**Report on Academic Chat-bot**

Author: Jose Manuel Alvarez Gonzalez

Email Id: [jmanuel.ag89@gmail.com](mailto:jmanuel.ag89@gmail.com)

Date: 13/3/2023

Contents

[Introduction 3](#_Toc161158378)

[NLP 3](#_Toc161158379)

[Sentence-BERT 4](#_Toc161158380)

[Vector Database 4](#_Toc161158381)

[Methodology 5](#_Toc161158382)

[Models 5](#_Toc161158383)

[Evaluation and Results 5](#_Toc161158384)

[Risks 6](#_Toc161158385)

[Deployment 6](#_Toc161158386)

[GPT-based for ‘text-generation’ 7](#_Toc161158387)

[LangChain 7](#_Toc161158388)

[Methodology 8](#_Toc161158389)

[Models 9](#_Toc161158390)

[Evaluation and Results 10](#_Toc161158391)

[Risks 11](#_Toc161158392)

[Deployment 11](#_Toc161158393)

[Conclusion and Recommendation 12](#_Toc161158394)

[Reference 12](#_Toc161158395)

# Introduction

The propose of the following work, is to investigate the potential uses of NLP to build an Academic Chat-Bot to support students during non-office hours. In the following work we are going to take two different approaches for building a chat bot, in each of the approaches we will be discussing the new terminology, methodology behind, the data used in process, evaluation, scope, limitations, risks, and applicability.

Approaches used for building the chat-bot:

* Sentence-BERT, in this approach we are going to be using different type of BERT style models. The focus of the model will be on converting sentences on a numeric form call vector and finding similarity across sentences to match with a predetermine answer to the question.
* GPT-based for ‘text-generation’, in this approach we are going to be working with a mixture of different models designed for text-generation. We are going to look at different methods to interact with the models and evaluate the answers they provide.

# NLP

Before exploring the details of this project, it's crucial to define some key terms and concepts we'll encounter in the report. NLP, or Natural Language Processing, examines how computers and humans interact through text, aiming to understand language's structure and translate it into a form computer can interpret.

This translation often involves converting text into vectors, which is a multi-dimensional numeric representation. This vector representation can be assigned with multiple approaches, one of the most revolutionary approaches we will use in this work is call BERT, which is designed to understand the context of words in a sentence by looking at the words that come before and after them. For the context of NLP, the vectors are called embeddings. This numeric representation can be given to a word, sentence, or a document.

LLMs, or Large Language Models, are machine learning models trained on extensive text corpus. Almost all LLMs utilize a neural network architecture known as "Transformers". Transformers are distinguished by their ability to consider the position of words, as well as the attention and context surrounding them, allowing for a deeper understanding of language nuances. These models are also train and used for one or multiple tasks like: classification, summarization, feature extraction, text generation and more.

During this work, all models used are hosted on Hugging Face Hub and they are open source, Hugging Face also offers a working library called “transformers” which allows you to interact with the models.

# Sentence-BERT

BERT revolutionized NLP by introducing an embedding approach that assigns numeric representations to words, capturing their context within sentences. This advancement allows for the application of mathematical operations to identify patterns and similarities among words.

Sentence-BERT (SBERT) is an adaptation of the pre-trained BERT model that modifies the architecture to produce sentence embeddings. This allows for more efficient and semantically meaningful comparisons between sentences, enabling tasks like semantic similarity assessment, clustering, and information retrieval to be performed more effectively than with traditional BERT embeddings.

This capability is crucial for our approach, enabling us to recognize similarity across questions that may be phrased differently. For example, in an FAQ section, while one question might be "What are your opening hours?" another user might ask, "When are you open?" SBERT helps us to identify the semantic similarity between these two questions, ensuring that users receive the correct information regardless of how their query is worded.

The conversion of sentence to embedding is simply done with the python library call “sentence-transformers” created by the same people that made SBERT. This conversion doesn’t require any parameter setting or special configuration. It all comes down to the embedding model used for the conversion.

## Vector Database

Now that we can convert sentences into vectors, we need a way to store them, compare similarity between sentences and retrieve information. This can be done with a Vector Database, this is like any type of database but the key difference is that it uses the vector as the key value to identify each entry, then they are retrieve by comparing the similarity with another vector, this is done with a mathematical function called cosine similarity.

## Methodology

When using a Vector Database for storage and retrieval, we can efficiently manage a FAQ dataset for our chatbot. This strategy allows precise control over the chatbot's responses, significantly reducing the risk of inappropriate or inaccurate answers. We compiled a list of the 20 most frequently asked questions, spanning various topics typically addressed by our Academic Counsellor. Each question-answer pair will be inputted into the VD, with the question's vector serving as the identifier and the corresponding answer as the retrievable text. This setup enables the chatbot to provide accurate answers to questions that are semantically similar to those in the database.

## Models

To build the VD we need to choose an embedding model, we are going to test the efficiency of four different models. First, we build the database with 20 of the most frequently asked questions with their corresponding answers. As a safety measure, we also included questions that can capture inappropriate language and respond appropriately to it. At the same time, I included a greeting when users say “Thanks”.

An example of a question will be the following:

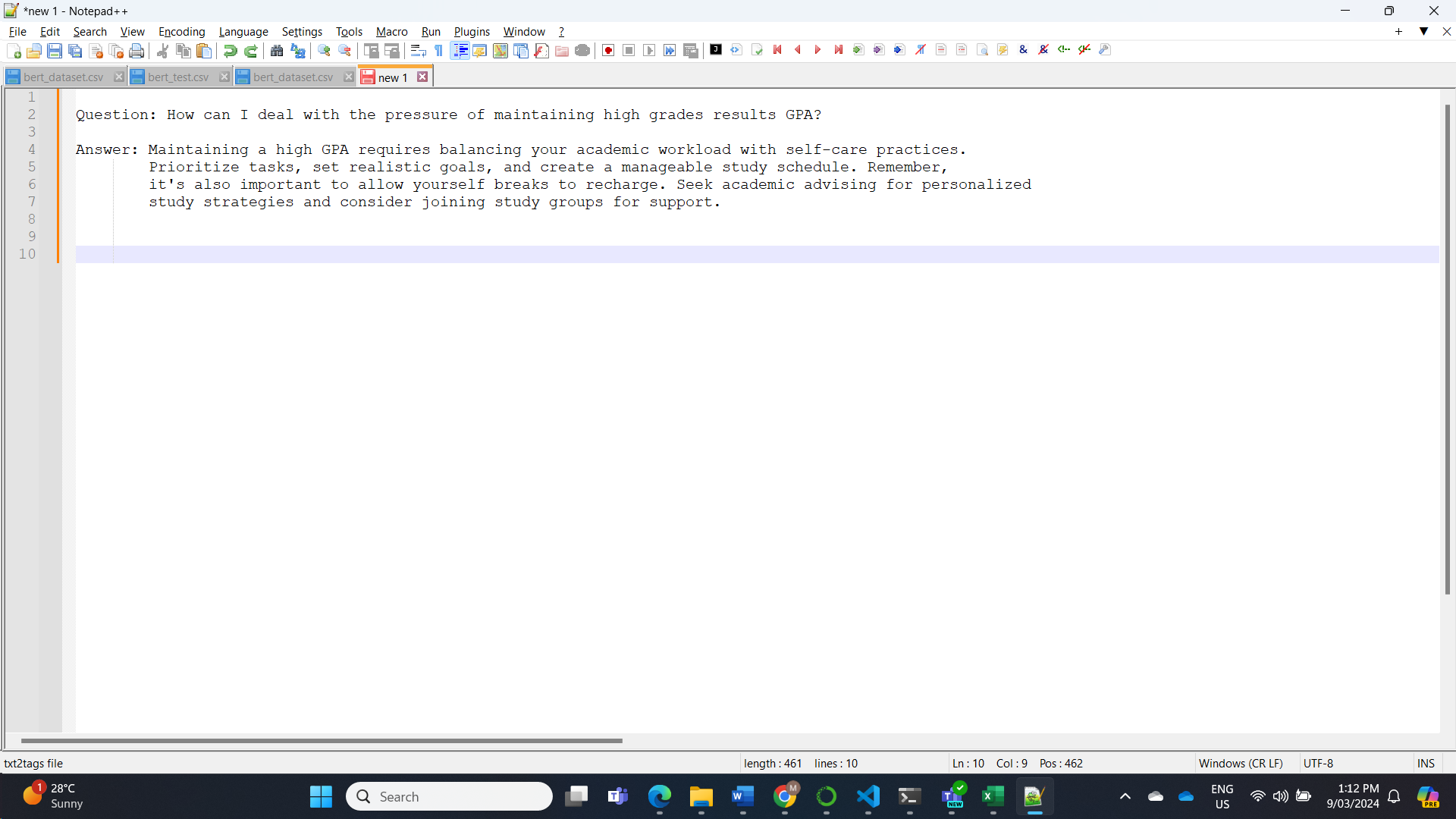


Figure 1, Example question.

From the example, is important to clarify that questions are not 100% grammatically correct, they are trying to “catch” the context of what emotion or topic is on question. Models used in the evaluation:

* **bert-base-nli-mean-tokens**, BERT network that use siamese and triplet network structures to derive semantically meaningful sentence embeddings. Fine tunned on NLI data to find cosine similarity across sentences.
* **sentence-t5-base** uses the pretrained *T5-base model* and fine-tuned in on a 1B sentence pairs dataset: “snli”, “wikihow”, “code\_search\_net”.
* **all-distilroberta-v1**, uses the pretrained *distilroberta-base model* and fine-tuned in on a 1B sentence pairs dataset: “snli”, “wikihow”, “code\_search\_net”.
* **all-MiniLM-L6-v2,** Uses the pretrained *microsoft/MiniLM-L12-H384-uncased model* and fine-tuned in on a 1B sentence pairs dataset: “snli”, “wikihow”, “code\_search\_net”.

## Evaluation and Results

To evaluate the performance of the models at matching different sentences on our dataset. I created an evaluation dataset, which takes 13 of the original questions but rephrases the question differently. The goal is that the models will output the correct answer, even if the question is asked different.

An example question will be the following:

A screenshot of a computer

Description automatically generated

Figure 2, Example test question.

Running the test on each of the models, we obtain the following accuracy results:



Figure 3, Test result of sentence embedding models.

## Risks

One of the major advantages of this approach, is the full mitigation of any bias and unethical behaviours from the model. From the initial test, we can guarantee a 92.3% accuracy on retrieving the correct responses, allowing the chatbot to still have a conversational feeling rather than just a categorical search of output.

Another risk mitigation strategy used in the data, is the identification of any rude behaviour our inappropriate language by the user, followed by an appropriate response from the model.

In terms of accuracy of the information given by the model, as we are in full control of the output given, the final dataset used by the chatbot can be created by professionals in the industry with a guarantee of the information given to the user.

## Deployment

From the list of results, we have a tie on the first place. Looking at the models **all-MiniLM-L6-v2** is only 80mb and **sentence-t5-base** is 210mb, making the first more suitable for the deployment stage.

To create a visual app for our chatbot, we use the library “Streamlit”, which allows us to host a local page and a visual interaction with the chatbot. Here we can see the instructions of usage, the query, and answers from the AI.

A screenshot of a computer

Description automatically generated

Figure 4, Example of deployed app.

# GPT-based for ‘text-generation’

For the second approach, we are going to interact with models that can perform “text-generation” as one of their tasks. As we discussed before, LLMs can perform multiple tasks, in approach 1 we used the models to convert sentences into a vector and maintain the context and meaning of the sentence.

In this approach we are going to work with text-generation, what the models do in this case, is predict the next sequence of words from an initial input. The goal is that the sequence of words is coherent and contextually relevant to the initial input.

This approach is aiming at overcoming the major shortcoming from using a predetermined Chat, by allowing the model at generating different answers based on the questions the user is giving.

## LangChain

There are several ways to interact with LLMs, either locally on your computer or online via hosted models. The choice depends on specific needs, such as:

* **Data Privacy:** For confidential information, running the model locally ensures data does not go online.
* **Computing Power:** Local models demand substantial memory and hardware for storage and processing during inquiries.

Going further on memory, our best sentence embedding model is only 80mb, in the other hand current best performing text-generation models can easily go over 200Gb. LLMs of this type, are measure on the number of parameters they have, going from millions up to hundreds of billions.

As we are currently researching best type of usages and models, we are going to interact with models that are hosted in Hugging Face Hub. This can be done for free with models that are up to 10Gb of space, and we will limit to models up to 7 billion parameters.

This interaction will be done with a platform called LangChain, which is one of the main libraries used to interact with LLMs. Besides just asking questions to the model, we are also going to try different “prompts”. Which is a way of asking the model questions and affect their behaviour.

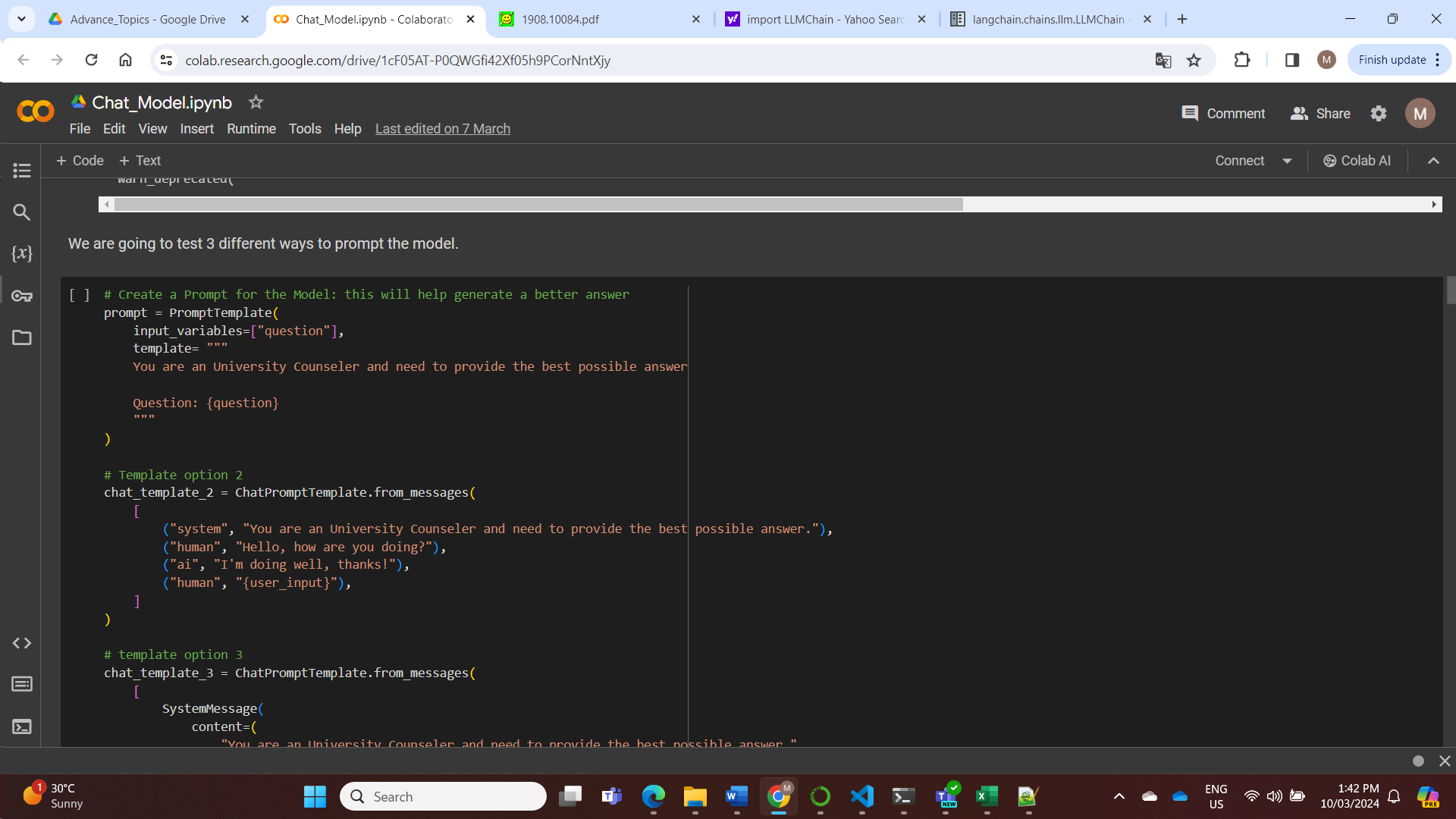


Figure 5, Prompt example.

In the example, we can see how we expect certain behaviour from the model, this allows us at mitigating any risk on the way questions are ask. I also tried to include “if you don’t know the answer say: I don’t know”, which it works well with our bigger models, but it doesn’t work well with our lighter models used to test.

## Methodology

All model interactions are done with the same “prompt”, at the same time we can also change the parameters on how the models predict the output. LLMs predict outputs from various probabilities, the parameters change the ranges on how those outputs are calculated and can change dramatically the result in just one model. This is another mitigation process to prevent the models to just fabricate answers.

The following are the main parameters we will modify for interaction:

* **min\_length** and **max\_length:** Set the minimum and maximum number of tokens in the output.
* **temperature:** Adjusts randomness in prediction; lower values lead to more predictable text.
* **top\_p (nucleus sampling):** Generates text based on the top p% probability distribution, allowing for more varied output.
* **early\_stopping:** Stops generation once the model predicts an end-of-sentence token if set to True.
* **length\_penalty:** Adjusts output length; values >1 encourage longer sequences.
* **num\_beams:** Number of beams in beam search; higher values can improve quality but reduce diversity.
* **no\_repeat\_ngram\_size:** Prevents repetition of n-grams to increase uniqueness in the text.
* **do\_sample:** Chooses whether to sample from the model's output distribution.
* **repetition\_penalty:** Penalizes repetition to encourage diverse and creative text generation.

After exploration of parameters, the following are the final values we are going to use in the evaluation of all models:

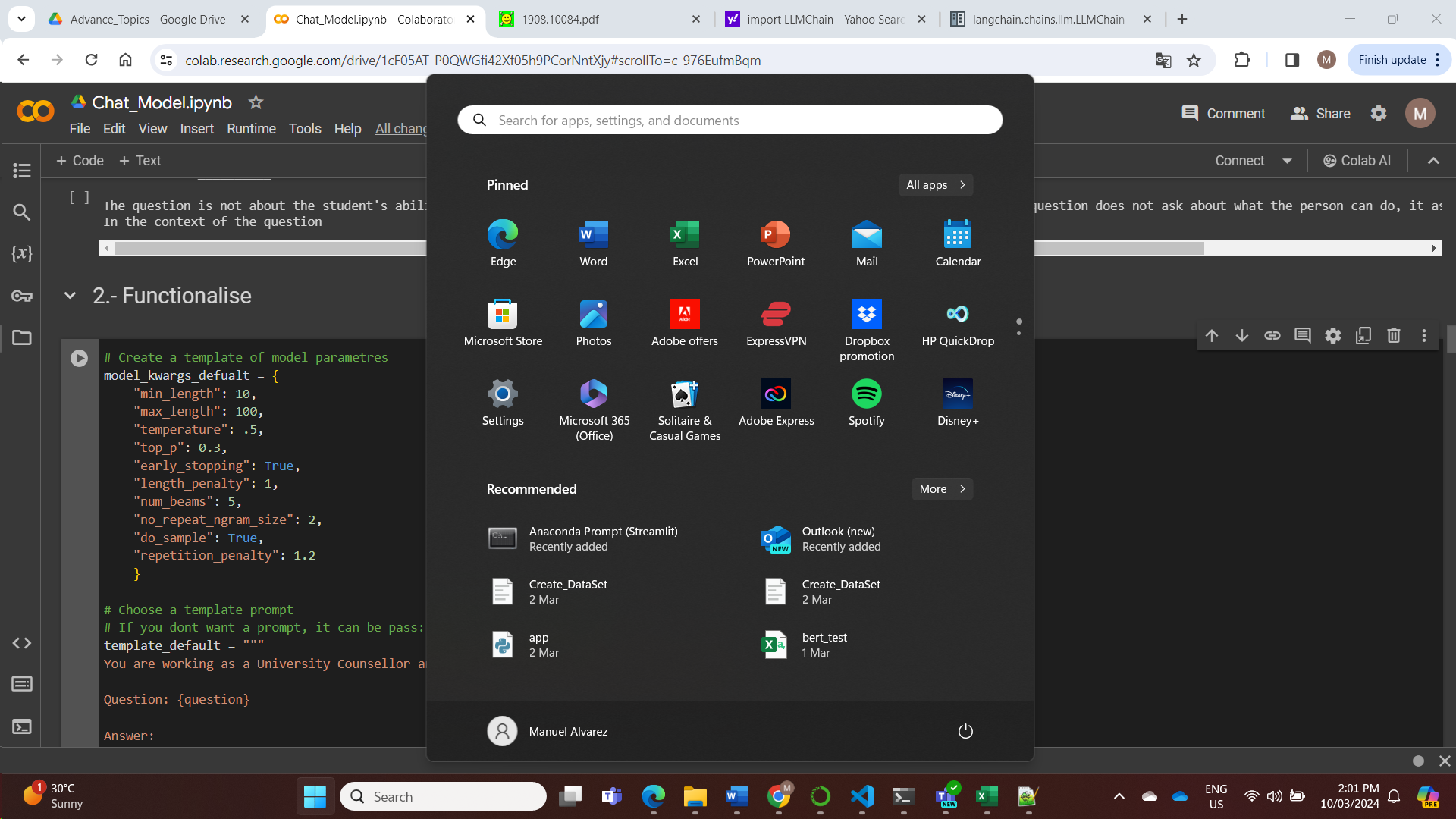


Figure 6, Final parameters.

## Models

For this task, we are going to use five different models:

* **gpt2,** created by OpenAI**.  
  -** year: 2018  
  - parameters: 137M  
  -data used for training: Scraped all the web pages from outbound links on Reddit which received at least 3 karma. Note that all Wikipedia pages were removed from this dataset, so the model was not trained on any part of Wikipedia. The resulting dataset (called WebText) weights 40GB of texts.
* **gpt-neo-2.7B,** designed using EleutherAI's replication of the GPT-3 architecture**.  
  -** year: 2021  
  - parameters: 2.72B  
  -data used for training: trained on the Pile, a dataset known to contain profanity, lewd, and otherwise abrasive language. Depending on your use case GPT-Neo may produce socially unacceptable text.
* **flan-t5-large,** created by the Google team, improvement of the T5 model. **-** year: 2022  
  - parameters: 783M  
  -data used for training: Trained on an extensive 750GB corpus of text known as the Colossal Clean Crawled Corpus (C4).
* **Mistral-7B-v0.1,** created by Mistral AI. **-** year: 2023  
  - parameters: 7B  
  -data used for training: Not available, they claim to use a more robust corpus than other models.
* **zephyr-7b-beta,** fine-tuned version of Mistral-7b by the Hugging Face team. **-** year: 2023  
  - parameters: 7B  
  -data used for training: fine-tuned on a mix of publicly available, synthetic datasets using Direct Preference Optimization (DPO).

## Evaluation and Results

To ensure that all models are tested equally, first we ensure that all parameters and prompts are the same. We took five questions from the previous approach and used them to question each of the models.

Initially I tried to use a more robust metric to measure the performance of the models, unfortunately this option didn’t work as the models require more human intervention into measuring what it will be consider a correct answer. From that we consider the following criteria:

* Context: Does the response accurately address the background and specifics of the question?
* Relevance: Is the response directly answering the question or providing useful information related to the query?
* Coherence: Does the response logically flow and maintain consistency in its argument or narrative?

With that in mind, I evaluated the modes on each of the five answers, in a binary scale:



Figure 7, Initial evaluation of LLMs

Looking at the results, we can identify the trend that as more parameters the model has, the better the score. This type of evaluation is subjective, and only looks if the outputs make sense and still on context to the topic. The evaluation doesn’t consider factors like the validity of the response, this will require a professional on the topic to judge.   
  
This evaluation and results, gives us a better understanding of which models can be more suitable for our tasks. I consider the results as an entry evaluation for later research, and a method to exclude models that are not suitable for the task.

This is an example answer of the models:

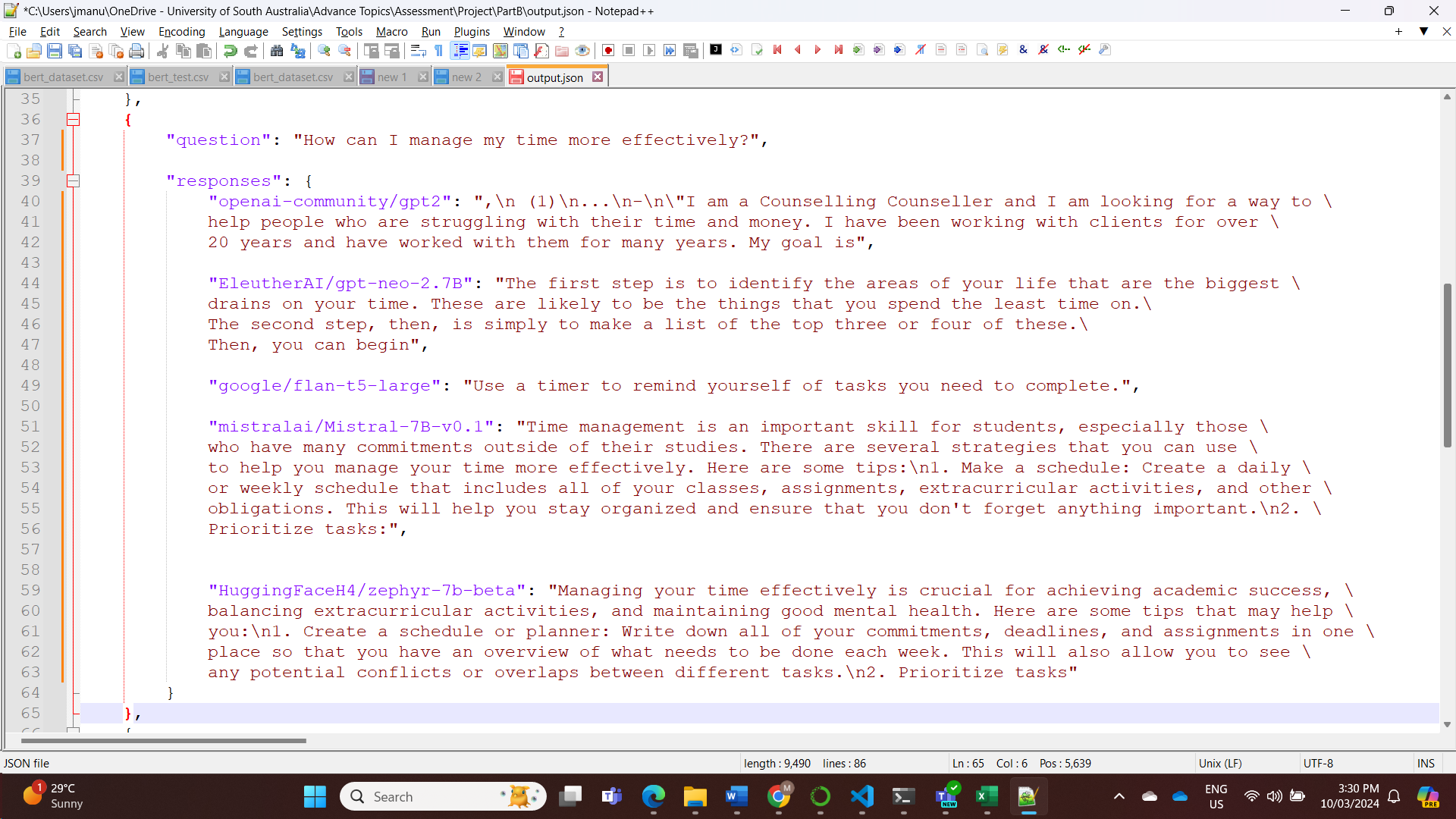


Figure 8, Example answer for question.

## Risks

During this process, we follow two important mitigations strategies:

* Use adequate prompts to ensure the answers are in context of the intended usage for the chatbot.
* Identify the parameters that minimise the creation of ambiguous or inexistent material.

Unfortunately, these strategies are not enough to fully eliminate any inherited risks, bias, or hallucinations from the model. During the testing, we can see some inaccuracies on the outputs and bias inherited from the data used to train the model. Some models might have some safeguards during their training, but we are out of control.   
  
An external tool that we can use to find out about inherited risks on the models, is the HELM project created by Stanford University. They have a list of the major LLMs, evaluated on various areas, including toxicity, bias, and unethical behaviours.

## Deployment

This approach was also deployed, in this case, we give the user the opportunity to choose any of the models we used and change the value of the parameters.

# Conclusion and Recommendation

The exploration of NLP for an Academic Chat-Bot through two distinct approaches has provided valuable insights. Approach 1, utilizing Sentence-BERT for embedding and matching, offers greater control over outputs, ensuring accuracy and minimizing risks—ideal given the chatbot's sensitive educational and emotional context. Approach 2, focusing on GPT-based text generation, creates more human-like interactions but introduces higher risks of generating inappropriate content. Given these findings, Approach 1 is recommended for its reliability and safety, though future work could refine Approach 2's parameters to better safeguard against risks while maintaining its conversational quality.

# Reference

[1] Han, Y., Liu, C., & Wang, P. (2023). A Comprehensive Survey on Vector Database: Storage and Retrieval Technique, Challenge. arXiv. https://arxiv.org/abs/2310.11703

[2] Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., Casas, D. de las, Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., & Sayed, W. E. (2023, October 10). Mistral 7B. ArXiv.org. https://doi.org/10.48550/arXiv.2310.06825

[3] Liang, P., Bommasani, R., Lee, T., Tsipras, D., Soylu, D., Yasunaga, M., Zhang, Y., Narayanan, D., Wu, Y., Kumar, A., Newman, B., Yuan, B., Yan, B., Zhang, C., Cosgrove, C., Manning, C. D., Ré, C., Acosta-Navas, D., Hudson, D. A., & Zelikman, E. (2022, November 16). Holistic Evaluation of Language Models. ArXiv.org. https://arxiv.org/abs/2211.09110

[4] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2018). Language Models are Unsupervised Multitask Learners. https://d4mucfpksywv.cloudfront.net/better-language-models/language\_models\_are\_unsupervised\_multitask\_learners.pdf

[5] Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., & Finn, C. (2023, May 29). Direct Preference Optimization: Your Language Model is Secretly a Reward Model. ArXiv.org. https://doi.org/10.48550/arXiv.2305.18290

[6] Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv. https://arxiv.org/abs/1908.10084

[7] Reimers, N., & Gurevych, I. (n.d.). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. SBERT.net. Retrieved 1st March 2024, from https://www.sbert.net/

[8] Salinas, A., & Morstatter, F. (2024). The Butterfly Effect of Altering Prompts: How Small Changes and Jailbreaks Affect Large Language Model Performance. arXiv. https://arxiv.org/abs/2401.03729

[9] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. arXiv. https://arxiv.org/abs/1910.01108

[10] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017, June 12). Attention Is All You Need. ArXiv.org. https://arxiv.org/abs/1706.03762

[11] Won, H., Le Hou, C., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Shixiang, A., Gu, S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., & Valter, D. (n.d.). Scaling Instruction-Finetuned Language Models Language model Instruction finetuning Chain-of-thought finetuning Inference: generalization to unseen tasks Multi-task instruction finetuning (1.8K tasks). https://arxiv.org/pdf/2210.11416.pdf