

There will be certain challenges pertaining to the adoption of chatbots in the education system. Development issues are more common in software programs. The creation of software from scratch is a tedious task. Especially NLP algorithm models. They are trained from the basics. The more they are trained, the more accurate the answers/replies will be. User attitude issues are also found in many chatbot applications. When the person ahead realizes that they are talking to a chatbot, their attitude towards having the conversation changes drastically. Data privacy issues can also occur very frequently. Sometimes, people go to great lengths to not get the data into the wrong hands. Chatbot conversations must be completely private and should not be shared with anyone except the entities included. Training the algorithm while it is in the application can also create some problems at times. Feeding the data to the already working algorithm can be a risky task because if not properly trained, the result can disrupt the flow of accuracy that the algorithm was provided previously. In this manner, we have achieved all of our objectives for this research paper.

5.2 Future Directions

The model created in this research can be improvised more with training and reducing the validation loss. The XLNet based question-answering chatbot gives can be evolved in a mobile app or a web app as well. With a proper and interactive user interface, it can be made more interesting and easier to use.

Different types of datasets can be used to train the model further. It will give the model cutting edge accuracy. It can also be trained in order to answer tabular questions. After training and using the tabular questions, the model can also be used by teachers and in order to determine various ranks of students. It can also be of help in creating the student profile automatically on the usage of chatbot. The student can also be given a multiple-choice questions test and upon answering the question, it can also justify by giving the context of the question asked.

The application of the XLNet based question-answering chatbot in the field of education is enumerous. With proper development, it can be made easy for teachers to include custom subjects and accessible to only authorized students. Including everything, the XLNet based question-answering chatbot will provide one-on-one teaching to the student and reduce the gap between the student and a teacher where the number of students is way larger and the teacher cannot focus on each and every one of them.

References

A. R. Mohammad Forkan, P. P. J. Y. K. a. A. M., 2020. ECHO: A tool for empirical evaluation cloud chatbots. In 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing. *IEEE,* pp. 669-672.

Ali, B. R. V. B. C. S. M. S. S. O., 2021. Chatbot via Machine Learning and Deep Learning Hybrid. In: *Modern Approaches in Machine Learning and Cognitive Science: A Walkthrough. Studies in Computational Intelligence.* s.l.:s.n.

Beklemysheva, A., n.d. *Why Use Python for AI and Machine Learning?.* [Online] Available at: https://steelkiwi.com/blog/python-for-ai-and-machine-learning/ [Accessed 30 05 2022].

Botsify, n.d. *Why is Education Industry opting for AI Chatbots? How Are They Benefiting It? - Botsify.* [Online] Available at: https://botsify.com/blog/education-industry-chatbot/ [Accessed 30 05 2022].

Chat Compose, n.d. *How to use Chatbots for Education and Learning.* [Online] Available at: https://www.chatcompose.com/chatbot-learning.html [Accessed 30 05 2022].

Ching-Yi Chang, G.-J. H. M.-L. G., 2020. Promoting Students' Learning achievement and self-efficacy: A mobile chatbot approach for nursing training. *British Journal of Educational Technology.*

E. Kasthuri, S. B., 2021. A Chatbot for Changing Lifestyle in Education. *IEEE.*

E.Kasthuri, S., 2021. Natural language processing and deep learning chatbot using long short term memory algorithm. *Science Direct.*

EleniAdamopoulou, L., 2020. Chatbots: History, technology, and applications. *Science Direct.*

Engati Team, n.d. *6 types of chatbots that&#x27;ll help your business grow! | Engati.* [Online] Available at: https://www.engati.com/blog/types-of-chatbots-and-their-applications#toc-what-are-some-applications-of-chatbots-
[Accessed 30 05 2022].

Fryer, L. K. & Nakao, K., 2019. Chatbot learning partners: Connecting learning experiences, interest and competence. Computers in Human Behavior. *Research Gate,* pp. 279-289.

Inamdar, V. A. & S. R., 2019. Development of college enquiry chatbot using snatchbot. *International Research Journal of Engineering and Technology (IRJET).*

Irina Dokukina, J. G., 2020. The rise of chatbots – new personal assistants in foreign language learning. *Science Direct.*

J. E. Zini, Y. R. M. A. a. J. A., 2019. Towards A Deep Learning Question-Answering Specialized Chatbot for Objective Structured Clinical Examinations. *IEEE*, pp. 1-9.

José Quiroga Pérez, T. D. J. M. M. P., 2020. Rediscovering the use of chatbots in education: A systematic literature review. *Computer Applications in Engineering Education,* 28(6), pp. 1549-1565.

Joshi, P., 2019. *Transformers In NLP | State-Of-The-Art-Models.* [Online]
Available at: https://www.analyticsvidhya.com/blog/2019/06/understanding-transformers-nlp-state-of-the-art-models/
[Accessed 30 05 2022].

Kandru, P., 2021. *Guide to XLNet for Language Understanding.* [Online]
Available at: https://analyticsindiamag.com/guide-to-xlnet-for-language-understanding/
[Accessed 30 05 2022].

Kee-Man Chuah & Muhammad Kamarul Kabilan, 2021. Teachers' Views on The Use of Chatbots to Support English Language Teaching in a Mobile Environment. *International Journal of Emerging Technologies in Learning (iJET),* 16(20).

Kelleher, J. D., 2019. *Deep Learning (The MIT Press Essential Knowledge).* s.l.:The MIT Press.

Kumar, S. P. R. K. a. B. S., 2020. Deep Learning Techniques for Implementation of Chatbots. *IEEE.*

Liang, Y., Yu, Y. & Ouyang, W., 2020. *Intelligent chat robot in digital campus based on deep learning.* s.l., Journal of Physics: Conference Series.

M. Polignano, F. N. A. I. C. M. M. D. G. a. G. S., 2020. HealthAssistantBot: A Personal Health Assistant for the Italian Language. *IEEE,* Volume 8, pp. 107479-107497.

Mathawan, J. et al., 2021. *Automatized Educational Chatbot.* s.l., IEEE.

Motahhir, S. & Bossoufi, B, 2021. Enhancing Education System with a Q&A Chatbot: A Case Based on Open edX Platform. In: *Digital Technologies and Applications.* s.l.:Springer, Cham.

Numpy, n.d. *NumPy.* [Online]
Available at: https://numpy.org/
[Accessed 30 05 2022].

Oracle, n.d. *What is a Chatbot | Oracle India.* [Online]
Available at: https://www.oracle.com/in/chatbots/what-is-a-chatbot/
[Accessed 30 05 2022].

pandas, n.d. *pandas - Python Data Analysis Library.* [Online]
Available at: https://pandas.pydata.org/
[Accessed 30 05 2022].

Pavel Smutny, P. S., 2020. Chatbots for learning: A review of educational chatbots for the Facebook Messenger. *Science Direct,* Volume 151.

S. Kumari, Z. N. A. A. a. P. D., 2020. Enhancing College Chat Bot Assistant with the Help of Richer Human Computer Interaction and Speech Recognition. *IEEE,* pp. 427-433.

Sreelakshmi, A. &. S. B. A. &. N. A. &. N. J., 2019. *A Question Answering and Quiz Generation Chatbot for Education.* s.l., Grace Hopper Celebration India (GHCI), pp. 1-6.

Srinivasan, A. V., 2019. *XLNET explained in simple terms !! | by Aishwarya V Srinivasan | Towards Data Science.* [Online]
Available at: https://towardsdatascience.com/xlnet-explained-in-simple-terms-255b9fb2c97c
[Accessed 30 05 2022].

Tarek AITBAHA, M. E. Y.-S. H., 2022. Towards highly adaptive Edu-Chatbot. *Science Direct,* Volume 198, pp. 397-403.

TensorFlow, n.d. *squad | TensorFlow Datasets.* [Online]
Available at: https://www.tensorflow.org/datasets/catalog/squad
[Accessed 30 05 2022].

uPlanner, 2017. *8 Causas de deserción estudiantil en la educación superior - uPlanner.* [Online]
Available at: https://uplanner.com/es/8-causas-de-desercion-estudiantil-en-la-educacion-superior-2/
[Accessed 30 05 2022].

Wikipedia, n.d. *Machine learning - Wikipedia.* [Online]
Available at: https://en.wikipedia.org/wiki/Machine_learning
[Accessed 30 05 2022].

Wikipedia, n.d. *PyTorch - Wikipedia.* [Online]
Available at: https://en.wikipedia.org/wiki/PyTorch
[Accessed 30 05 2022].

Wikipedia, n.d. *Regression analysis - Wikipedia.* [Online]
Available at: https://en.wikipedia.org/wiki/Regression_analysis
[Accessed 30 05 2022].

Wikipedia, n.d. *Semi-supervised learning - Wikipedia.* [Online]
Available at: https://en.wikipedia.org/wiki/Semi-supervised_learning
[Accessed 30 05 2022].

Wikipedia, n.d. *Statistical classification - Wikipedia.* [Online]
Available at: https://en.wikipedia.org/wiki/Statistical_classification
[Accessed 30 05 2022].

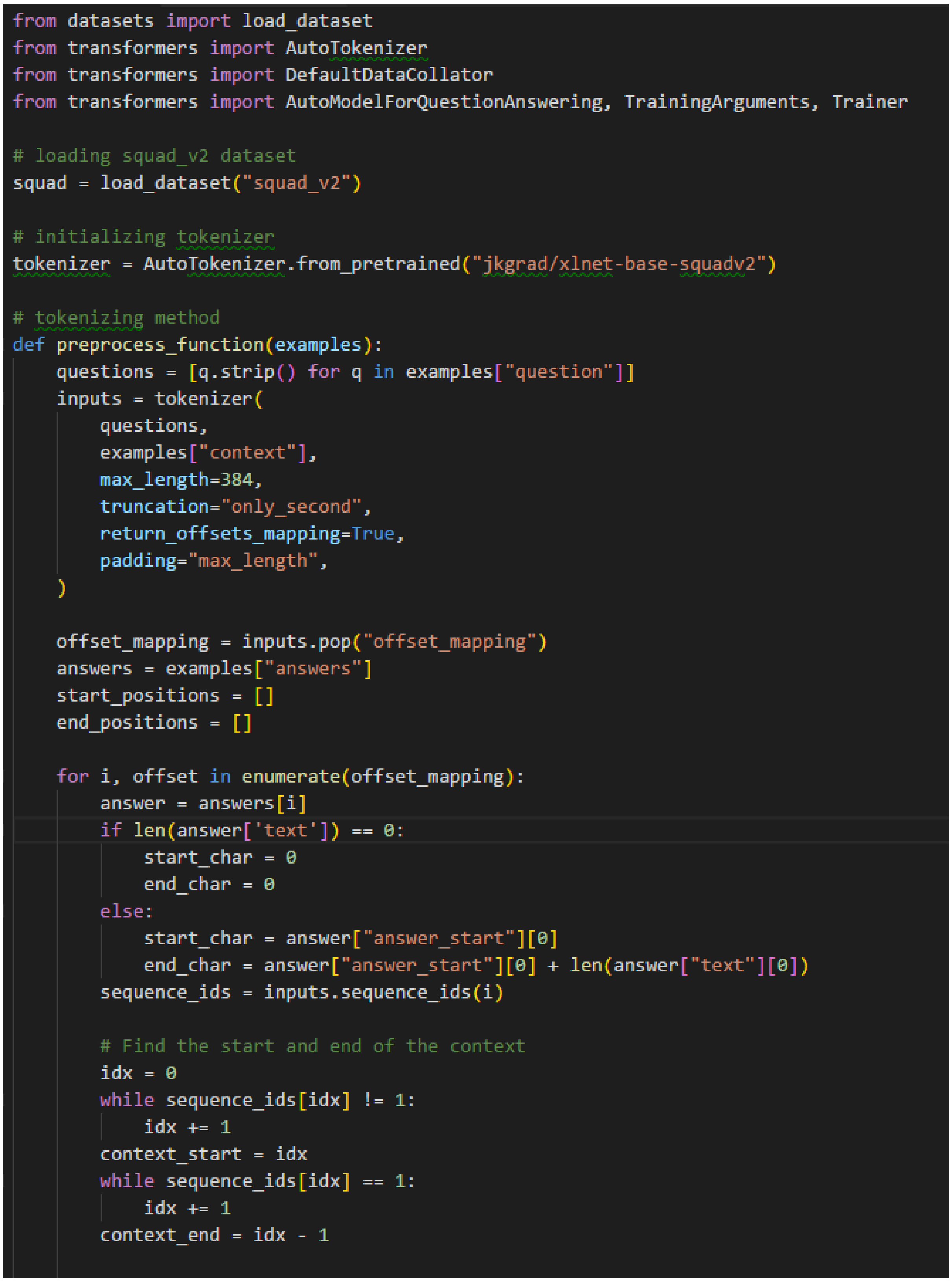
Wikipedia, n.d. *Supervised Learning - Wikipedia.* [Online]
Available at: https://en.wikipedia.org/wiki/Supervised_learning
[Accessed 30 05 2022].

Wikipedia, n.d. *Transformer (machine learning model) - Wikipedia.* [Online]
Available at: https://en.wikipedia.org/wiki/Transformer_(machine_learning_model)
[Accessed 30 05 2022].

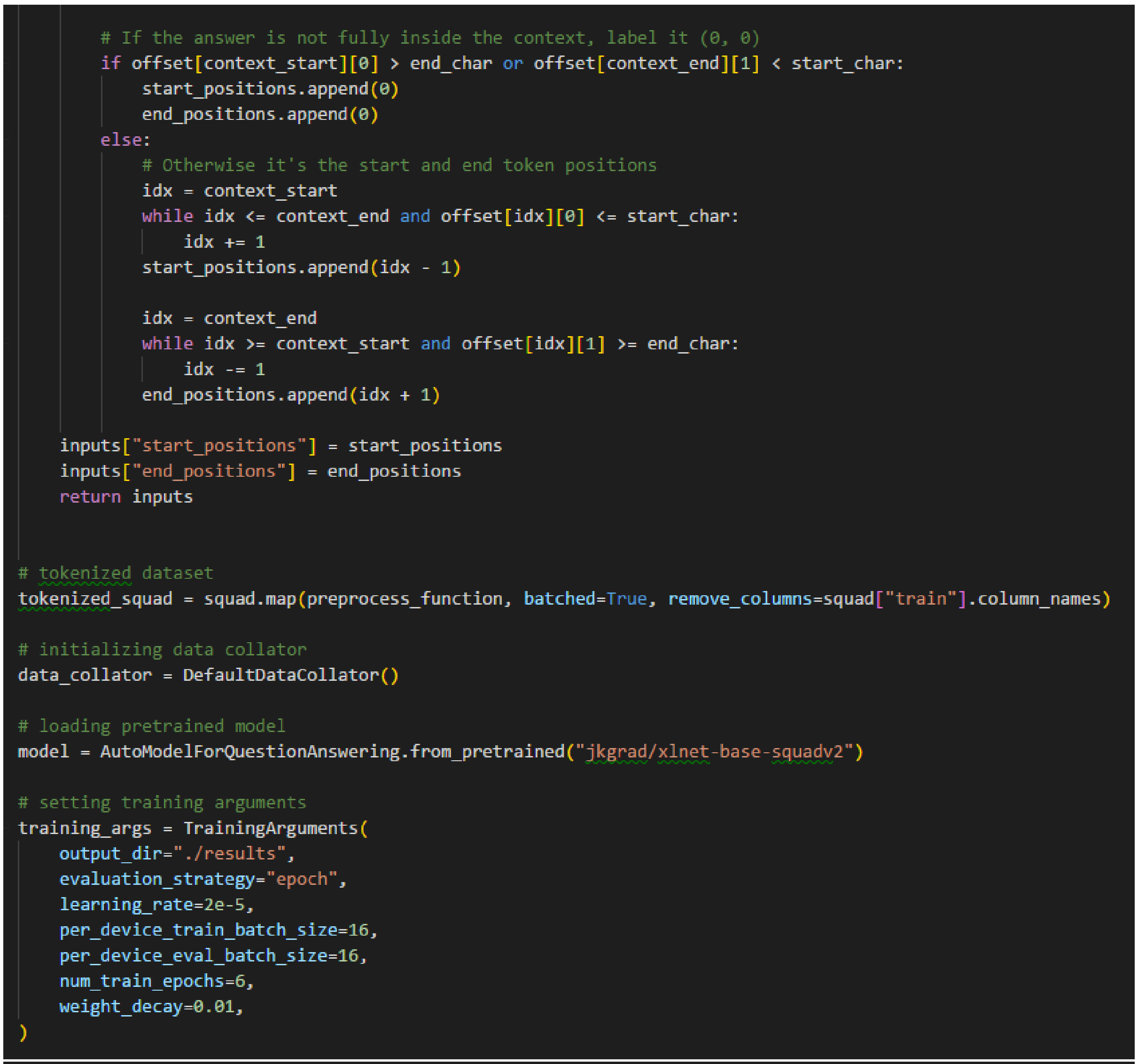
Wikipedia, n.d. *Unsupervised learning - Wikipedia.* [Online] Available at: https://en.wikipedia.org/wiki/Unsupervised_learning [Accessed 30 05 2022].

Yadav, D., 2019. *Categorical encoding using Label-Encoding and One-Hot-Encoder | by Dinesh Yadav | Towards Data Science.* [Online] Available at: https://towardsdatascience.com/categorical-encoding-using-label-encoding-and-one-hot-encoder-911ef77fb5bd
[Accessed 30 05 2022].

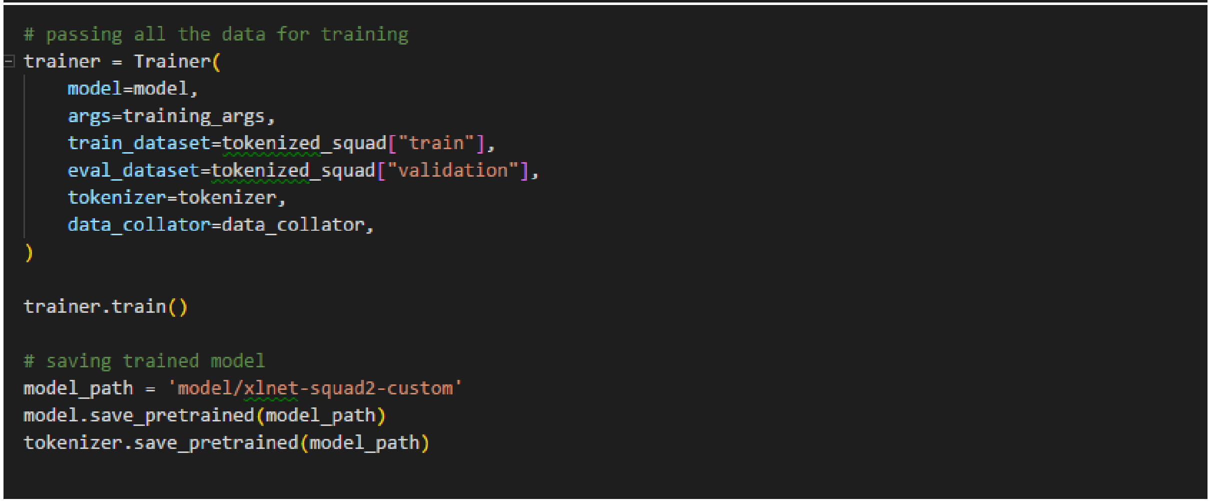
Appendix A: Fine tune code (fine_tune.py)



```python
from datasets import load_dataset
from transformers import AutoTokenizer
from transformers import DefaultDataCollator
from transformers import AutoModelForQuestionAnswering, TrainingArguments, Trainer

# loading squad_v2 dataset
squad = load_dataset("squad_v2")

# initializing tokenizer
tokenizer = AutoTokenizer.from_pretrained("jkgrad/xlnet-base-squadv2")

# tokenizing method
def preprocess_function(examples):
    questions = [q.strip() for q in examples["question"]]
    inputs = tokenizer(
        questions,
        examples["context"],
        max_length=384,
        truncation="only_second",
        return_offsets_mapping=True,
        padding="max_length",
    )

    offset_mapping = inputs.pop("offset_mapping")
    answers = examples["answers"]
    start_positions = []
    end_positions = []

    for i, offset in enumerate(offset_mapping):
        answer = answers[i]
        if len(answer['text']) == 0:
            start_char = 0
            end_char = 0
        else:
            start_char = answer["answer_start"][0]
            end_char = answer["answer_start"][0] + len(answer["text"][0])
        sequence_ids = inputs.sequence_ids(i)

        # Find the start and end of the context
        idx = 0
        while sequence_ids[idx] != 1:
            idx += 1
        context_start = idx
        while sequence_ids[idx] == 1:
            idx += 1
        context_end = idx - 1
```

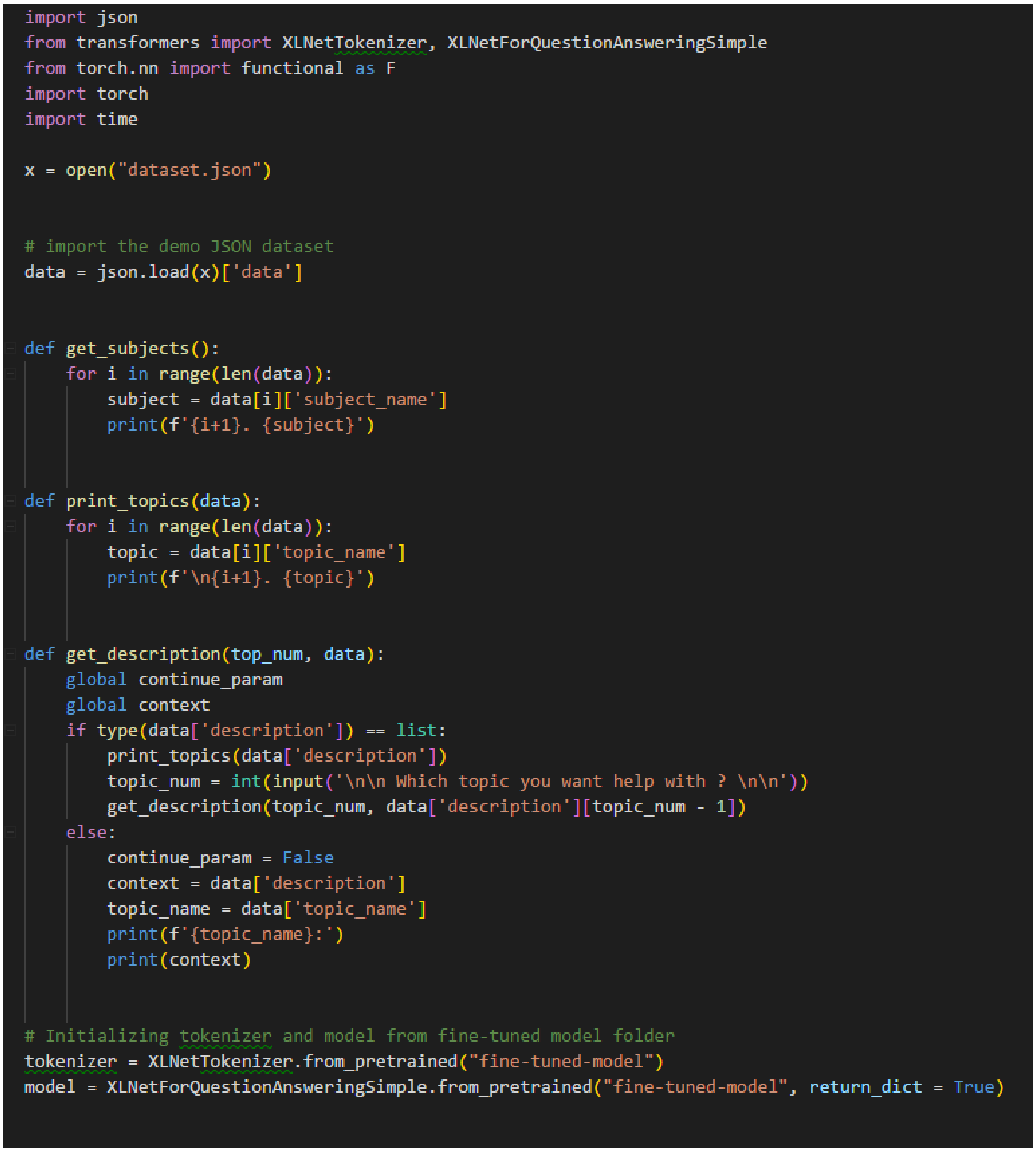


```
        # If the answer is not fully inside the context, label it (0, 0)
        if offset[context_start][0] > end_char or offset[context_end][1] < start_char:
            start_positions.append(0)
            end_positions.append(0)
        else:
            # Otherwise it's the start and end token positions
            idx = context_start
            while idx <= context_end and offset[idx][0] <= start_char:
                idx += 1
            start_positions.append(idx - 1)

            idx = context_end
            while idx >= context_start and offset[idx][1] >= end_char:
                idx -= 1
            end_positions.append(idx + 1)

    inputs["start_positions"] = start_positions
    inputs["end_positions"] = end_positions
    return inputs


# tokenized dataset
tokenized_squad = squad.map(preprocess_function, batched=True, remove_columns=squad["train"].column_names)

# initializing data collator
data_collator = DefaultDataCollator()

# loading pretrained model
model = AutoModelForQuestionAnswering.from_pretrained("jkgrad/xlnet-base-squadv2")

# setting training arguments
training_args = TrainingArguments(
    output_dir="./results",
    evaluation_strategy="epoch",
    learning_rate=2e-5,
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    num_train_epochs=6,
    weight_decay=0.01,
)
```

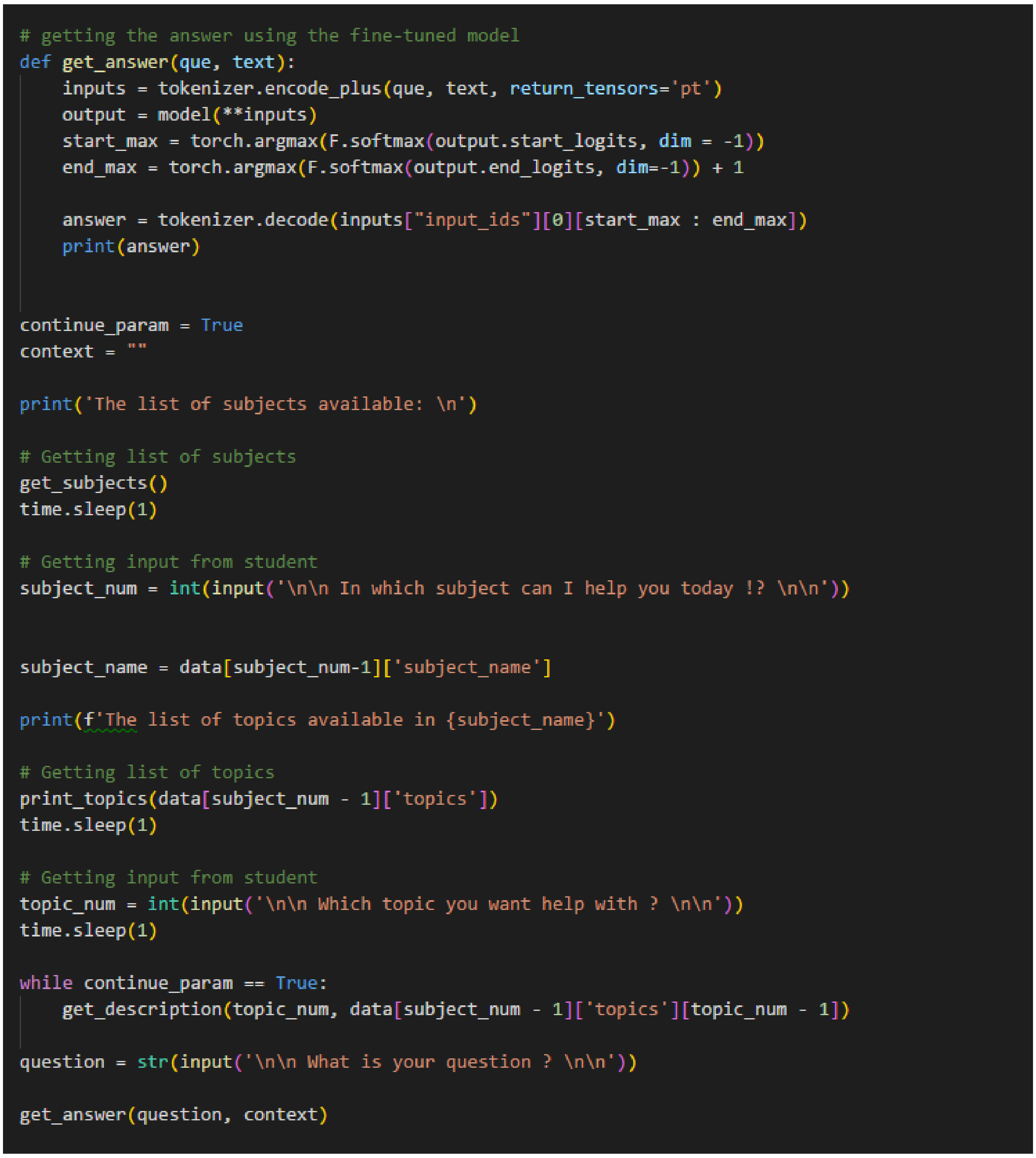


```
# passing all the data for training
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_squad["train"],
    eval_dataset=tokenized_squad["validation"],
    tokenizer=tokenizer,
    data_collator=data_collator,
)

trainer.train()

# saving trained model
model_path = 'model/xlnet-squad2-custom'
model.save_pretrained(model_path)
tokenizer.save_pretrained(model_path)
```

Appendix B: Main Script Code (main_script.py)



```python
import json
from transformers import XLNetTokenizer, XLNetForQuestionAnsweringSimple
from torch.nn import functional as F
import torch
import time

x = open("dataset.json")


# import the demo JSON dataset
data = json.load(x)['data']


def get_subjects():
    for i in range(len(data)):
        subject = data[i]['subject_name']
        print(f'{i+1}. {subject}')


def print_topics(data):
    for i in range(len(data)):
        topic = data[i]['topic_name']
        print(f'\n{i+1}. {topic}')


def get_description(top_num, data):
    global continue_param
    global context
    if type(data['description']) == list:
        print_topics(data['description'])
        topic_num = int(input('\n\n Which topic you want help with ? \n\n'))
        get_description(topic_num, data['description'][topic_num - 1])
    else:
        continue_param = False
        context = data['description']
        topic_name = data['topic_name']
        print(f'{topic_name}:')
        print(context)


# Initializing tokenizer and model from fine-tuned model folder
tokenizer = XLNetTokenizer.from_pretrained("fine-tuned-model")
model = XLNetForQuestionAnsweringSimple.from_pretrained("fine-tuned-model", return_dict = True)
```



```python
# getting the answer using the fine-tuned model
def get_answer(que, text):
    inputs = tokenizer.encode_plus(que, text, return_tensors='pt')
    output = model(**inputs)
    start_max = torch.argmax(F.softmax(output.start_logits, dim = -1))
    end_max = torch.argmax(F.softmax(output.end_logits, dim=-1)) + 1

    answer = tokenizer.decode(inputs["input_ids"][0][start_max : end_max])
    print(answer)


continue_param = True
context = ""

print('The list of subjects available: \n')

# Getting list of subjects
get_subjects()
time.sleep(1)

# Getting input from student
subject_num = int(input('\n\n In which subject can I help you today !? \n\n'))


subject_name = data[subject_num-1]['subject_name']

print(f'The list of topics available in {subject_name}')

# Getting list of topics
print_topics(data[subject_num - 1]['topics'])
time.sleep(1)

# Getting input from student
topic_num = int(input('\n\n Which topic you want help with ? \n\n'))
time.sleep(1)

while continue_param == True:
    get_description(topic_num, data[subject_num - 1]['topics'][topic_num - 1])

question = str(input('\n\n What is your question ? \n\n'))

get_answer(question, context)
```