UNIVERSITÀ DEGLI STUDI DI SALERNO

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE ED ELETTRICA E MATEMATICA APPLICATA

Computer Engineering

NATURAL LANGUAGE PROCESSING AND LARGE LANGUAGE MODELS

Chatbot

Final project

Professors:
Prof. Antonio Greco
Prof. Nicola Capuano

Students:
Carmine Attianese
Giovanni Casella
Christian Salvatore De Angelis
Nunzio Del Gaudio

YEAR 2024/2025

Contents

Chapter 1

Introduction

This document is the final report for the project assigned at the end of the Natural Language Processing (NLP) course at the University of Salerno. The objective of the project was to develop a bot capable of answering questions exclusively related to the NLP and Large Language Models (LLM) course content, while avoiding responses that are out of context.

The main tools used in this project were Langchain, Ollama, and Hugging Face.

Langchain: Langchain is a framework designed to simplify the development of applications powered by large language models. It provides essential building blocks for working with language models, including components for prompt management, chaining different operations, and integrating external data sources, making it an ideal choice for building sophisticated NLP applications.

Ollama: Ollama is a tool for running and managing open-source language models locally. It offers a streamlined environment to deploy and experiment with LLMs on personal hardware, facilitating quick iterations and testing without relying on cloud-based solutions.

Hugging Face: Hugging Face is a leading platform in the NLP community, providing a comprehensive ecosystem for machine learning. It offers a vast repository of pre-trained models, datasets, and utilities, making it a cornerstone for any NLP project. Its Transformers library is particularly popular for working with state-of-the-art language models efficiently.

GradIO: GradIO is an open-source Python library that allows developers to create customizable user interfaces for machine learning models with minimal code. In this project, Gradio was used to build an interactive web-based interface for the NLP bot, enabling users to input questions and receive responses seamlessly. Its ease of integration and real-time feedback capabilities made it an essential tool for testing and demonstrating the bot's functionality.

Chapter 2

Building an initial chatbot

In this initial phase, we introduced the RAG mechanism in the chatbot, thus defining a text splitter, a vector store, and a retriever. All of this, processed by a chain, will then be inserted into the context along with the input.

We then focused on implementing memory to make the bot very similar to a chatbot that could answer questions while also considering the past conversation history.

The solution was to bring back the past history (chat history) into the prompt after each query. This is managed using an object that takes care of storing the past conversation, and the chain then uses it by reintroducing it into the input with the memory attribute.
Chain: **ConversationalRetrievalChain**, It is a chain specifically created for conversations with RAG that allows inserting a memory of past messages in addition to the retriever and the LLM.

The bot correctly answers questions about the proposed context, such as "What is meta-meta prompting?", thus showing good results in understanding the documents. It also responds well to some generic questions to encourage conversation.

However, as the conversation progresses, it becomes clear that the bot struggles to answer some questions because it has difficulty understanding the context. Additionally, the response time increases significantly.

- Model considered (decoder only): mistralai/Mistral-7B-Instruct-v0.2
- Embedding model considered: BAAI/bge-small-en-v1.5

Prompt

```
Prompt:
    Use the following pieces of context and previous conversation
    to answer the question at the end.
    Please follow these rules:
    1. If unrelated to context, reply: "I can only answer questions
    related to the provided notes."
    2. If unsure, say: "I can't find the final answer."
    3. Avoid offensive language.
    4. Keep answers concise (max 5 sentences).
    5. Consider conversation history.
    6. Answer in proper English.

    Previous conversation:
    {chat history}

    Context: {context}

    Question: {question}

    Helpful Answer:
```

Figure 2.1: First prompt we tried, it is simple and tries to keep the answer related to the context with few rules.

Chapter 3

A new model

To address the issues raised by the initial tests on the first Mistral model, namely:

- Text formatting
- Response time

It was decided to compare the answers, on the same chain, with another model: Llama3. This model was chosen because it is one of the best for RAG applications. Additionally, it was implemented locally using OLLAMA to try to reduce response time.

3.1 LLama3 test

LLama3 immediately proved to be better than the Mistral model in terms of answer quality. It understands past data better and responds accordingly. However, there were no improvements in response speed; in fact, it turned out to be slower. These tests revealed that the Mistral model can respond well, like LLama, when provided with precisely formatted data. On the other hand, LLama manages to provide answers even on incorrectly formatted data.

3.2 Resolving problems with new Text Splitter

Based on various tests, it was understood that the best structure for organizing the material for the RAG is to use a single file organized in such a way that paragraphs and sections are clearly distinguished from each other. This modification led to significant improvements in the model's performance, both in terms of response speed and the accuracy of the answers.
To make the bot more general and thus suitable for any type of file, another TextSplitter was considered for generating the embeddings to be inserted into the Vector store.

```
# Initial text splitter
recursive text splitter = RecursiveCharacterTextSplitter()

# New text splitter
semantic text splitter = SemanticChunker(self.embedding model)
```

The TextSplitter considered so far divides the text based on clearly distinct paragraphs or sections, implying a good division of the input text. The SemanticChunker splitter, on the other hand, divides the text based on the semantic meaning of the sections according to an embedding model passed to it as input.

3.3 Performance with the new Text splitter

The LLama model performs well with both splitters. The Mistral model significantly boosts performance, managing to respond well even to questions based on material that is not precisely formatted, using the new splitter. Additionally, for both models, the quality of the answers has improved, capturing references to the question even from distant parts of the text in the documents. From these attempts, it is clear that the LLama model processes the response better even if the context is not precise; however, it still remains significantly slower compared to the Mistral model.

Learning Word Embeddings

Word2Vec:

Introduced by Google in 2013:

- Based on neural networks.
- Uses unsupervised learning on large, unlabeled text corpora.

Continuous Bag-of-Words (CBOW):

- Predicts the central word based on surrounding words.
- Input: Multi-hot vector of surrounding tokens.
- Output: Probability distribution over the vocabulary.

Skip-Gram:

- Predicts surrounding words based on the central word.
- Effective for small corpora and rare terms.

Improvements to Word2Vec

Frequent Bigrams:

- Combines common word pairs (e.g., "Elvis_Presley").
- Focuses on meaningful predictions.

Word2Vec Alternatives

Glove: similar accuracy to word2vec, much faster and more effective on small corpora.It is based on singular value decomposition rather than neural networks .

FastText:

- based on sub-words,which predicts the n characters rather than the surrounding words.
- Effective for rare words, works on missed words and multiple languages.
- Generate word vectors even for unknown words

All these are**STATIC EMBEDDINGS** that is, each word is represented by a single static vector that captures the average meaning of the word based on the corpus.

But a word could have different meanings depending on the context : it's A PROBLEM

Another PROBLEM is that the semantics of a word changes according to time

Furthermore, word embedding could perpetuate and amplify social bias since the vector is static and depends on a fixed context (man is to doctor as woman is to nurse)

The datasets are full of content with BIAS.

WEs usually**They cannot process unknown words not present in the training data. FatText can.**

Lack of transparency:It can be difficult to interpret the vector space, difficult to analyze and improve the model or explain its behavior to stakeholders.

Figure 3.1: The notes in the first image are well-organized, the LLama model works well, while the Mistral model struggles. With the new text splitter, both models perform well with notes that any general user could input like in the second image.

Chapter 4

How we obtained a corrected formatted answers

In this phase, an attempt was made to resolve some issues that arose with both models. All models were able to stay within context, though not for all conversation turns. However, each model, with the previously shown prompt, displayed some errors: For normal questions like "What is RAG?", it responded with empty strings. For questions referring to the past, like "Can you give me some examples of it?", it often responded by repeating the same sentence or getting confused, showing examples of an unrelated past topic. As the conversation progressed, the model, in addition to going off-topic, started presenting the errors more frequently, making hard to chat.

4.1 Possibile solutions: new prompt

An attempt was made to resolve the issues by assuming that the prompt was not specific enough, adding an initial sentence in which the bot can better understand the context. Additionally, the question was enclosed in delimiters to clarify what the bot should actually respond to, in order to limit answers referring to past messages. Some rules, like rule number 8, were added to prevent the bot from responding randomly with strings such as spaces or numbers.

Many prompts similar to this one were tested, and one of the best results was achieved using the same prompt with the addition of "Think step by step" (which was later removed as it caused the bot to reason out of context) and "Give me only 1 complete result please." (kept in the final solution).

Prompt

```
You are a bot specialized in Natural Language Processing
answer questions relative to this context using the context provided
and the general Course question.
Use the following pieces of context and previous conversation
to answer the question after <<<>>>.
Please follow these rules:
1. If unrelated to context provided the answer to the question is:
I can only answer questions related to the provided notes.
Unrelated means that it does not concern the field of
Natural Language Processing
and good manners like "thanks".
2. Avoid offensive language.
3. Consider conversation history for question that reguard the past,
you are AI, the user is HUMAN.
4. Answer in proper English.
5. Question is the actual question you have to answer.
6. Just answer the question, don't start sayng "Based on the
provided context and previous conversation, I can answer your
question."
7. Motivate your answer.
8. If you don't know the answer, just say that you don't know,
don't try to make up an answer.

You have to follow these examples to respect the rules.
Right is what you should do, wrong is what you shouldn't do.
Example rule 6: Format of answer
Question: "What is RAG?"
Wrong Answer: "Based on the provided context and previous
conversation,  I can answer your question ... "
Wrong Answer: "Answer: "
Right Answer: "Rag stands for Retrieval ..."

Chat History:
{chat_history}

Context: {context}

<<<
Question: {question}
>>>

Answer:
```

Figure 4.1: New prompt, more specific with new rules and delimiters for the questions.

4.2 Results with new prompts

The results are remarkable: the bot responds much more frequently with meaningful text, it still doesn't fully respect the context, but for most conversation turns, it no longer confuses the actual question with the chat history. However, the issues with context remain, and as the conversation progresses, the bot significantly worsens, showing the initial problems that have not yet been fully resolved.

4.3 Possibile solutions: other models

It was then thought that the problem might lie in the model: perhaps the model struggles to understand more complex questions or is not trained on question-answering tasks with such large contexts.
Other models were then tested: **llama3-chatqa**: a llama model with fine-tuning on question answering. **wizardlm2**: an excellent model for handling complex chats. However, the results were not satisfactory; both models displayed the same issues as the initial two: random strings are still present, as well as confusion with the previous context.

4.4 Solution: a new embedder

The final solution was to test a new embedder, thinking that the previous one was not suitable for contexts as long as those a chatbot with RAG might present. As a result, the embedder was not able to understand the context well and link the different parts of the conversation together, leading to the chatbot receiving a query that did not accurately represent what was being asked. It was then decided to test: **nomic-embed-text**, a state-of-the-art embedding model that can handle very large context windows, implementable with OLLAMA. The results were excellent: the chatbot understands the query and the prompt well, no longer responding with random text strings, and often understands the context when referring to past conversations. The concern was that the computation might become slower, but this issue did not arise.

4.5 Performance with the new embedding model

At the end of this phase, the chatbot:

- Understands every type of document sent.
- Responds with properly formatted answers, not random strings or repetitions of text.
- Correctly understands which question to answer.
- Interprets the prompt well.

Problems:

- Although it responds correctly, it often answers out-of-context questions.
- As the number of conversation turns increases, the context expands, consequently increasing the probability of answering out-of-context questions.

Chapter 5

How we obtained a model that answers only in-context questions

With the previous prompt, the bot, using all the models considered, is able to respond well to questions, and often recognizes when they go out of context, refusing to answer. However, the main issues arise primarily when:

- The bot is overwhelmed with questions, filling the chat history and changing the context.
- The bot is asked questions about topics similar to those discussed, but that have not actually been covered.
- The bot recognizes that the question is out of context but still provides an explanation.

In this phase, we aim to implement guardrails to define the system's behavior.

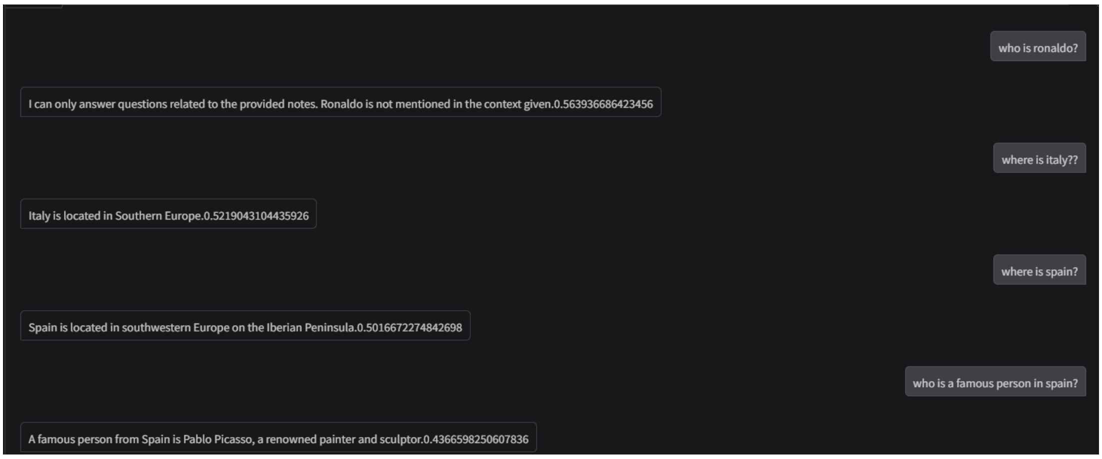

Figure 5.1: After a good response, the model answers out of context question.

5.1 Possibile solutions: temperature and rephrase the question with prompt chaining

The first solution attempted was to modify the parameters of the models (focusing only on the Mistral and Llama3 models), particularly the temperature: it was thought that the model was going out of context because it was generating less probable tokens due to the temperature being too high. The temperature was then lowered for both models to low values (between 0.1 and 0.4) to make the generation deterministic.

It was noticed that the generated responses were shorter and more concise, however, it was understood that this did not affect the model, as despite being smaller in size, the responses still went out of context. Consequently, it was thought that the problem was the formulation of the question being too generic. Users often ask questions like "What is a transformer?", but perhaps a formulation like "What is a transformer neural network in natural language processing?" could be more accurate and constrain the model to give more precise answers.

Therefore, it was decided to expand the Chain by adding an additional request to the LLM model, asking it to rephrase and return the same question based on the past context. This question would then be passed to the actual model, implementing a sort of Prompt Chaining. This solution definitely brought improvements, the bot confused less frequently and the answers also seemed more precise.

Prompt

```
adapt_template = """
Adapt the following question
relative to Natural Language Processing.
Question: {question}
Answer with only one single better structured question.
Give me only 1 result please.
"""
```

However, all of this came at too high a cost: the bot, already slow, in the case of LLama, became even slower and was very inconvenient for the user to use. Moreover, the problem was not completely solved, and as a result, the solution was discarded.

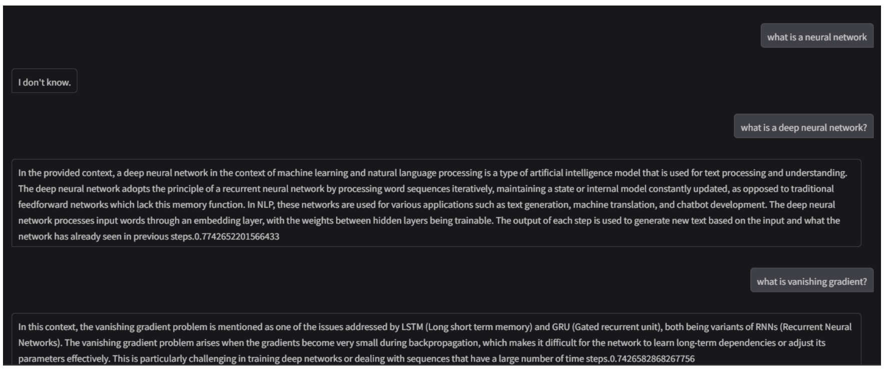

Figure 5.2: The bot is asked questions about topics similar to those discussed, but that have not actually been covered and it answers.

5.2 Possibile solutions: Fact-checking model and discard based on similiarity

Another solution attempted was to use an additional model for fact checking, to determine if the answer provided was indeed relevant to the context. The model used was: bespoke-minicheck, which takes as input a document or a sentence and a claim. It then returns 'Yes' or 'No' depending on whether the claim is supported by the document. Example:

- **Document:** A team of researchers conducts an experiment in the laboratory to test their new hypothesis on chemical reactions.
- **Claim:** The researchers are performing a scientific experiment.
- **Answer:** Yes

In our case, we input the documents from which the query is extracted and use the model's response as the claim. This is then output only if the fact-checking model responds with 'Yes'; otherwise, the output is "I don't know". This solution was discarded because, while it is faster than the previous one, it is incorrect in many cases, probably because the response returned consists of a set of different documents, and the model is unable to distinguish between them in this context.
The last attempt was to implement a function that calculates the cosine similarity between the model's response and the documents from which it is extracted. If this exceeds a certain threshold, the response is output; otherwise, "I don't know" is printed. This is the solution that brought us closest to a correct model because it addresses the following issue:

- The bot is asked questions about topics similar to those discussed but that have not actually been covered.

In these cases, the similarity is low, and consequently, the answer is not considered correct because it is not supported by any sufficiently similar document. The weakness of this solution is that the threshold could discard correct answers, so a good fine-tuning of the threshold is needed. Additionally, it does not solve the problem of:

- When the bot recognizes that the question is out of context but still explains it.

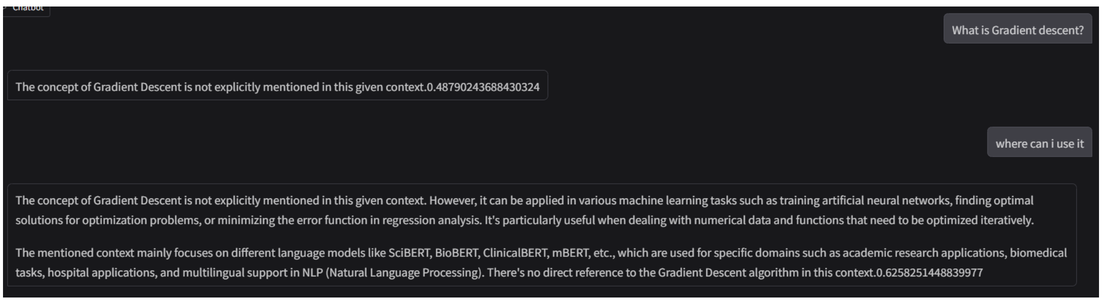

Figure 5.3: the bot recognizes that the question is out of context but still explains it.

5.3 Solutions: Prompt with few shot learning

The solution that led us to the final result was to rewrite the prompt with additional restrictions:

```
"You know only the information explicitly provided by the context,
don't try to answer to something else or to give
more information, do not think."
```

This clearly tells the robot what it actually knows.

```
Example of how to answer if the concept is not explicitly
mentioned in the context:

QUESTION: What is Gradient Descent? (Not mentioned in the context)

WRONG ANSWER: The concept of Gradient Descent is not explicitly mentioned
in the given context. However, it is a common optimization algorithm used to
minimize some function by iteratively moving in the direction of steepest
descent as defined by the negative gradient.

RIGHT ANSWER: The concept of Gradient Descent is not explicitly
mentioned in the given context.
```

This example is provided by indicating a possible question and two answers, one correct and one wrong, showing the model that it should not explain if it recognizes that something is not present within the context. Only these additional instructions brought significant improvements in the responses, further refined by an additional example and the following:

```
When you say: The ARGUMENT is not explicitly mentioned in this given context,
give a summary of the context and don't infer about the argument.
```

This clarifies the behavior the model should follow in the case of an unknown answer.

New Prompt

```
You are an agent specialized in Natural Language Processing.
Use the following pieces of
context to answer the question after <<<>>>.
Do NOT provide any explaination if something is not present in
the context!

Please follow these rules:
1. If unrelated to context provided the answer to the question is:
I can only answer questions related to the provided notes.
Unrelated means that it does not concern the field of Natural
Language Processing.
You know only the information explicitly provided by the context,
don't try to answer to something else or to give more information,
do not think.

Example of how to answer if the concept is not explictly mentioned
in the context:

QUESTION: What is Gradient Descent? (Not mentioned in the context)
WRONG ANSWER: The concept of Gradient Descent is not explicitly
mentioned in the given context. However, it is a common optimization
algorithm used to minimize
some function by iteratively moving in the direction of steepest
descent as defined by the negative gradient
RIGHT ANSWER: The concept of Gradient Descent is not explicitly
mentioned in the given context.

QUESTION: WHAT IS THE CANNY ALGORITHM? (Not mentioned in
the context)
WRONG ANSWER: The Canny Algorithm is not mentioned in this
given context,
so I can't provide an accurate answer based on the provided details.
The context only discusses Natural Language Processing (NLP) tasks
such as understanding and generation, and various models like
SciBERT,  BioBERT, ClinicalBERT, mBERT, FlanT5, ByT5, T5-3B,
T5-11B, Ul2,
Multi-Modal T5, Efficient T5, etc. The Canny Algorithm is typically
used in image processing for edge detection and filtering
RIGHT ANSWER: The Canny Algorithm is not mentioned in this
given context.
```

New Prompt

```
When you say: The ARGUMENT is not explicitly mentioned in this
given context, give a summary of the context and don't infer
about the argument.
When you say: is not explicitly mentioned in this given context.
Report where you found it in the context.

2. Avoid offensive language.
4. Answer in proper English.
5. Question is the actual question you have to answer.
6. Just answer the question, don't start sayng "Based on
the provided context,
I can answer your question."
7. Motivate your answer.
Give me only 1 complete result please.


Chat History:
{chat_history}


Context: {context}

<<<
Question: {question}
>>>

Answer
Before answering, check if the provided context contains
the necessary information. If yes, answer based on it. If not,
reply  I don't know and don't try to answer.
"""
```

Figure 5.4: Final prompt, additional restrictions for the context and implementation of few shot learning.

5.4 Performance with the new prompt

This solution leads to the definitive resolution of the context-related issues. With cosine similarity, the problem was solved:

- The bot is asked questions on topics similar to those covered but that have not actually been discussed.

And now the problem has also been solved:

- When the bot recognizes that the question is not in context but explains it anyway.

These solutions could be implemented together; however, testing the system with only prompting led to the same results as using cosine similarity, which has the additional difficulty of threshold tuning. For this reason, it was decided to remove cosine similarity and retain only the prompting.

Chapter 6

How we obtained a time-resistant model

The last issue to address concerns:

- The bot gets overwhelmed with questions, filling up the chat history and changing the context.

Even though the final prompt explicitly mentions taking into account only the last question and the context, as the conversation progresses, the bot gets confused by questions that lack explicit references, such as: **"Give me some examples"**
Moreover, it has happened that after providing information on a specific topic, such as **"football"**, even though the model initially refuses to answer, as the conversation continues, it starts to show signs of yielding, gradually offering some explanations.
It was hypothesized that the cause was related to the fact that the memory used in the chain retained the entire conversation. As a result, several tests were conducted to replace it and verify this hypothesis.
Initial memory: `ConversationBufferMemory`, which keeps track of all conversation turns.

6.1 Solution: SummaryMemory and WindowMemory

These are the types of memory considered for the test:

ConversationSummaryMemory: It returns a summary of the conversation up to this point. This solution is effective; however, it requires another model to actually summarize the conversation. Like previous solutions that use additional models, this one is also slow. Moreover, the summary risks altering the meaning of the conversation if not performed with an optimal prompt. Consequently, this solution was discarded.

ConversationBufferWindowMemory: It considers only the last k conversation turns. This solution does not use a model, so it does not slow down the system. The results are as positive as with the summary memory. The model rarely confuses the context and is able to understand which questions to answer, even if they do not directly refer to the context. Therefore, this was the chosen solution.

This choice is also supported by the fact that it is rare for a user to refer to questions asked at the beginning of the conversation.
An additional solution could be to create a hybrid-performing memory summary once after k conversation turns. However, the window memory alone resolves the issues effectively.

6.2 Speed the model

A final problem to solve is precisely the response speed. Beyond the choice of the model, another solution was adopted.
It was decided to limit the number of tokens generated, as the time the model takes to generate the response is directly proportional to the time it takes to generate individual tokens.
As a result, both the Mistral model and the Llama model are limited to 512 possible new tokens that can be generated.
This solution improved the response speed.
Another possible solution could be to show the tokens in real-time as they are generated, giving the user the illusion that the model is actively producing something.

Chapter 7

Test and final model

To choose the final model between LLama3 and Mistral, a final test of 10 questions was conducted, taking into account the following factors:

- Number of correctly answered questions
- Questions answered out of context
- Response speed
- Response format

The questions asked are both about the course and its topics, as well as out of context. They also attempt to test the bot's ability to refer to past conversations without getting confused.

7.1 LLama3 vs Mistral

Below we show 5 of the 10 questions asked to LLama3 and Mistral

who are the professors?

LLAMA3

```
The professors mentioned in the given context are:
- Nicola Capuano (DIEM  University of Salerno)
    Email: ncapuano@unisa.it
- Antonio Greco (DIEM  University of Salerno)
    Email: agreco@unisa.it
```

Mistral

```
The professors mentioned in the given context are:
- Nicola Capuano (DIEM { University of Salerno)
    Email: ncapuano@unisa.it
- Antonio Greco (DIEM { University of Salerno)
    Email: agreco@unisa.it
```

What is a trasformer?

LLAMA3

```
The concept of a transformer is explicitly mentioned in
the given context.  According to the text, "Transformer
is just the architecture to create the language model" and
later it's mentioned that "Transformers:  t5, bart, mt5".
This suggests that a transformer is an architecture used to
create language models.
```

Mistral

```
The concept of a transformer is explicitly mentioned in
the given context.  According to the text, "Transformer
is just the architecture to create the language model" and
later it's mentioned that "Transformers:  t5, bart, mt5".
This suggests that a transformer is an architecture used to
create language models.
```

what is a trasformer? and who is optimus prime?

LLAMA3

```
According to the given context, "Transformer is just the
architecture to create the language model" and later
it's mentioned that "Transformers:  t5, bart, mt5".  This
suggests that a transformer is an architecture used to
create language models.
```

Mistral

```
According to the given context, "Transformer is just the
architecture to create the language model" and later
it's mentioned that "Transformers:  t5, bart, mt5".  This
suggests that a transformer is an architecture used to
create language models.
```

where is spain?

LLAMA3

```
I can only answer questions related to the provided notes.
The concept of Spain is not explicitly mentioned in the
given context.  However, it discusses Natural Language
Processing (NLP) tasks such as understanding and generation,
and various models like SciBERT, BioBERT, ClinicalBERT,
mBERT, FlanT5, ByT5, T5-3B, T5-11B, Ul2, Multi-Modal T5,
Efficient T5, etc.
```

Mistral

```
I can only answer questions related to the provided notes.
The concept of Spain is not explicitly mentioned in the
given context.  However, it discusses Natural Language
Processing (NLP) tasks such as understanding and generation,
and various models like SciBERT, BioBERT, ClinicalBERT,
mBERT, FlanT5, ByT5, T5-3B, T5-11B, Ul2, Multi-Modal T5,
Efficient T5, etc
```

who are the professors?

LLAMA3

```
I can only answer questions related to the provided notes.
The context discusses various NLP tasks, models, and
applications, but does not explicitly mention machine
learning.
```

Mistral

```
I can only answer questions related to the provided notes.
The context discusses various NLP tasks, models, and
applications, but does not explicitly mention machine
learning.
```

7.2 How we chose the final model

From the tests, it emerged that both models correctly answer the questions; however, the LLAMA3 model seems to be more knowledgeable about the context. When asked "What is machine learning?", the Llama3 model responds bluntly, stating that it is not included in the context, while the Mistral model, although recognizing that it is out of context, tries to provide an answer based on the information it has. The choice, however, fell on the Mistral model because, on average, it takes 8 seconds to respond with complete answers, while the LLama model takes about 5 seconds longer and, in the worst case, even 10 seconds more.

7.3 Conclusions

The chosen model responds accurately to the questions posed in a quick manner, rarely goes off-topic, and manages not to get confused with previous messages.
The system works well with various types of documents thanks to the selected embedder and splitter.
It does not respond to generic phrases like "Hello" or "Thanks" as it considers them out of context. This aspect was not deemed necessary for the project since the bot behaves correctly concerning the task for which it was programmed. However, the prompt could be modified to include this and make the conversation with the user smoother.
The model does not respond in a toxic manner; from the beginning, it has performed well in this regard. Nonetheless, a rule was added to the prompt to further mitigate this aspect.
To improve the system and make it more user-friendly, one could try to personalize the way it responds: it often repeats "in the context," "based on the context," and this could become repetitive. With further prompt engineering, this could be resolved, as well as with Reinforcement Learning techniques to customize the response.