

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE TELECOMUNICACIÓN



MÁSTER UNIVERSITARIO EN INGENIERÍA DE
TELECOMUNICACIÓN

TRABAJO FIN DE MÁSTER

DESIGN AND IMPLEMENTATION OF A
LLM-BASED CHATBOT PROTOTYPE
ENDOWED WITH MEMORY MECHANISMS

ALFREDO GARRACHÓN RUIZ

27 de junio de 2024

MÁSTER UNIVERSITARIO EN INGENIERÍA DE TELECOMUNICACIÓN

TRABAJO FIN DE MÁSTER

Título: Design and implementation of a LLM-based chatbot prototype endowed with memory mechanisms.

Autor: D. Alfredo Garrachón Ruiz.

Tutor: D. Luis Fernando D'Haro Enríquez.

Departamento: Dpto. de Ingeniería Electrónica

MIEMBROS DEL TRIBUNAL

Presidente: D.

Vocal: D.

Secretario: D.

Suplente: D.

Los miembros del tribunal arriba nombrados acuerdan otorgar la calificación de:
.....

Madrid, a de de 20...

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE TELECOMUNICACIÓN



MÁSTER UNIVERSITARIO EN INGENIERÍA DE
TELECOMUNICACIÓN

TRABAJO FIN DE MÁSTER

**Design and implementation of a
LLM-based chatbot prototype
endowed with memory mechanisms**

Alfredo Garrachón Ruiz

27 de junio de 2024

Abstract and Key Words

Abstract

The introduction of LLMs in chatbots has been a revolution in this field with models that are much more capable and intelligent than those that existed previously. Thus, although research and experimentation with these models is rising, there is an aspect, to some extent, forgotten in this research focused mainly on improving their intelligence: the proximity and personalization of these chatbots.

With this objective in mind, in the present project a new chatbot architecture, based on LLMs, with long-term memory over the user's personal data has developed. This development has involved the study of the state of the art of technology related to this topic and the study of human memory to take inspiration from it. Thus, the two vital processes in a memory architecture have been designed: the process of remembering and the process of memorizing.

For the latter, a novel technique using the capabilities of LLMs has been proposed. In order to make use of it, its use has been validated by tests involving volunteers for the human subjective factor and tests specifically designed for LLMs. A new score formula has also been designed to compare the performance of different LLMs in this task.

After analyzing the results, it was concluded that the LLMs are capable of extracting memorable personal data from a conversation in a manner similar enough to that of a person to validate their use. Thus, using the score formula developed, it has also been concluded that the LLM "Llama 2 13b" is the one that best performs this task among the 4 LLMs compared.

Once the validation of the processes has been carried out, the final architecture has been designed and developed involving both processes of memorizing and remembering personal data. At the end of the project a demo of the chatbot was carried out to demonstrate the added value it provides in a conversation.

The final result was a new chatbot capable of memorizing the most important personal data of the user, making use of this data at the most appropriate moments of the conversation. In this way, the chatbot manages to deliver the feeling of closeness, interest in the user and personalization.

The repository with all generated data, evaluation results and scripts necessary for the experimentation and execution of the chatbot system have been released in open-source form for free use in the corresponding repository: [Official Repository](#)

Resumen

La introducción de los LLMs en los chatbots ha supuesto una revolución en este campo con modelos mucho más capaces e inteligentes que los que existían previamente. Así, pese a que la investigación y experimentación con estos modelos está en auge, existe un aspecto, en cierta medida, olvidado en esta investigación centrada principalmente en mejorar su inteligencia: la cercanía y la personalización de estos chatbots.

Con este objetivo en mente, en el presente proyecto se ha desarrollado una nueva arquitectura de chatbot, basada en los LLMs, con memoria a largo plazo sobre los datos personales del usuario. Este desarrollo ha involucrado el estudio del estado del arte tecnológico relacionado con esta temática y el estudio de la memoria humana para tomar inspiración de esta. Así, se han diseñado los dos procesos vitales en una arquitectura de memoria: el proceso de recordar y el proceso de memorizar.

Para este último, se ha propuesto una novedosa técnica utilizando las capacidades de los LLMs. Para poder hacer uso de esta, se ha validado su uso mediante tests involucrando

voluntarios para el factor subjetivo humano y tests diseñados específicamente para los LLMs. Se ha diseñado una nueva fórmula que permite comparar el rendimiento de diferentes LLMs en esta tarea.

Tras el análisis de los resultados, se ha concluido que los LLMs son capaces de extraer datos personales memorables de una conversación de una forma suficientemente similar a como lo haría una persona como para validar su uso. Así, haciendo uso de la fórmula score desarrollada, se ha concluido también que el LLM "Llama 2 13b" es el que mejor realiza esta tarea entre los 4 LLMs puestos en comparación.

Una vez llevada a cabo la validación de los procesos, se ha diseñado y desarrollado la arquitectura final involucrando ambos procesos de memorizar y recordar datos personales. Al final del proyecto, se ha llevado a cabo una demo del uso del chatbot para demostrar el valor añadido que proporciona este en una conversación.

Como resultado final, se ha obtenido un nuevo chatbot capaz de memorizar los datos personales más importantes del usuario, haciendo uso de estos datos en los momentos más adecuados de la conversación. De esta forma, el chatbot consigue entregar la sensación de cercanía, de interés hacia el usuario y de personalización.

El repositorio con todos los datos generados, resultados de la evaluación y scripts necesarios para la experimentación y ejecución del sistema chatbot se han liberado de forma open-source para su libre uso en el repositorio correspondiente: [Repositorio Oficial](#)

Key Words

Artificial Intelligence, LLMs (Large Language Models), NLP (Natural Language Processing), Memory Architecture, "Transformers", Chatbot, Human Memory.

Palabras Clave

Inteligencia Artificial, LLMs (*Large Language Models*), NLP (*Natural Language Processing*), Arquitectura de Memoria, "Transformers", *Chatbot*, Memoria Humana.

Greetings

Por acompañarme durante el camino que me ha llevado hasta este hito, quiero agradecer:

A mi familia por criarme y enseñarme los valores que me definen.

A Tiro y Tejo por darme la serotonina necesaria de cada día.

A los “Fotónicos” por sacar interés de donde solo había transistores.

A los “Indimaestros” por vivir como en la “mili” de la mejor forma posible.

A “Derecha” poluisfernando.dharo@upm.esr demostrarme que hasta la gente más lista no sabe elegir un nombre.

Y a todos los que, sin estar etiquetados, aportan más que nadie.

Where there's a will, there's a way, kind of beautiful. - Tim Bergling -

Contents

Abstract and Key Words	i
Greetings	iii
Contents	iv
Figure Index	vii
Table Index	ix
1 Introduction	1
1.1 Problem description	1
1.2 Innovation	1
1.3 Objectives	2
1.4 Document Structure	2
1.5 Conclusions Insights	3
2 State of Art	4
2.1 Large Language Models	4
2.1.1 Overview LLMs	4
2.1.2 Transformers	5
2.1.3 Prompt Engineering	7
2.1.4 New Models Architectures	8
2.1.5 LLMs with large context windows	9
2.2 Human Memory	10
2.2.1 Introduction	10
2.2.2 Memory Systems	11
2.2.3 Explicit and Implicit Memory	11
2.2.4 Declarative or Explicit Memory	12
2.2.5 Functional memory	13
2.2.6 Memory theory applied to chatbots	15
2.2.7 Inspiration for the memory architecture	16
2.3 Towards Longer Memory	16
2.3.1 Optimizing the memory management	17
2.3.2 Augmenting Context Window	18
2.3.3 Longer Memory involving re-training	20
2.3.4 Longer memory without re-training	21

2.4 Suite Architectures	25
2.5 Databases	26
2.5.1 Different types of databases	27
2.5.2 Non-Relational Databases	27
 3 Validation of the Architecture	 27
3.1 Initial Architecture	28
3.2 Preparation of validation of memorizing process	29
3.2.1 Initial thought	29
3.2.2 Selection of the LLMs	30
3.2.3 Design of the human evaluation test	32
3.2.4 Conversational Data	34
3.2.5 Preparation of the environments of LLMs	36
3.2.6 Prompt engineering on LLMs	37
3.2.7 Human Evaluation	40
3.3 Validation of memorizing process	43
3.3.1 Consistency among Users	44
3.3.2 Deterministic trusted semantic comparison	46
3.3.3 Score Formula Design	49
3.3.4 Score Formula Evaluation	51
3.3.5 Score on time projections	54
 4 Evaluation Results	 55
4.1 Validation of the Hypothesis	56
4.2 Results of extracting memories	57
4.3 Results for time projection	57
 5 Development of the System	 59
5.1 System Architecture	60
5.1.1 Architecture Overview	60
5.1.2 Conversational LLM	60
5.1.3 LLM for extracting memories	61
5.1.4 Database of memories	62
5.1.5 Central computing unit	62
5.2 Main Process	63
5.2.1 System Configuration and Initialization	63
5.2.2 Design of the conversational LLM prompt	65
5.2.3 Main Process Logic	65

5.3 Memorization Process	66
5.3.1 General Idea behind the Memorization Process	67
5.3.2 Prompt Optimization	67
5.3.3 Logic behind the memorization process	68
5.4 Recall Process	70
5.4.1 General Idea behind the Recall Process	70
5.4.2 Score Function of the Recall Process	71
5.4.3 Logic behind of the Recall Process	72
6 Execution of the System	73
6.1 Initiation of the System	73
6.2 System Demo	74
6.2.1 Initial Conditions	74
6.2.2 Conversation Demo	75
6.2.3 Memorization Process Demo	79
7 Conclusions	81
7.1 Future Work	81
Índice de Tablas e Imágenes	83
A Annex: Ethical, economic, social and environmental issues	93
A.1 Introduction	93
A.2 Description of relevant impacts related to the project	93
A.3 Detailed analysis of some of the main impacts	94
A.4 Conclusions	95
B Annex: Financial Budget	96
C Annex: Examples of conversational dataset	97
D Annex: Task delivered to human annotators	102
E Annex: Log information of the example of execution	103

Figure Index

1	Transformer Architecture	7
2	A taxonomy of mammalian long-term memory systems	11
3	Memory Optimization of FlashAttention	17
4	Structure Recursive Summarization	22
5	Overall Structure MemGPT	23
6	Memory Architecture of Interactive Simulacra	24
7	Benchmarks comparison between models	32
8	Diagram of test operation	34
9	Example of conversation structure	36
10	Instruction used for the comparison experiment	40
11	First prompt used for the comparison experiment	40
12	Second prompt used for the comparison experiment	41
13	Example of test proposal to the human volunteers	42
14	Exemplification of the inter annotator performed	45
15	Deterministic Semantic Comparison	52
16	Manual Semantic Comparison	52
17	Final Scores Task Comparison	54
18	Scores over Time Projections	55
19	Abstract of the system architecture	61
20	Prompt designed for Conversational task LLM	65
21	New prompt designed for LLama2 specific	68
22	Initial Data Base Conditions of the Demo	74
23	Conversation of the demo	76
24	First relevant aspect of the conversation	76
25	Second relevant aspect of the conversation	77
26	Third relevant aspect of the conversation	78
27	Fourth relevant aspect of the conversation	78
28	Fifth relevant aspect of the conversation	79
29	Memorization Process of the Demo Example	79
30	Contents of the database after the demo conversation	80
31	First example of Conversational Dataset	97
32	Second example of Conversational Dataset	97
33	Third example of Conversational Dataset	98
34	Fourth example of Conversational Dataset	98
35	Fifth example of Conversational Dataset	99
36	Sixth example of Conversational Dataset	99

37	Seventh example of Conversational Dataset	100
38	Eighth example of Conversational Dataset	100
39	Ninth example of Conversational Dataset	101
40	Tenth example of Conversational Dataset	101
41	Example of the Task delivered to the volunteers	102

Table Index

1	Benchmarks comparison between models	32
2	Statistics of the conversations	36
3	Deterministic semantic comparison between models	49
4	Manual semantic comparison between models	53
5	Final Scores Task Comparison	53
6	1 Week Projection Scores Comparison	55
7	2 Months Projection Scores Comparison	55
8	1 Year Projection Scores Comparison	56
9	Economical budget	96

1 Introduction

Day by day we see how artificial intelligence is growing exponentially in practically all aspects of our lives. More specifically, the large language models (LLMs) are the ones that have had a more widespread acceptance in everyday tasks and are demonstrating a growth that was thought impossible a few years ago.

These specific models have had such an impact on society because of the way they interact. They are controlled by human language in a very simple way and are capable of solving a great variety of problems, from doubts about knowledge to human work tasks such as programming or even resolving complex equations.

Thus, an area of daily life where these LLMs have a lot of projection is based on communication as chatbots. Due to the intrinsic nature of the models, they adapt very well to this aspect, since they are developed with a text-to-text architecture, so it is easy to think that it is possible to have a conversation with them, either written or oral, by means of human language.

Although the advancement of technology has provided us with much more capable LLMs, which in terms of intelligence could resemble humans, they still lack a touch that makes them attractive to have conversations with: personalization and closeness. These models in their current state present a remoteness when conversing with them that leads to stop conversing with them due to the lack of a sense of interest or empathy.

1.1 Problem description

Researchers and developers are making great efforts to develop larger and larger models, with greater problem-solving capabilities and covering a much wider field of possibilities. Thus, these efforts are focused on increasing the capabilities of the models, but there is one area that requires more research: user personalizaion and closeness. There exists a significant potential to investigate the customization of LLMs for delivering personalized user experiences, enabling individuals to receive responses that are specifically tailored to meet their unique requirements [1].

Current research in customized large language models (LLMs) focuses mainly on designing comprehensive prompts that enable LLMs to generate contextually relevant responses aligned with user-specific dialogue preferences [2] [3]. Such strategies often fall short of the efficacy achieved through fine-tuning methods and are sensitive to the precise structure of the prompts, including the choice of words and their arrangement [4].

A recent paper called "LaMP" [5] suggest an alternative method to personalize LLMs that involves embedding user profiles in the initial training phase, which equips LLMs with knowledge specific to users. However, the development of entirely personalized LLMs for each individual might not be financially feasible yet due to the resources required for training these models.

All these proposals require on the one hand data related to the context or the user himself, and on the other hand an adaptation or finetuning to each specific case, which results in a problem when expanding its use and a significant expenditure of resources.

1.2 Innovation

In this project we propose to solve this problem of lack of closeness and personalization by LLMs by providing them with a long-term memory where they can store the user's personal data.

In this way, when the developed chatbot has a conversation with the user, it will have access to a series of personal data about the user that it can use in the scenarios it deems appropriate.

Thus, by providing the LLM with personal information, the user will perceive a greater sense of closeness and interest on the part of the chatbot.

This memory system will be developed so that it can be freely used around any available conversational LLM. It will consist of a database where the user's personal memorable data will be stored and two key processes in the system: the memorization process and the recall process.

The memorization process will be responsible for extracting from the conversation with the user the personal memorable data and storing them, together with several associated attributes, in the database. The recall process will be in charge of selecting the data from the database most relevant to the current state of the conversation in order to generate an appropriate response, taking into account the attributes of the stored memorable data and the current context of the conversation.

For the process of memorization, we propose to use a new technique that has not been previously tested in the state of the art: using an LLM for the extraction of personal memorable data said during a conversation. Therefore, this work involves the validation of this technique which includes the generation of conversational data, design and execution of a human test, design and execution of a test for the LLMs, validation of the data, design and validation of a score formula attending to the nature of the memories and more.

1.3 Objectives

The main objectives to be completed during the course of this project are listed below.

1. State-of-the-art research on the development and experimentation of artificial memory systems chatbots.
 - (a) Research on the functioning of human memory.
 - (b) Research on LLMs with a focus on memory-related developments.
2. Long-term memory system architecture design.
3. Validation of the designed memorization process.
 - (a) Design, development and validation of a test with human annotators and LLMs on the task of extracting memorable data from a conversation.
 - (b) Design, development and validation of a specific score formula for the evaluation and comparison of LLMs.
 - (c) Analysis and conclusion of the results obtained.
4. Development of a chatbot system with long-term memory.
 - (a) Design and implementation of the final architecture.
 - (b) Design and development of the overall data flow, the memorization process and the recall process.
 - (c) Execution and demonstration of the final system.

1.4 Document Structure

The development of this paper begins with the investigation of the state of the art on issues related to memory and the technologies and systems that seek to improve it in LLMs. This comprises research on LLMs, research on the functioning of human memory, research on innovative systems and technologies on memory enhancement in LLMs and some extra aspects.

After that the development starts with the initial design of the chatbot system architecture. Then we continue with the validation of this architecture and more specifically, with the validation of the memorization process. This validation includes conversational data

generation, design and development of a human and a LLMs test, validation of the data obtained and development, participation of more than 20 volunteers and validation of a score formula to evaluate and compare the 4 LLMs selected in this specific scenario. In the following section, we proceed to analyze and draw conclusions from the results obtained in the previous validation.

Subsequently, we start with the design and development of the final system. First we start with the implementation of the designed architecture, then we develop both the main process of the system and the two key processes of the chatbot: the memorization process and the recall process. To finish the development, a demo of the final developed system is made, analyzing its advantages and added value. Finally, conclusions are drawn about the whole development of the project that has been carried out.

1.5 Conclusions Insights

After the generation of the data, more than 20 volunteers for the human participation, design and execution of the tests and score formulas and analysis of the results, it has been possible to validate the use of an innovative memorization process based on the use of LLMs to extract information by a chatbot. It has also been proved that LLMs are not good at estimating the duration of memorable data and this is an area that requires further research.

The final architecture has been designed and a long term memory chatbot system has been developed with its processes involving memorization and recall previously mentioned. A demonstration of the chatbot's operation has been performed, showing how the chatbot shows interest and closeness to the user by memorizing and recalling personal information about the user at the indicated times.

2 State of Art

In order to develop the present project, the state of the art surrounding both LLMs and human memory has been investigated.

First of all we have investigated the operation and technology of the Large Language Models, as well as the operation of these and the most interesting models that exist at the time of developing this document. Also the most common techniques to improve their performance and the near future of this technology.

After this, the functioning of the human memory has been studied with the aim of taking inspiration from it for the long-term memory system for a chatbot. This study includes both types of memory systems and types of memory from a functional point of view. Thus, also interesting applications of the theory in a practical environment.

We have also investigated the most current techniques and works that are related to this project in order to take inspiration from them. Specifically, we have investigated papers related to the optimization of memory usage, the increase of the context window of LLMs and papers that seek to implement an increased memory in LLMs involving or not involving training.

We have investigated different suites designed to develop systems where different LLMs interact with each other and also with external elements such as APIs and databases. Finally, we have also investigated the different existing databases to find the most suitable for this project.

2.1 Large Language Models

In this section, we will investigate Large Language Models. First, we will introduce how they work and some key aspects of their use. Then we will analyze the main architecture behind all these LLMs, the "Transformers" architecture.

Then we will investigate and study one of the most important techniques around the use of LLMs to improve their performance and obtain better results without changing their internal structure, only modifying the prompt used to interact with the model.

After this, we will investigate new architectures that try to improve the performance of transformers and may be interesting in the future. Finally, we will also analyze current LLMs with larger context window and, therefore, larger working memory according to the relationship established in the section on human memory (see Section 2.2.7).

2.1.1 Overview LLMs

Large Language Models (LLMs) are artificial intelligence systems within deep learning designed to understand and generate text inspired in the mechanisms present in the human brain. These models are trained on large volumes of textual data, enabling them to capture patterns, linguistic structures, and complex semantic contexts.

LLMs can perform a wide range of tasks, from answering questions and translating languages to writing creative texts and summarizing information. Thanks to their ability to process and generate text with high accuracy and consistency, LLMs are revolutionizing a variety of industries, from customer service to scientific research.

These models were originally thought of as a text-to-text model, meaning that they accept text input and generate output in text format. This has changed in recent years, accepting different types of input information and generating different types of output. However, in this project we will focus on text-to-text oriented LLMs.

One of their most relevant features in comparison with previously existing text generation models is their ability to generalize and execute "zero-shot" tasks. That is, these LLMs have the ability to perform tasks for which no information was available in their training corpus and therefore have not been trained on these tasks [6].

Functioning. The operation of LLMs is based on the use of deep neural networks, especially Transformers architectures. These neural networks consist of multiple layers that process and transform input data (in this case, text) into meaningful outputs.

Transformers, in particular, use an attention mechanism that allows the model to focus on different parts of the input text dynamically, improving its ability to understand context and word relationships. During training, the models adjust their internal parameters to minimize error in predictions, thus learning to generate accurate and contextually appropriate responses.

The way in which we can interact with these LLMs are called prompts. These prompts represent the model input data in a textual format. The prompt is where the task to be performed by the model must be specified, as well as the data necessary for it to perform this task.

The amount of data that can be passed to an LLM as input is defined by the LLM's context window. The context window is constituted as all the data that is passed to an LLM as input in each LLM inference and has a limited maximum size that varies with each type of LLM.

In order for these LLMs to perform their task efficiently, it is necessary to transform the input data into a format that this model is able to understand. This is done using so-called "tokenizers", which are also useful to represent in natural language the output of the LLMs when they have the function of generating text.

Tokenizer. The tokenizer in Large Scale Language Models (LLMs) plays a fundamental role within the operation of LLMs. Initially, the tokenizer separates the input text into manageable units called tokens, which can be whole words, subwords, or individual characters, depending on the type of tokenizer used (e.g., BPE [7], WordPiece [8], or unigram). Then, each token is mapped to a unique numeric identifier, using a predefined vocabulary that the model has learned during its training.

Once the tokens have been converted into IDs, they are transformed into tensors, which are multidimensional arrays of numerical data. These tensors represent the text in a form that neural networks can process. In short, the tokenizer not only partitions the text, but also translates these partitions into a numerical form suitable for processing in LLMs, enabling the model to apply its deep learning capabilities to understand and generate natural language [9].

2.1.2 Transformers.

All of today's most capable and advanced LLMs are based on the Transformers architecture. This fact is not a mere coincidence, but shows how Transformers is currently the most suitable deep learning architecture for large language models.

History. The Transformers architecture was presented in 2017 by Google in the highly acclaimed paper "Attention is all you need" [10]. Prior to the publication of this paper, the vast majority of deep learning models were based on Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) and Convolutional Networks (CNNs).

When this paper was published and models using this new architecture started to be developed, it was observed how these new models based on Transformers obtained much better results, especially in tasks related to natural language processing, so their use began to be extended and generalized to other tasks.

Some of the key aspects of this performance improvement is the elimination of recursion compared to some of the previous architectures and the easy scalability and efficiency to build much larger models with more parameters that achieve better results than smaller models.

But the most important factor introduced by this new architecture is attention. This paper introduced a new system that allows the model to evaluate the relevance of different parts of the input text simultaneously and more efficiently. Instead of processing sequentially as in recurrent models (RNNs), attention allows the model to dynamically focus on different words, capturing long-term relationships and dependencies more effectively. This new mechanism is called attention and is the key to the improved results with this new architecture.

Architecture. Transformers consist of two main components: the encoder and the decoder (see Figure 1). Both components are made up of multiple identical layers stacked on top of each other. The encoder processes the input sequence to create an internal representation, while the decoder uses this representation to generate the output sequence. Unlike recurrent neural networks, Transformers process sequences in parallel, enhancing efficiency and the ability to handle long sequences [10].

The encoder (left part in the Figure 1) in Transformers comprises several identical layers, each containing two main sub-layers: a self-attention sub-layer and a feed-forward neural network sub-layer. The self-attention mechanism allows each position in the input sequence to attend to all other positions, capturing long-term dependencies within the sequence. Following the self-attention, the data is passed through a feed-forward network that applies additional transformations. Residual connections and layer normalization surround each sub-layer, helping to stabilize and accelerate the training process.

The decoder (right part in the Figure 1) also consists of multiple identical layers, each featuring three sub-layers: a self-attention sub-layer, an encoder-decoder attention sub-layer, and a feed-forward neural network sub-layer. The self-attention sub-layer in the decoder allows each position in the output sequence to attend to previous positions, essential for sequential text generation. The encoder-decoder attention sub-layer enables the decoder to focus on the encoder's output, leveraging the input sequence's representation to generate the output. Similar to the encoder, residual connections and layer normalization are present in each sub-layer to improve stability and training efficiency.

Uses. The first uses given to this architecture were oriented to natural processing language. That is, any task related to text processing. Among these, the generation of summaries, the detection of sentiments in sentences, text translations, detection of topicality in text, etc. stand out. Some widely used models for these tasks are the BERT[11] and the T5[12], as well as all their variations.

Parallel to these developments aimed at creating a model that would perform a specific task on text, other branches were developed using this architecture. The first was the beginning of the development of LLMs, large models developed by taking advantage of the easy scalability and efficiency of transformers.

The goal with these LLMs was to be able to develop large models that were not oriented to a single specific task, but could perform many tasks and even generalize to tasks on which they had not been trained. To this end, large datasets have been used to cover all the tasks to be performed and text generation in general for language understanding. An example of the first LLM to achieve this was GPT-2[13] and XLNet[14].

On the other hand, this architecture has also been tested and used in other sectors that are not directly related to the processing or generation of text, such as images, videos, and audio. In these tasks, it has been discovered that the Transformers also have great benefits thanks to

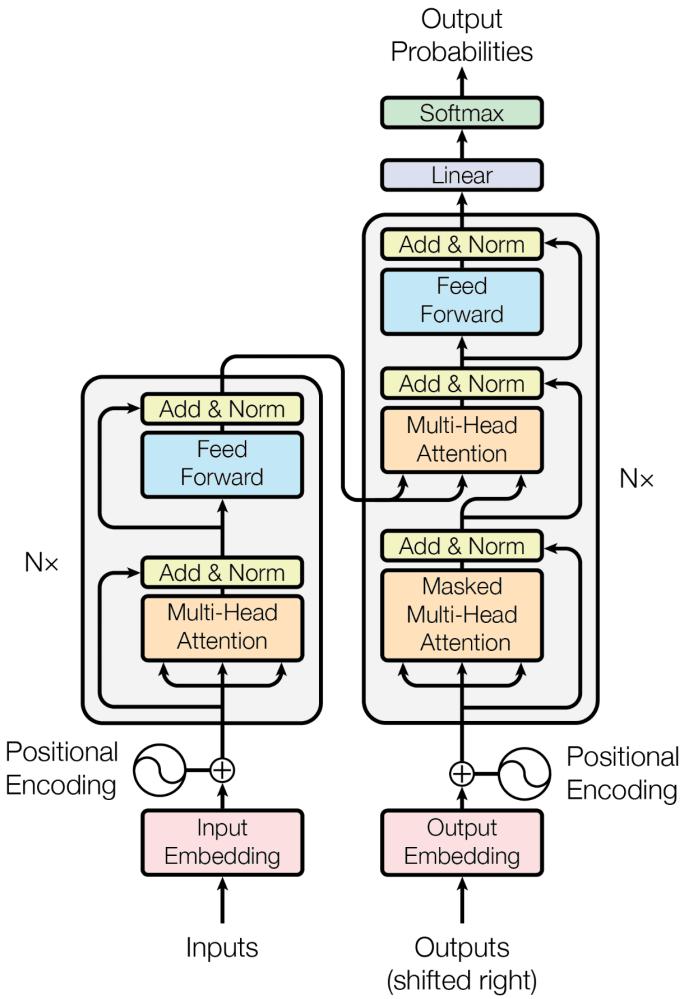


Figure 1: Transformer Architecture

the attention mechanisms and the ability to process many data in parallel. Examples include the Transfomer-Traducer for audio[15] and the Vision Transformer for images[16].

2.1.3 Prompt Engineering

As discussed in the section associated with the operation of Large Language Models (see Section 2.1.1), the way in which we can interact with these models is through the use of prompts. These prompts include both the information about the task to be carried out by the LLM and the data necessary to carry it out.

Several studies on recent large language models affirm that the design of the prompt is a key aspect for the correct operation of this and that the variation of the results when modifying the structure and expression of the prompt is very significant [17] [18]. Some even name it as one of the most significant challenges in relation to obtaining the best possible results, which can result in dramatic output changes [19].

For this reason a new stream of experimentation has emerged called "Prompt Engineering" which is based on designing natural language queries that steer the model's outputs toward desired outcomes [20] [21]. This new practice seeks to investigate how LLMs understand

the prompt through experimentation by modifying the prompt in various ways and observing the output at each iteration.

All this study has led to the definition of different tactics to attack the problem of low prompt efficiency. Most of these focus on explaining well and in detail the context, the function of the LLM, the task to be carried out and the data to be passed to it.

There are also ideas that differ from the previous ones and seek to improve performance from other more elaborate points of view. The following are two of the most relevant in the current state of LLMs.

Few Shot Learning. This prompt engineering technique aims to make the model learn to perform a task for which it has not received any examples during the training of this model. For this purpose, a series of examples of the performance of the desired task with the ideal result are introduced at the prompt.

In this way, the LLM can get an idea of how the desired task is to be performed and mimic this behavior with the rest of the data. This technique contrasts with "zero shot learning" where the LLM is expected to perform a task without the need to provide previous examples of how to perform the task [22].

Chain-of-thought. The Chain-of-Thought (CoT) technique involves structuring the prompt so that the language model generates a step-by-step response, breaking down the reasoning process into a logical sequence. This helps the model tackle complex problems by breaking them down into more manageable and explicit sub-tasks.

Instead of asking for a straightforward answer to a difficult question, the model is guided through a series of intermediate steps, improving the consistency and accuracy of the final answer. For example, to solve a mathematical problem, the prompt might first ask to identify the relevant data, then set up the necessary formulas, and finally calculate the solution [23].

Reasoning. These techniques will be very useful when developing the system and seeking to take full advantage of the capabilities of the LLMs used. In each aspect where an LLM is used to accomplish a task, the design of the prompt will be specified using some of these techniques and guidelines.

2.1.4 New Models Architectures

All of the most famous LLMs today are based on the Transformers architecture. This architecture was published in 2017 [10], so it is coherent to think that in recent years, with the advance in technology and the widespread use of transformers, there have been attempts to develop an architecture superior to this one.

Although there is still no LLM with this new architecture that matches the models based on Transformers and it is not within the scope of the project to develop a new LLM from scratch, it is good to take some of them into account since they can provide interesting ideas and even represent the future of LLMs.

Retentive Networks. This new architecture called "Retentive Networks (RetNet)" [24] aims to achieve low-cost inference, efficient long-sequence modelling, Transformer-comparable performance, and parallel model training simultaneously. The key to this improvement is based on the replacement of the famous multi-head attention of Transformer [10] by a multi-scale retention mechanism which has three computation paradigms: parallel, recurrent, and chunkwise recurrent representations.

According to the authors, these three computation paradigms allow the model to optimise GPU usage, achieve O(1) efficiency in terms of memory and computation in inference compared to the current O(n) base efficiency of transformers, and perform efficient long-sequence modelling.

Mamba. This new architecture called "Mamba" [25] is presented as a new class of selective state space models. It takes inspiration from the classical state space models [26] but improves on the processing of information-dense data such as text to achieve the modeling power of Transformers while scaling linearly in sequence length. This improvement, focused mainly on the case of linear scaling of the context window, is based on two key changes: the selection mechanism and hardware-aware algorithm.

The paper proposes a simple selection mechanism by parameterizing the SSM (state space model) parameters based on the input. This allows the model to filter out irrelevant information and remember relevant information indefinitely. Also, a hardware-aware algorithm that computes the model recurrently with a scan instead of convolution, but does not materialize the expanded state in order to avoid IO access between different levels of the GPU memory hierarchy. The resulting implementation is faster than previous methods, scaling linearly in sequence length.

Reasoning. Both proposed architectures should be taken into account as they present interesting ideas especially in the aspect of memory usage and the scaling in sequence length (related to context window), which is interesting for this project. But until researchers develop large models based on these architectures no concise conclusions can be drawn. All the results that are currently available on these architectures come from the paper where they have been presented, so there is a lack of support from the community to evaluate these architectures.

2.1.5 LLMs with large context windows

The advancement of technology both on the software side with more optimized memory and window management techniques (as some explained in Section 2.3.1) and on the hardware side with increasingly more powerful hardware both in terms of optimization to LLM-related tasks and raw power and less consumption is leading to an exponential increase in the capabilities of the models in recent months.

Some of these advances focus on the increase of the context window up to figures that allow processing an entire book at the time of inference. The following are some of the most relevant examples in this regard.

Claude 3. Recently the Anthropic team has released a series of models with different sizes that fall into the Claude 3 group, the successor to Claude 2.1. Although this model evolution does not imply an increase in the context window, both Claude 3 and Claude 2.1 are still some of the models with the largest context window today.

Both of these groups of LLMs have a context window of 200 thousand tokens, which allows a large amount of information to be accepted at the prompt. At the unveiling of this new Claude 3 model, the developers promised that these models are capable of achieving a context window of 1 million tokens, but as of the date of this research no public experiments on the use of this 1 million token context window have yet been released [27].

Gemini. Gemini is a family of LLMs from Google with different types of capabilities and pricing depending on the required use. Thus, in February of this year Google announced the new "Gemini 1.5 Pro" model and its clear intentions to focus on increasing the context window as a major asset [28].

While the original "Gemini 1.0" models were well received, they did not improve on the capabilities of the existing models that had been around for some time (largely referring to the OpenAI models). Thus, shortly after the release of the "Gemini 1.0" models, Google introduced a new model called "Gemini 1.5 Pro" to try to outshine its competition with the main feature that this model supports a context window of 1 million tokens, the previous model with the largest context window being the Claude 2.1 and Claude 3 model with a context window of 200 thousand tokens.

Although this presentation caused a lot of commotion due to the great leap in size of the context window, as of writing this report, there are still many unknowns in the air since only a proof of concept has been shown in a controlled environment by Google and few other developers. Thus, in the paper where Google presents the results of the various tests and benchmarks that has passed this model, it does not give more information about its architecture or explain how it has achieved a context window of 1 million tokens and the resources required to run it.

Google Deepmind's own CEO, Demis Hassabis, commented in a recent podcast called "Hard Fork" by the New York Times that the DeepMind team was internally testing models with a context window of 10 million tokens or more, achieving very promising results but that due to costs and resources these could not be released to the public [29].

This gives us to indicate that, despite the optimization by the Google team that the increase of the context window does not lead to a huge increase of the necessary resources, there is still a scaling in the necessary resources to be able to use this model. This is why, although techniques for optimizing the increase in the size of the context window are advancing, the premise of the present project is still attractive to implement an external memory with a selection of memorable data to these conversational LLMs without incurring an increase in the context window, and thus saving on costs, resources, and carbon footprint.

2.2 Human Memory

Human memory is presented as the most advanced memory system that exists today. Therefore, this research aims to better understand how it works and to try to take inspiration from some of the procedures on which human memory is based in order to assess whether they can be applied to the long-term memory system around an LLM.

The paper "Towards learning 'self' and emotional knowledge in social and cultural human-agent interactions" argues the following: "If we want to create more pleasant and acceptable agents, capable of carrying on with more engaging interactions and consequently maintaining the relationship for a longer period of time, we need to consider the memory aspect and its sharing and social function" [30].

2.2.1 Introduction

Melton (1963) [31] suggested that the functioning of human memory depended on at least three phases: encoding, storage, and retrieval. Encoding refers to the processing of incoming information so that it can be stored. The encoding process includes, in turn, two separate processes: acquisition and consolidation. Acquisition involves recording inputs from sensory memories and subjecting that information to different analyses; consolidation, on the other hand, consists of a process of creating increasingly strong representations or "memory traces".

The result of acquisition and consolidation is storage, which involves maintaining or retaining over time a permanent record of such traces. The term "storage" does not really describe any process, but simply refers to the state of information that has been encoded. Finally, retrieval is the process of extracting stored information for use in the present moment [32].

This functioning of the human memory is easily extrapolated to the idea of creating the system of this project. The encoding, storage and retrieval phases are directly related to the basic operations for the management of a database. This is why we had the idea of, in order to create a bot that has the ability to remember, we first looked at the most advanced information system in the world: the human memory.

2.2.2 Memory Systems

The idea that human memory is composed of distinct systems has a long history [33]. Some of the first writings about this topic are dated more than a century ago. For example, Maine de Biran wrote in 1804 about mechanical, sensitive and representative memory [34] and it was not until the 1923 that the distinction between explicit and implicit memory appeared by McDougall. In the 1970s, in one of the most important leaps in human memory research, we began to move from philosophical discourse to experimentation to really know how the brain's memory functioned intrinsically.

For some years, experimental studies were conducted on people with amnesia. A milestone in the history of understanding the functioning of human memory occurred when an animal model of human amnesia in the nonhuman primate was obtained [35]. Since that time work with experimental animals has been invaluable for understanding the memory systems of the brain. Thanks to these experiments, progress began to be made in the field, as for example with the idea that the hippocampus and related structures were involved in one particular type of memory. [36].

In the following years, interesting data were discovered, for example that priming (prior exposure to a stimulus influences response to a subsequent stimulus without conscious intention) was intact in amnesia [37] and that motor skills are a subset of a larger domain of skill-like abilities [38].

Realizing that the types of memories that were not affected by amnesia were implicit (non-declarative), they realized the need for further and specific differentiation of the memories. Thus, with the publication in 1994 of the article "The Memory Systems" [39], a diagram was established that is still present today [2]. In this diagram, we can see how the first differentiation between memory systems is based on the classification into two types: declarative (explicit) or non-declarative (implicit).

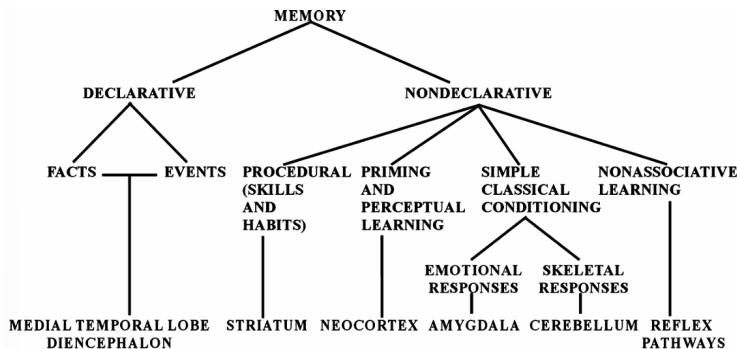


Figure 2: A taxonomy of mammalian long-term memory systems

2.2.3 Explicit and Implicit Memory

Explicit and implicit memory differ mainly in the awareness of the memory. Explicit memory involves the conscious recall of information, facts and events. On the other hand, implicit memory refers to learning and experiences that affect behavior or thinking without our being consciously aware of those memories. Examples of implicit memory include motor and cognitive skills such as riding a bike, or the effects of conditioning such as priming discussed above. While explicit memory can be communicated verbally and requires conscious effort to retrieve, implicit memory is manifested through performance and does not require consciousness for retrieval.

Returning to the goal of the project, development of a memory architecture around an LLM in such a way that it offers a sense of closeness, we see how the abstraction of explicit memory is the one that most closely aligns with this goal. In order for the bot to simulate showing more interest towards the user and for the user to feel valued, it is convenient to remember things about the user, which falls directly into explicit memory.

Some articles support this thought. In the article "I've been here before" a non-interactive experiment was conducted with 96 participants to test the hypothesis "Autonomous agents with episodic memory retrieval of emotionally relevant events, will be perceived as more believable, than similar agents without it. [40]". In another paper called "MAY: My memories are yours" an experiment was conducted with MAY (an agent based on an engine that uses AIML (Artificial Intelligence Markup Language) to create responses to questions and inputs) to measure the quality of the friendship between the user and the agent and found that when memories about the user were exhibited the scores for intimacy and companionship were higher [41].

2.2.4 Declarative or Explicit Memory

The current state of knowledge of the major memory systems is based on five distinct systems: Procedural Memory, Perceptual Representation System (PRS), Working Memory, Semantic Memory, and Episodic Memory. Of these five types of memory, the first two are associated with implicit memory, which is outside the scope of this TFM as discussed above. Working memory is considered an operational definition of STM (short-term memory), being this system the one that allows to keep activated and accessible a limited amount of information during a short period of time while mentally manipulating this and other information. Thus, because of the previous association we made of the STM to the context window itself, we will not investigate it further for the time being.

Focusing on the long-term explicit memory system, we have the semantic and episodic memory systems. Going back to 1972, Endel Tulving, an Estonian-Canadian psychologist and one of the fathers of current human memory theory, published an article called "Episodic and Semantic Memory" [42]. He established the first division of memory systems within the subset of declarative or explicit memory, which in future articles would be referred to as long-term explicit memory.

He established the subdivision between episodic (events) or semantic (facts) memory. In his article, he defined that semantic memory refers to the general knowledge of the world without being linked to a specific resource, and that episodic memory refers to the capacity to remember specific events of our lives, including the temporal and spatial connection of these events. From a more anthropological point of view, episodic memory requires the participation of brain systems in addition to those that support semantic memory, for example, the frontal lobes [43].

Semantic Memory. Semantic memory is the system responsible for the acquisition, retention and utilization of knowledge about the world in the broadest sense, i.e. facts, ideas, beliefs, concepts and vocabulary; knowledge detached from the spatial and temporal circumstances of its acquisition [42]. Semantic representations constitute the general knowledge of individuals about the world. Thanks to this system, people can represent states, objects and relationships between them without the need for them to be physically present. Tulving points out that the main function of the structured representation of this semantic knowledge is the cognitive modeling of the world [44].

Episodic Memory. Episodic memory is the memory for personally experienced events. Thanks to this system we can retrieve, in a deliberate and conscious way, the experiences of our personal past that occurred at a specific time and place, that is why it is also called "personal memory" or "autobiographical memory" [32]. Tulving proclaimed that episodic memory refers to a system that makes possible mental "time travel" through subjective

time: from the present to the past and into the future, a feat that no other memory system can perform. In doing so, it allows the individual to re-experience, through autonoetic consciousness, experiences already lived, and to project similar experiences into the future [45].

Relationship between Episodic Memory and Semantic Memory. Although the two memories evoke and refer to different types of thinking, the relationship between them (both their distinction and similarity) is still a matter of debate. Tulving and Markowitsch in 1998 proposed the properties common to both systems that they considered: both memories are long-term systems, both receive information from sensory systems, the encoding of the memories of both is very similar, both operate independently of manifest behavior, both handle propositional information, both are context-sensitive, and both can be used to draw inferences [46]. All this means that these two systems are often considered as a unit and labeled together as declarative memory ("know what") versus procedural memory ("know how").

However, episodic memory has evolutionarily transcended semantic memory. Tulving and Wheeler's hypothesis is that these two memory systems differ essentially with respect to the nature of the conscious experience that accompanies their respective information retrieval processes. For the episodic memory, it is this conscious experience that allows us to relive an experience with a specific time and place context into the past with an act of consciousness, which they call "autonoetic consciousness", when all other memory systems are oriented to the present [45].

Reasoning. After investigating and understanding better these two human memory systems that may be the most related to our project, we have to reason if this distinction is effective for our scenario. The idea of the system to be developed is to equip the bot (LLM) with a long term memory where it stores relevant and memorable data of personal character of the user with whom it is talking to have these available and simulate a closeness or interest towards the user.

It is key to keep in mind that this memory to be developed is not related to the point of view of the bot and the experiences or memories that it may have about the conversation. The bot will store in long term memory only information that the user tells it, so this is directly encapsulated in the semantic memory system.

For the episodic memory system to make sense in this implementation, it would be necessary to treat the bot as an entity of its own that can have past experiences and sufficient "autonoetic consciousness" to be able to make these experiences conscious in order to relive them when recalling them. This capability has not yet been demonstrated by LLMs, so we cannot treat it as a reality.

On the other hand, when a user relives and recounts an experience from episodic memory to another person, from the other person's point of view this information is stored in the semantic memory since this second person has not lived the experience itself and only has the information as a piece of information about the other person. Therefore, from the point of view of the bot, the information from the user's episodic memory is also recorded as normal memorable data in the semantic memory system.

2.2.5 Functional memory

Historically, experts in psychology have worked to decompose memory into different categories, forms or systems in order to better understand its functioning. As we have already seen, there is a distinction of memory into different systems depending on their nature, but there is also a distinction of memory from a functional point of view.

One of the first endorsed distinctions was made in 1890, where William James, in his influential work "Principles of Psychology", already suggested differentiating between what

he called "immediate memory" and "remote memory" [47]. This concept was taken up by Donald Hebb in 1949, who introduced the need to separate "short-term memory" (STM) from "long-term memory" (LTM) in his theory known as the "two-trace theory" [48].

This approach reached its apogee in 1968 with Atkinson and Shiffrin's "multiple storage model" theory, which postulates that memory is composed of three stages or stores: the sensory register, the short-term store, and the long-term store, arranged in a sequence through which information transits [49].

This model of Atkinson and Shiffrin is considered the most influential "structural memory" proposal in history, being considered as the inescapable theoretical and working frame of reference in memory research. Taking this model as a starting point and making the basic corrections suggested by the most recent models, the functioning of memory can be adjusted to the three categories shown below: sensory memory, shot-term memory and long-term memory [32].

Sensory Memory. Environmental information is captured by sensory memories specific to each sensory modality and is immediately registered. Only a part of this information that manages to capture the user's attention, either voluntarily or involuntarily, and that finds some association with the long-term memory (LTM), is further processed and transferred to the short-term memory (STM), while unselected (unattended) information is quickly lost through a decay process. Sensory records are characterized by being specific to the sensory modality, having an almost unlimited capacity and a very short retention time, varying between 0.5 and 4-5 seconds.

Short-Term Memory or Working Memory. Once the sensory information is in the STM, it will receive a lot of attention (makes it conscious) and can be recognized and categorized. The process of recognition and categorization occurs in the STM because it is the place where the user's memories (coming from the LTM) intersect with the sensory inputs. This memory (STM) is always active and accessible for immediate retrieval and use, which is why it is also called "working memory (WM)" [50], also defined as the memory with which the subject works or performs conscious mental operations. Examples dependent on the work of the MCP could be holding a telephone number in the mind while dialing, mental calculation, reasoning tasks and general thought processes.

Two of the most relevant characteristics of STM are its time limit and its capacity limit. Since the work of Peterson and Peterson in 1959, it has been argued that the maximum retention time of information in STM is between 15 and 30 seconds [51]. As for the capacity, in 1956 Miller established that it was «7 +- 2» items, although more recent work, such as that of Cowan in 2001, maintains that in adults it is around «4 +- 1» items and that it is strongly related to the focus of attention.

Long-Term Memory. As soon as the information receives attention it becomes conscious, which implies, in the opinion of some researchers, that this information is transferred and encoded in the LTM [52]. Atkinson and Shiffrin proposed that information entering the STM will disappear by a displacement process unless the subject undergoes a review process that involves allocating attention and mentally repeating the information, and that this review process is necessary for the memory to be transferred to the LTM.

Any memory that enters the LTM is considered to be permanently stored, even though, although all of it will always be available, at certain times only part of it will be accessible. The LTM is not a single system, as traditionally considered, but a set of independent and interacting systems. There is now abundant experimental evidence supporting the existence of at least four major long-term memory systems in the brain: the procedural memory system, the perceptual representation system, the semantic memory system, and the episodic memory system.

Reasoning. In the context of this project, the idea is to take inspiration from part of this operation so that the bot is able to manage a permanent memory. The STM or WM can be related to the context window of the models themselves, since this information has a limited capacity and is information that is present at all times and immediate on which the bot generates the response. On the other hand, the implementation of an external memory would be similar to the LTM, since it would have a virtually infinite capacity and from which the appropriate memories would be extracted depending on the context and situation of the conversation.

The passage of information from short-term memory to long-term memory is what we consider the process of memorizing. In human memory this is done when the user pays attention to specific information and makes it conscious. This process gives us inspiration to look for ways to apply it to our scenario and somehow simulate the ability to pay attention to certain information by a human in our system.

2.2.6 Memory theory applied to chatbots.

There are some examples that have already tried to apply these concepts of human memory to chatbots from the point of view of working memory and from the point of view of memory systems. One of the most interesting examples is shown below.

MAY: My Memories Are Yours. "MAY" [41] is defined as a dialog system based on AIML (Artificial Intelligence Markup Language) to create responses to questions and inputs, being this system a much more archaic one compared to current LLMs. This bot tries to establish a relationship with the user following the premise that in human relations, engagement and continuous communication is promoted by the process of sharing experiences. They also rely on Sudan Bluck's statement in his article on autobiographical (episodic) memory: "It is this process of sharing personal memories, available by autobiographical remembering, that enriches our conversations, making them seem more truthful, believable and engaging" [53].

This is why the memory structure they propose is based on an autobiographical memory architecture suggested by Conway [54]. This architecture divides the memory system into 3 levels: lifetime periods, general events, and event-specific knowledge. Depending on the duration of the event itself, they are assigned to one level or another. If the event has a duration of years, it corresponds to lifetime periods; if it has a duration of days or months, it corresponds to general events; and if it has a duration of seconds or hours, it corresponds to event-specific knowledge. This subdivision of memories according to their temporal extension allows them to make a hierarchical grouping by establishing a timeline where different memories can overlap and therefore be related. For example, a lifetime period memory such as "when I went to school I had long hair" can be related to an eventual general level memory such as "when we went on a 5th grade field trip we went to the beach".

According to the paper, the results showed that when the agent manifests acquaintance about the user's life, even little details of common events, the users classified more positively the intimacy and companionship dimensions of friendship.

This paper[41] presents good ideas such as the intuition that storing information from the user's episodic memory and then retrieving it to generate responses from the bot gives good results. Thus, the creation of a timeline in our project is not considered feasible. It could be implemented for events that take place in the present and in the future, but when a user talks about events in the past, in real conversations, it does not give so much temporal information as to be able to create such a timeline. The good results of this paper with this timeline come from its tests in optimal conditions where the user does indicate well the temporal period of the events.

2.2.7 Inspiration for the memory architecture

To address the task of providing the chatbot with a sense of closeness or well-being towards the user, there are many different approaches to work on. Leaving aside the LLM's own ability to generate the actual conversation with the user, other approaches may include adapting the bot's personality to one that is more user-friendly or that the bot is able to manage relevant information about the user so that the user feels listened to and appreciated. In this project, we will focus on the last of these approaches, creating a system that allows the chatbot to simulate a memory system that allows it to give that feeling of paying attention and showing interest towards the user by storing personal information.

Memory Systems. Focusing first on the different systems of human memory, we are presented with 3 memory systems related to this project: operative memory, semantic memory, episodic memory. For the first of these systems, the operative memory, we will make the direct correspondence with the operation of the LLM context window. This is because the operative memory is an operational definition of short-term memory, being the one that keeps the most recent information active and directly accessible, exercising a functionality very similar to the context window which contains the active and present information at the time of LLM inference.

As for the semantic and episodic memory systems, as discussed in the corresponding section, only the semantic memory will be matched with the database where the personal data that the user has told the bot will be stored. This is mainly due to the fact that we only want to store relevant user information and the lack of awareness and experience on the part of the LLM as explained in Section 2.2.4.

Functional Memory. Turning now to the theory and classification of memories from the functional point of view, we find three types: sensory memory, short-term memory and long-term memory. Sensory memory focuses on information acquisition, which we can relate to the input of data to the system. The short-term memory or working memory is directly related to the context window of the LLM, since they both refer to a limited but instantly accessible and active information space.

The long-term memory is related to the database to be developed where all the memorable personal data about the user will be stored. This, unlike the short-term memory or context window, does not have all its information active at the time of generating the response, but needs a recall process to retrieve the relevant information in the context of the conversation 2.2.5.

Memorizing Process. For the process of memorizing information, inspiration is taken from the passage of information from short-term memory to long-term memory in the human mind. This process happens when the user pays attention to some piece of information available in the short-term memory and makes it conscious.

This process has a very biological and subjective character of each person. Looking for ways to apply it to our system, taking into account its limitations, we propose the following question: a current LLM has the necessary capabilities to perform the task of extracting memorable personal data from a conversation with results sufficiently similar to those of a person to validate the use of this? This scientific question will be explored during development to validate if its use is valid for the memory system.

2.3 Towards Longer Memory

Another very important aspect before starting the development of the project is to investigate the techniques and proposals that exist in the current state of the art related to the objective of the project. By analyzing this we can take inspiration from those that are considered more interesting for our scenario and also validate that the work to be done has not been done previously.

Therefore, in this section we will analyze the techniques and systems that currently exist related to memory augmentation and management around LLMs. First we will analyze techniques to optimize the use of memory of the LLMs, second we will analyze techniques that seek to augment the LLM context window itself. Then we will analyze the techniques and systems that seek to directly increase the memory of LLMs from two different points of view: those that involve retraining and those that do not.

2.3.1 Optimizing the memory management

One of the major limiting factors in the capacity of LLMs is memory usage. Much of the model must be active in memory to make use of it, so the larger a model is, the more memory it occupies. On the other hand, focusing on the size of the context window, increasing this window not only increases memory as a function of the size of the input, but also in the attentional mechanisms and other aspects [10].

Therefore, optimizing the memory usage by the LLM can serve to increase the size of the context window for the same memory size, which is an interesting aspect to investigate. In this section, we present two of the most famous and used techniques for the optimization of this memory usage: FlashAttention 1 and FlashAttention 2.

FlashAttention. Before the publication of the "FlashAttention" paper [55], most existing techniques to optimize memory usage without requiring a quadratic increase in resources were based on sparse-approximations [56] [57] low-rank approximation [58] [59] and their combinations [60] [61] [62]. While these achieved the goal of memory optimization, it came at the cost of a drop in the quality of attention results on long input sequences. The researchers in this paper proposed a paradigm shift in the field, instead of focusing on FLOP reduction, to try to optimize the number of times the GPU's HBM (main memory) is written/read in favor of using more of the GPU's SRAM (fast cache) [63] [3], since most of the operations on the Transformers are limited by access to the memory itself [64].

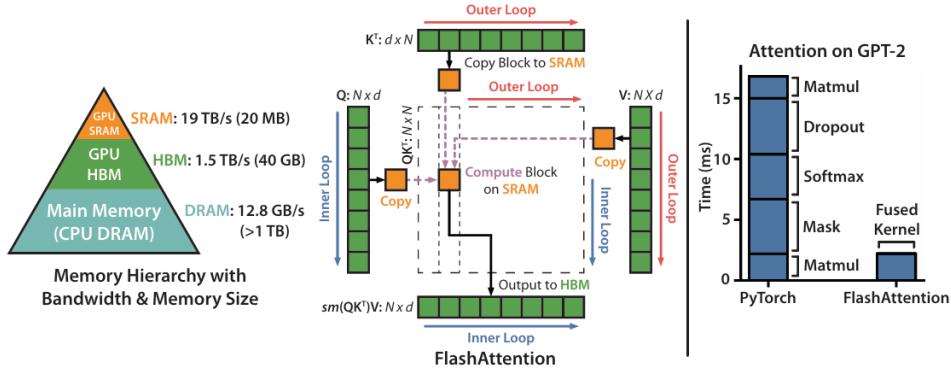


Figure 3: Memory Optimization of FlashAttention

Thus, they propose two techniques to avoid reading and writing the attention matrix as many times as possible. First, by dividing the input into input blocks and making several passes over them in the GPU's SRAM (fast cache) and then writing the output to the GPU's HBM (main memory). Second, storing the softmax normalization factor from the forward pass to quickly recompute attention on-chip in the backward pass, which is faster than reading the intermediate attention matrix.

These techniques allow optimizing the memory management, managing to increase the context window by several factors while maintaining a linear increase in the required resources, as opposed to the quadratic increase required in the initial architecture. Even so, these techniques fall short of the capacity increase required by current models.

FlashAttention 2. The renewal of the paper “FlashAttention” [55] with “FlashAttention 2” [65] comes to make up for some of the original shortcomings, such as the lack of parallelization or the inefficiency in some operations. This new paper introduces techniques for parallelize both the forward pass and backward pass along the sequence length dimension, in addition to the batch and number of heads dimension. Also, within one block of attention computation, partition the work between different warps of a thread block to reduce communication and shared memory reads/writes.

They also implement an algorithm to reduce the number of non-matmul FLOPs. The GPUs are optimized and have special computation units for matrix multiplications (matmul) [63]. Although non-matmul FLOPs are a fraction of matmul FLOPs, eliminating them could achieve up to 16x higher throughput.

Reasoning. As interesting as these two memory optimization techniques (“FlashAttention 1” and “FlashAttention 2”) are, being able to increase the context window in already trained models without increasing the memory required when executing the model, they have a problem to highlight: increasing the context window directly does not mean that the LLM is able to synthesize all the information in the context window. [66]

The foundational models have been trained with a specific context window. Thus, if during inference we make use of memory optimization techniques such as FlashAttention to increase this context window, the LLM may have difficulties and inefficiencies in computing attention in that new context window because it has not been trained with this size of information. These techniques will be a major breakthrough when the foundational models are trained with these techniques, thus being able to increase the window natively by optimizing memory management.

Thus, an intermediate option is to perform the finetuning of a foundational model such as “Llama2” with an augmented context window and this technique. This would allow to increase the context window in a more efficient way since this new model has received a training with the increased window and could have been able to learn how to manage all the attention of the new information.

This can be seen with an example of “HuggingFace” of a model based on “Llama2” of 7 billion parameters with a context window of 32K tokens [67], when the context window of the original model of “Llama2” is 4K [68]. This model was trained using, among other optimization techniques, FlashAttention 2 which helped optimizing the use of memory and therefore, augmenting the context window with the same memory limitation.

The most widespread current use of these memory optimization techniques is focused on training models in a more optimized way in terms of time and memory required. These techniques allow training LLMs on general-purpose graphics cards by reducing the memory required on these cards. In the future, we will see these techniques natively integrated into the new foundational models.

2.3.2 Augmenting Context Window

One of the factors that make LLMs memory limited is the context window itself. As already explained in Section 2.1, the context window represents how much information can be fed into the LLM at the time of inference and increasing it natively quadratically increases both the time and computation required [10]. Since the publication of the paper on transformers and especially since the beginning of the exponential growth in performance of LLMs, we have been looking for techniques to optimize the memory management of these models and thus be able to both reduce the resources needed to train and use these models and to increase the context window of these models without incurring the quadratic increase in resources.

The increase of the context window is directly related to the improvement of the short-term or working memory of these models. It is also worth noting that, without extra memory management mechanisms, the long-term memory would not be improved at all. This is because an increase in the context window allows the model to have direct access to a larger history of the conversation. But without external memory management, it will still not have access to data that falls outside the context window boundary in very long conversations over time and is therefore long-term memory data.

The following papers address both the augmentation of the context window directly (TransformerXL) and the augmentation of the context window in a virtual way with external storage (Memorizing Transfomers and Augmenting Language Models with Long-Term Memory).

Transformer XL. The "TransformerXL" model paper. [69] was published in 2019 when the Transformers architecture was just starting to take off. This was characterized as one of the first Transformer-based models that experimented with the idea of increasing the context window of Transformers, positioning itself as the first self-attention model that achieves substantially better results than RNNs.

The techniques employed to increase the context window are twofold: recursively reusing the hidden states obtained from the attention mechanisms of past iterations, as well as introducing a new relative positional encoding formulation that generalizes to attention lengths longer than the one observed during training, thus allowing the attention mechanisms to attend also to previous information that is outside the attention margin of the context window.

Being able to reuse the last hidden states from past iterations provides a simple and effective way to increase the context window and thus the amount of history-related information present for inference. These ideas laid a foundation for what would be significant improvements to the context window over the next few years.

Memorizing Transfomers. Papers like "Memorizing Transformers" (MemTRM) [70] already consider the possibility of including an external memory on which to store the memories of the conversation history. In this case, the researchers had the idea of implementing a non-differentiable memory on which to store all the data (in "key-value" form) on which later, in each inference scenario, to use a search algorithm to find the memories most similar to the current scenario.

With the idea of storing memories on which to later recall, an idea similar to that of the "Transformer XL" paper is followed [69]. The data that is stored belongs to the key-value pair obtained after a series of embedding layers and attention mechanisms. When doing self-attention at inference of a new input text, the selected keys and values from the memory are prepended to the current keys and values, and a sliding-window causal mask is used so that each token has a local context that includes the previous 512 tokens related with the stored memory.

An algorithm based on "k-nearest-neighbor (kNN) lookup" [71] [72] is responsible for finding the "k" most similar key-value pair entries in the database with respect to the corresponding key-value pair of the input text. This idea allows storing the memories in their complete form without the need to be summarized. The memory is non-differentiable, which allows reuse keys and values that were previously computed on prior training steps, which drastically reduces the amount of computation for large memories.

Augmenting Language Models with Long-Term Memory. The paper "Augmenting Language Models with Long-Term Memory" [73] tries to solve a problem in the paper "Memorizing Transformers" which faces the memory stateless challenge during training. As the model learns and updates its parameters, the older cached memory representations

may not align with the new model's understanding, leading to a mismatch or "distributional shift."

This new model employs a decoupled network architecture with a frozen backbone LLM for memory encoding and an adaptive residual side-network for memory retrieval and fusion. This design prevents the issue of memory staleness by separating the encoding and retrieval processes, allowing for efficient updating and usage of memory without the negative impacts of distributional shifts in the cached representations.

This idea of using a frozen backbone LLM for memory encoding and a residual side-network for retrieval, while keeping the rest of the methodology similar to that of the "Memorizing Transformers" paper, yields significantly better results compared to the aforementioned paper or the vanilla LLMs.

Reasoning. Reviewing the papers mentioned above, in spite of proposing very interesting ideas each one in its historical context, they require a new training of the model to be applied. This creates a strong limitation in their use, since the retraining of the foundational large languages model implies a strong investment in resources and training times that can easily exceed six months as it has been seen in the last "OpenAI" models [74].

The papers [70] [73] where an external memory is used to store past data that do not fit into the windows context and therefore increase it in a viral way present very interesting ideas but lack an efficient management of the memorable data in this external storage in order to be able to use them directly as long-term memory.

The way they select memorable data is based on an algorithm that looks for similarity between key-value pairs after attention mechanisms. This technique is not the most efficient since it only takes into account the similarity with the user's input and not with the context of the current conversation. It also does not take into account other characteristics of the memories such as longevity or relevance of the information.

Furthermore, they also lack the transparency of being able to observe the data in the database as they are represented by the "key-value" of the attention mechanism, which limits the user's handling of this database in the case of wanting to eliminate any unwanted memory.

2.3.3 Longer Memory involving re-training

The idea of achieving memory augmentation (not to be confused at this point in the project with the context window) of models continues to be a hot topic in the LLM landscape. In this research, mechanisms are sought to reuse previous history information in some way, without it falling within the common context window.

Here is an interesting work (Memory-Augmented LLM Personalization with Short- and Long-Term Memory Coordination.) that aims to augment the LLM's own memory but to achieve this goal requires a training focused on each scenario, so it is not considered a valid option but it does contain ideas from which inspiration can be drawn.

Memory-Augmented LLM Personalization with Short- and Long-Term Memory Coordination. This paper [75] attacks the problem of the lack of memory on the part of LLMs from the point of view of their customization. It proposes ideas to elaborate a framework called "MaLP" that allows to customize LLMs for users with a much more superficial training and with a more advanced memory management, establishing several levels of memories that are treated in different ways.

In order to customize an LLM to a user, they use parameter-efficient fine-tuning techniques (PEFT). PEFT focuses on updating a small subset of parameters, ensuring that the trained LLM achieves promising performance on new tasks while minimizing computational costs [76]. Thus, with dialog data from the user or target scenario, an LLM can be fine-tuned in a superficial way so that it adapts.

A clear example they propose is the scenario where the LLM tries to resemble a doctor. With medical dialogues obtained from different sources, a superficial fine-tuning is performed so that the LLM learns to speak like a doctor and have a conversation with a patient.

Memory management is divided into two groups according to their longevity, short-term memory (STM) and long-term memory (LTM). For the differentiation of memories in STM and LTM they take a threshold of repetitions. If a memory belonging to STM is repeated in the conversation above that threshold, it is moved to LTM. To obtain the memories during the conversation, different techniques are used depending on the memory in question.

STM is retrieved in the order in which it is stored, while LTM is retrieved through association. In the case of STM memory it is used the "Levenshtein distance" which indicates the minimum number of deletions, insertions, or substitutions required to transform string "s" into string "t" [77]. For LTM memory it is used the already mentioned in other techniques "cosine similarity" which looks for the semantic similarity between two embeddings [78].

Reasoning. Although this paper presents more attractive techniques and ideas for the objective of this project of a more closeness conversational LLM, it focuses on the customization of LLMs for each of the users or scenarios, requiring a fine-tuning for each of them and previous conversational data that in many cases are nonexistent. The goal in mind is to be able to develop a system that adapts to any user who wants to have a conversation with it without the need of previous data or specific training for each user, so the idea of customization of this paper is not interesting.

As for the management of the memories, it is an improvement compared to previous ideas. Even so, it is a shortcoming of not taking into account more characteristics of the memorable data for a better selection, such as importance or longevity. On the other hand, the passage of information from the short-term memory to the long-term memory is done only taking into account the repetition of the information, so there may be important data that have not been repeated in the conversation and are lost once the limit of the short-term memory of the system is reached.

2.3.4 Longer memory without re-training

As discussed above, training from scratch is not the most suitable solution for the subject matter of the project, as it precludes us from using the best performing foundational models and requires a very high expenditure of training resources. In the case of jobs that require finetuning for customization, these require additional data that is not always available and an associated resource expenditure.

For this reason, we present below some works where they try to increase the memory without the need to retrain the model. These works range from the use of summaries to preserve information, entire memory systems around an LLM and memory architectures designed for scenario simulations.

Recursive Summarization. There are several techniques and papers that explore the possibility of using the technique of summarizing the history of the conversation in order to continue adding it to the context window. These aim to ensure that the model in charge of summarizing is able to synthesize the information well and that the most important points of the conversation are included in the summary.

A recent paper called “Recursively Summarizing Enables Long-Term Dialogue Memory in Large Language Models” [79] explores the summarization technique but in a recursive way. This technique is based on the fact that, in each turn of the conversation, let's call it i , the LLM in charge of summarizing is given the summary of the iteration $i - 1$, the user's utterance in the turn i and the user's answer in the turn i [4].

Thus, the LLM in charge of the summary has knowledge of the past information through the summary of the previous shift and the information of the current shift. This technique

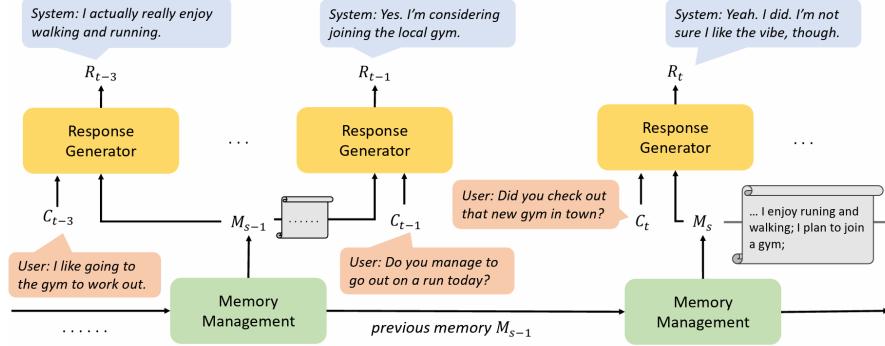


Figure 4: Structure Recursive Summarization

is mainly based on the confidence that the LLM is able to identify the most important information both in the current shift and in the summary of previous summaries recursively.

MemGPT: Towards LLMs as operating systems. The paper of “MemGPT” [80] proposes a series of ideas that are much more attractive to the objective of the present project. This paper attacks the problem of implementing long-term memory by creating a system around an LLM inspired by the memory management of operating systems.

This system proposes the joint use of two memories or contexts to give the sensation of an infinite virtual memory. The “main context” refers to the model’s own context window, information that is directly available to the model during inference. The “external context” refers to the large capacity memory that the model has to request in order to make use of this information.

The main context is divided into three components: system instructions, conversational context, and working context. In the system instructions, the bot is given information on how to operate and handle the conversation; the conversational context contains the recent history of the conversation following a first-in-first-out (FIFO) model; and the working context is a space reserved for the information requested from the external context.

In the “external context”, the bulk of the memory is stored, which corresponds to its long-term memory capabilities. Here, the permanent information is stored both in textual format and in embeddings. This in turn is divided into two memories: “recall storage” where all the conversation event history is stored and “archive storage” which serves as a read-write data store in case the model needs it due to overflow.

The management of the memories is left to the model itself. The system has implemented a series of methods to obtain and store information from the “external context”. These methods (and the operation of the system itself) are explained within the system instructions of the main context so that the LLM is aware of this operation and is itself in charge of this management.

The “MemGPT” system itself works around events that trigger LLM inference. These events can be not only messages from the user in the conversation, but also information from the system itself about aspects such as when the context window is getting full or other types of information about the memory that inform the LLM so that it can optimize its management

[5]

Although this system broadly complies with the premise of implementing long-term memory in conversational LLMs, we believe that it has some points for improvement that will be addressed (among others) in this project (Section 5.1.1). This paper leaves in the hands

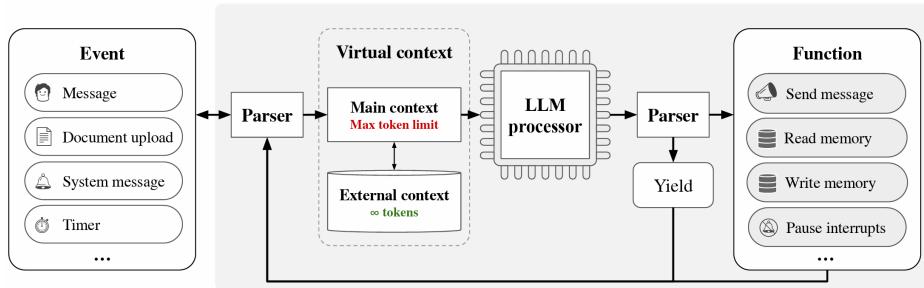


Figure 5: Overall Structure MemGPT

of the LLM all the control of the memory system without previously validating whether it makes good use of it in both the task of memorizing and recalling.

Thus, this system performs the function of searching for memorable data (recall process) solely on the basis of semantic similarity without taking into account other aspects such as longevity or memory relevance. On the other hand, it requires a highly capable LLM that is able to understand the whole system and make use of internal methods to memorize and retrieve data at the right time, thus limiting its use to a few very large and resource-intensive LLMs.

Generative Agents: Interactive Simulacra of Human Behavior. One of the most interesting papers on the simulation of human behavior using LLMs was recently published. This paper [81] published by Stanford seeks to simulate, as closely as possible, the social behavior of humans in a predefined environment. Achieving this goal could bring many benefits in different aspects of social science, such as train people on how to handle rare yet difficult interpersonal situations [82] [83], test social science theories [84] and much more.

To investigate this scenario, the Standford team has developed a complete system that simulates a rural village with diverse inhabitants where each of the inhabitants live their lives independently and interact with each other. In this simulation, each of the inhabitants of the village is represented by its own LLM (in this case, GPT 3.5) with an associated memory system that is intended to provide it with a capacity to remember and act accordingly through the simulation. For this project, we are interested in this memory system.

This memory system is divided into three concepts: memory and retrieval, reflection and planning and reacting. The last one, planning and reacting, will not be taken into account since it does not affect us in the task of developing a chatbot with long term memory, but it is thought to plan the next sequence of actions within the simulation by the agents, acting in consequence with the past events.

The memory and retrieval system takes care of most of the memory management, relying on memorizing and recalling memories. In this implementation, all agent interactions and thoughts are stored as an "observation", leaving the selection logic to the retrieval part. This retrieval function takes the agent's current situation as input and returns a subset of memories to pass on to the language model. This memory selection is based on three components: "recency", "importance" and "relevance".

"Recency" assigns a higher score to memory objects that were recently accessed. "Importance" distinguishes mundane from core memories by assigning a higher score to memory objects that the agent believes to be important, this is done asking directly to the LLM of the agent itself. Relevance assigns a higher score to memory objects that are related to the current situation, this is done computing the cosine similarity between the embedding vector of each memory and query memory's embedding vector. The retrieval function uses

a weighted score formula to score all memories as a weighted combination of the three components mentioned above.

The reflection system is in charge of creating more abstract thoughts about the stored memories. It asks the language model of the agent questions that can cover various types of memories, the retrieve function returns a certain set of memories on the same topic and then the LLM is asked again to extract insights from the subset of memories. This will result in a reflection, that will be stored as a normal memory. [6]

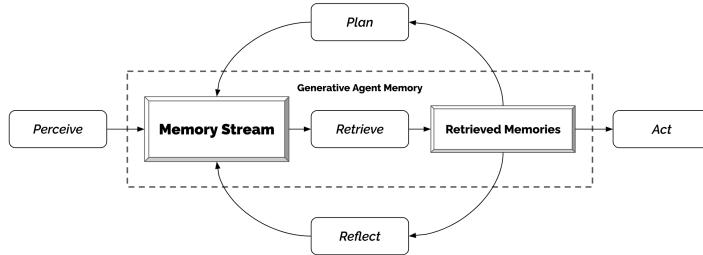


Figure 6: Memory Architecture of Interactive Simulacra

These ideas about the process of recalling memorable data can be very interesting if applied to a conversational LLM system rather than to a simulation of a social environment with interactive agents. During the development, these ideas will be synthesized with others to obtain the best possible results, since this system lacks some features such as a more efficient detection of memories (Section [3.1]).

This memory system stores all the information of the conversation, while by storing only the truly memorable data we manage to reduce the data set on which the retrieve system works. This way we can potentially improve the results and reduce the size of the long-term memory, saving resources. This technique will be explored in detail later (Section [3.2]).

Reasoning. These works are of much more interest for this project since they fulfill the premise of implementing a long-term memory mechanism or system without requiring training or modification in the internal architecture of the LLM, thus allowing to make use of the most advanced LLMs nowadays.

On the part of the iterative summarization technique, there is concern that summaries leave out some of the information and the lack of segregation of the various memorable data for more efficient management.

The MemGPT system looks much more advanced and more attractive for this project, but we do not want to leave the entire operation of the memory system to the conversational LLM since it can misuse the memories. It also requires a very advanced conversational LLM to be able to make use of the internal method call system, thus greatly limiting its use with different LLMs and going against the philosophy of this project. This one being able to make use of this system locally with open source models also executed locally that are not very demanding and can be used in costumer hardware but are not as capable as big models. This small models can not manage alone all the memory system and calls of the MemGPT system.

In the Stanford simulation paper there is a clear lack of a more advanced and selective memory process. But on the other hand, it also presents a very attractive retrieve function using a weighted score formula that takes into account several aspects of memorable data and from which inspiration will be taken for the elaboration of the recall process of this project.

2.4 Suite Architectures

To be able to carry out the development of the system it is necessary to value the different options where this one can be developed. With the explosion that LLMs have had during the last years, different platforms have been emerging recently that are positioned as architectures where you can develop your own system around LLMs with connection to external components such as databases in a simple and functional way. The following are the most important ones at the present time.

LangChain. "LangChain" [85] is self-defined as a suite where you can develop a system based on LLMs with different connections to external elements such as a database or Internet connection. Within this, a logic can be defined to control the actions taken by the LLM and the use of the resources around it.

The main disadvantage of using this environment with respect to this project is that it is intended for the deployment of a single LLM that is responsible for the overall management of the [86] system. As latter will be seen (Section 5.1.1), in this project we will test a system with more than one LLM with different functions such as extracting the memorable data and carrying out the conversation, so it is not the most suitable suite for this work.

LangGraph. "LangGraph" [87] is a very recent new functionality based on the "LangChain" suite but with some interesting additions. Although this new functionality is mainly designed to perform a loop within the system that, in each iteration, allows to infer the LLM with different types of logic, it also includes the possibility of including more than one LLM within the system so that they can interact with each other, creating a multi-agent system.

The coordination between the different LLMs that can be included in the system is done by means of elements called "Routers". These are in charge of analyzing the output of the LLMs and managing the flow from this. The global flow of these systems can be parameterized in flow diagrams where the "Routers" are in charge of modifying the intrinsic flow [88].

For this, inside the prompt that arrives to the LLM, a section of instructions is included that gives knowledge to the LLM of the paths that it can follow within the system (following a philosophy similar to that of the "MemGPT" paper [2.3.4]). If the model thinks that it has the final answer, it can emit it directly, but if it thinks that a processing of this answer is still needed, it can indicate the function to call whose execution will be in charge of the "Router".

This system is positioned as one of the most suitable for the present project, offering a modifiable flow diagram, allowing iterating over the same model several times and using LLMs only for memory management without affecting the final response. Even so, it has some peculiarities that are detrimental to the research and tests to be performed, such as leaving the real-time flow management in the hands of LLMs, the lack of customization in how messages are transmitted between the different modules or a possible lack of granularity in terms of code modification due to the lack of available methods or options that can be discovered during development.

Langroid. "Langroid" [89] is another suite designed to build a functional application around LLMs to fulfill a task in a simple and efficient way. This suite also supports more than one LLM within the system, but the drawback of this suite comes from the orchestration of these models. The way "Langroid" manages several LLM is through mechanism for hierarchical (i.e. tree-structured) task delegation [90].

Thus, these mechanisms are designed so that the first LLM passes its response to the following LLMs and so on until the last one, which is the one that gives the final response that is delivered to the user. In the present project, we are looking for this response management to be done in a much more segregated way in order to experiment with the different possible architectures and sequences, without the last LLM being in charge of

delivering the final response, as it could be a conversation analyzer that does not issue a response.

CrewAi. "CrewAI" [91] is presented as one of the most recent options (12/21/2023) and the easiest to use. This new suite is designed on "LangChain" but with the objective in mind to be able to use several LLMs within the same system (with the same premise as "LangGraph" but with a much simplified use). It has a very simple programming interface where to establish the agents that represent the LLMs with a defined personality or objective and another one for the tasks that relate these LLMs.

Thus, the disadvantage of this suite comes from two branches. First, its orchestrator of agents or LLMs is sequential, meaning that each LLM is executed one after the other and, again, the last of these is in charge of elaborating the final response [92]. Second, the simplicity of the system, although attractive for development, limits the range of operations that can be done within the suite.

AutoGen. "AutoGen" [93] is presented as one of the most interesting options within all the possibilities that exist today. This suite developed by Microsoft is one of the most powerful in terms of message flow management and execution. It has the possibility of including several LLMs within the system and gives them a lot of execution control so that they are in charge of making decisions and controlling the flow.

Thus, this suite presents an internal operation very similar to "LangGraph" both in the management of the agents and in the management of the execution flow. That is why the conclusions that can be drawn are very similar to those drawn previously, highlighting mainly the possible encounter with design barriers during development that limit testing.

The two most notable differences between these two systems that can condition their use is that "AutoGen" offers a greater number of methods and options to elaborate the system to the consumer's liking. In this way, it is also characterized for being a more complex system to handle and to have a total notion of the operation of this same reason.

Reasoning. With all the previous architectures explored and studied, it was decided to develop our own local architecture for the research of this project. In order to be able to carry out a thorough investigation and to be able to test all the ideas in an efficient way, a completely manageable architecture is needed, both in execution flow and in the flow of the messages exchanged during a system iteration.

Although the architecture that can be developed locally will not meet the standards of an application for commercial use, mainly due to latency issues, this is not the objective of this project, which is more focused on the research and testing of the techniques thought and developed during the elaboration of this project.

Thus, for a future work of implementation of this project in a commercial application it would be convenient to use some of these tools as indicated above for its optimization in terms of latencies. The most suitable for this project would be "AutoGen" or "LangGraph", which allow greater granularity compared to their alternatives and may contain an execution flow more similar to the one developed in this project.

2.5 Databases

As will be seen later during the development, a database is essential in this system since a persistent storage is needed to store all the memorable data of personal character that have appeared in the conversation with the user. It is therefore useful to analyze the state of the art of the different databases currently available.

2.5.1 Different types of databases

Within the database world, there are two major subgroups: relational databases and non-relational databases. Non-relational databases, also known as NoSQL, offer a structural flexibility that makes them ideal for applications that handle large volumes of unstructured or semi-structured data.

Unlike relational databases, which require a strictly defined schema, NoSQL databases allow data to be stored in varied formats without a fixed structure, facilitating horizontal scalability and adaptation to rapid changes in data requirements. This flexibility is especially valuable in agile development (such as this project) and in applications that need to evolve quickly, such as social networks, big data analytics, and cloud services, where data structures can change over time [94].

It must be taken into account that this project is a research project, so a clear structure of the data to be stored will not be defined until certain tests are carried out and the characteristics of the system are further defined. Therefore, it is convenient to be able to modify the structure of the stored data as the project develops, since through experimentation, this structure will be adjusted to adapt to the logic of the system.

Thus, we will focus on the non-relational databases that give us this advantage of having a much more flexible structure being able to modify it as we develop the system and store data with different structures within the same database.

2.5.2 Non-Relational Databases

There are several types of non-relational databases: document databases, key-value databases, wide-column databases and graph databases. After having studied them and without going into too much detail about the other types, a documentary database will be used for this system since it is the one that best meets the requirements. This type of database allows us to store the information in the form of documents with JSON or BSON formats, and each of these documents can have a different structure as required for this system [95].

There are several databases within this type of documentary database, but we have chosen the "MongoDB" database to be used during the development of this system. We could use any database of the documentary type, but we have chosen this one because of its strong community on the Internet that allows us to have many resources to solve problems and doubts, as well as its good reputation in uses both in the cloud and locally.

This MongoDB database uses the BSON document format (a binary version of the JSON format) [96]. It offers options to be used both in the cloud and locally (as in our case) and is organized by different collections that contain the documents within the same database. It also offers a python library called "pymongo" that allows us to manage the database directly from the python programming environment in a simple and well adapted way [97].

3 Validation of the Architecture

The main objective of this project is the development of a memory system or architecture around a conversational LLM to provide it with a long-term memory with the objectives of making the interaction with the conversational LLM more enjoyable, increase the feeling of closeness and interest and, therefore, get a better experience when having a conversation with a bot.

To achieve this, we first have to design the system architecture with all the elements that make it up and the processes driven by the data flow and execution. The two main processes of the system are the memory process and the recall process. In order to have a final architecture

and a reliable system, it is necessary to validate each of these processes beforehand to make sure that they work properly.

In this section, we will first carry out an initial specification and design of the architecture thought for this project where the ideas and operation of each of the elements of the system and of the two main processes that form it will be detailed in more detail.

Then, we will see that the recall process is validated by several papers analyzed and researched as this technique has been experimented more than the memorization process. From those papers inspiration was taken and the recall process is designed

On the other hand, the process of memorizing has undergone much less progress in research, so an idea has been thought for this process. This idea will be presented and, to validate and define this, two hypotheses will be presented that must be validated in order to make sure that we can use this technique in the final system.

Subsequently, all the work necessary to validate these two hypotheses on the memorizing process will be carried out. This work involves the comparison between several LLMs on a human evaluation with the development of a benchmark, the validation of the data obtained, the design of a specific score formula for this project and the validation of this formula with trusted techniques and theoretical foundations.

3.1 Initial Architecture

Once we have carried out the research on human memory and the current state of the art of the technologies and works that are related to this project, we already have a better vision of the possibilities that exist and what we want to develop in this project.

A central element of the architecture will be the conversational LLM that will have the purpose of carrying out the conversation with the user. Another element will be a database where all the user's personal memorable data will be stored and which is directly related to the long-term memory seen in the section on human memory theory.

After this it remains to specify the two key processes of this system: the process of memorizing and the process of remembering. For the recall process, several techniques will be used for which inspiration has been drawn both from previous work on the state of the art and from the theory of human memory. This recall process will be based on the calculation of a weighted score based on several attributes of the memorable data such as its similarity to the current context of the conversation, the longevity of the memorable data or its importance.

For the calculation of the weighted score, inspiration was taken from the recall process of the Stanford paper on LLM agent simulation (see Section 2.3.4). For the semantic similarity with the actual context of the conversation we took inspiration from several works that seek to increase the virtual context window and personalization (see Sections 2.3.2 and 2.3.3).

For the longevity of memorable data we draw on the human memory theory of long-term memory and again on the Stanford paper (see Section 2.3.4). And finally, to measure the importance of the memorable fact, we use the number of repetitions of how many times that fact has appeared in the conversation, taking inspiration from the paper on the personalization of LLMs (see Section 2.3.3) and the theory of human memory about making the data conscious (see Section 2.2.5).

Regarding the process of memorizing the data, there is not much development in the current work because most of them store all the data of the conversation in the database and do not make a previous selection. This results in an unnecessary waste of resources both in size of the database and in the time to process the entire database to retrieve the memorable data at the time of executing the recall process.

Therefore, we wanted to think of a way to select and extract only the memorable personal data from the conversation. In human memory, the process of long-term memorization occurs when data from the short-term memory is transferred to the long-term memory. For this to happen, the person must review these data in the short-term memory and make them conscious by paying attention to them (see Section 2.2.5).

We have taken inspiration from this process and we wanted to propose a hypothesis to be validated: *Do current LLMs have the ability to perform the task of extracting personal memorable data from a conversation with results that can be correlated with those extracted by a human person?*

At the same time, the validation of another hypothesis will be carried out: *Do current LLMs have the ability to estimate how long a memorable data is remembered and, therefore, to estimate its importance with results that can be correlated with the estimation that a human person would perform?*

Thus, if we can validate the first hypothesis, a specialized LLM could be used to extract the memorable data from the conversation with the user and this would be a key element in the process of memorizing the data. And if we can validate the second hypothesis, this information could be used in the recall process.

Therefore, in order to design and develop the final architecture, it is first necessary to experiment with these two hypotheses and try to validate them. During the next two sections we will carry out the methodology required to validate this hypothesis. This methodology involves human participation and assessment as it is a very subjective aspect, the design of a score to compare the results of the LLMs for this scenario and the validation of these results using contrasted techniques.

3.2 Preparation of validation of memorizing process

During this section, we will first explain the mental process behind the thought about the two hypotheses proposed to be validated in order to design the final memory system. Then we will carry out all the preparations required prior to following the validation methodology designed for these hypotheses, such as the choice of the LLMs to compare, the obtaining of the base data on which to obtain results, the preparation of the environments to run the LLMs, the design of the instruction and sequence of prompts through prompt engineering, the design of the human assessment test to be carried out by the volunteers, and the obtaining of all the results required to validate both hypotheses.

3.2.1 Initial thought.

As discussed above, the focus of this project is to implement an optimized memory system that only stores memories that are worth remembering in the future. By doing this we achieve that a chatbot has the ability to recall data from past conversations in a pseudo-infinite time frame without the need to increase the context window with the attendant problems such as cost, resource expenditure or carbon footprint. To this end, one of the first aspects to consider and investigate is how to detect data and facts related to the user that are worth remembering from a conversation with the user.

This selection system cannot be based on a rule-based system since the nature of memories and in general the memorable data of a conversation do not allow them to be identified simply by their syntactic structure. A ruled system could cover certain areas such as the detection of specific concepts like places, proper names or attributes through the use of syntax, which is responsible for analyzing the structure of the sentence and being able to intuit the characteristics of each of the terms of the sentence. This technique lags far behind in terms of capacity because it could only be based on the structure of the sentence without attending to a higher level of abstraction where not only the characteristics and relationships

between the terms of the sentence are taken into account, but also their meaning and context from a higher point of view of meaning and history.

Putting the focus on artificial intelligence and, more specifically on the most recent advances, we can easily realize that this task is classified within Natural Language Processing (NLP). In this field, the models based on Transformers[10] are the ones that have recently had a more exponential growth and are the most capable in this field, so we can explore this idea of using a model based on Transformer. As seen previously in the explanation of these, the first transformer-based models already achieved good results in tasks such as text classification, summarization, translation or feature extraction. Some examples of these that mostly fall into the simple transformer-based models are “BERT”, “T5”, “RoBERTa”, etc.

In order to adapt and train them in the task of extracting personal memorable data in a conversation would require a large dataset where both conversations between humans and the data that is considered memorable from that conversation would be found. This type of dataset requires human annotation of each of the conversations to decide what data is memorable in each of the conversations and unfortunately no such open source dataset exists.

Thus, we now focus on large language models (LLMs) to perform this task. Today’s LLMs have a capacity to understand human language that a few years ago would have been thought impossible. They also have the ability to carry out tasks for which they have not been trained using instructions and prompts. This makes them suitable for conducting experiments on whether they are capable of performing the task.

The task to be performed by these models can be specified both in the instruction and in the prompt that is passed to the model and by means of a technique called “prompt engineering” optimize it to the maximum to obtain the best possible results. Thus, it has been proven that in many scenarios where it is not necessary to carry out a very complex and specific task, these models are able to generalize and perform the task assigned in the instruction and in the prompt with very good results.

This action of assessing at a precise instant the data and facts of a personal nature that are considered memorable for different time projections cannot be directly related to the human memory process for these different time projections, since this process involves many more circumstances. This latter process cannot be replicated outside the human mind, but the hypotheses to be validated can be compared taking into account certain considerations that will be explained. This is explained in more detail in Section 3.2.7

3.2.2 Selection of the LLMs

For the choice of LLMs that we will compare during the validation of these two hypotheses on the task of extracting memorable personal data and its projection over time, we will focus on both open source and proprietary foundational models.

The selection of these models and not community-trained models lies in the fact that these community models are finetuned versions of the foundational models, so the comparison of the foundational versions gives us a good basis on which to do experiments. In addition, these foundational models are better documented both in performance and in the data used in training.

Among the range of existing foundational models to be compared, four different models will be chosen as this number is considered sufficiently representative to have an idea of the capabilities of the most advanced models at the time of this experiment (February 29, 2024). Starting with the open source models, we have chosen three open source models that allow their use on a local machine. The model that will be used in the final system will be one of these 3 models because of two aspects: they allow to run locally and they are open source.

The requirement of being able to run on a local machine comes from the reliability of this by having total control over its execution without having to depend on a company that has available an API which can change or be revoked at any time. On the other hand, the quality of being open source models is the most striking. We want to give visibility and recognition to the capabilities of these models. Also this quality allows a possible future commercial or recreational use in some scenarios where it is fully free open use. These two aspects allow anyone to use this system on a local machine free of charge, free to use and instantaneously.

The first model chosen has been the recently published Gemmma model by Google [98]. This has been the latest foundational model to be released in open source format as of research date. The top version consists of 7 billion parameters and its creators attribute to it the best results among the open source foundational models for its size, even surpassing the 13 billion parameter LLama2 model of Meta.

Gemma models share technical and infrastructure components with Gemini (Google's proprietary foundational model). In the paper where this model is presented it is specified that the Gemma model architecture is based on the transformer decoder [10] and that it uses some techniques already commented previously as rotary positional embeddings or the reuse of embedding to optimize the use of the memory [99].

The second model chosen was the Mistral-7b by the French company Mistral AI. This was positioned as the best open source model in terms of size-performance ratio before the appearance of the gemma model by Google, which is on a par in terms of results according to Google. The Mistral-7b paper specifies that its architecture is based on Transformers [10] and that it uses techniques such as Sliding Window Attention, also seen above, or the Rolling Buffer Cache to achieve a better use of memory and the context window [100].

As the third open source model we have chosen the Llama2 model in its 13 billion parameter version. The Llama2 family of models marked a turning point when Meta released them in July 2023 [101], becoming at that time the best performing open source foundational models by a wide margin [68].

Currently they are not among the best rated models in the various benchmarks (especially among models with the same number of parameters) by the appearance of models with higher capacities in recent months, but still wanted to introduce a model of this family in the comparison because of the importance they have in the open source world. The model chosen has been the 13 billion parameter model that has higher requirements to run locally compared to the other models but it is still feasible to run it locally on consumer hardware.

As mentioned above, a proprietary model has also been included in the comparison, although it is not intended to be used in this task of extracting relevant data and facts. This is because we wanted to include in the comparison the model currently with the best capabilities and results in order to have a best case scenario with which to compare. This can give us an idea of the future of open source models and the capabilities that they may have, as well as to observe the results that gives the best current LLM can draw interesting conclusions.

For the foundational proprietary model, the GPT-4 model has been chosen by the OpenAI company. This multimodal model was initially presented in March 2023 [102] and since then it has been retrained and improved with more and more updated data. The paper published by Open AI [103] does not give any information about its operation and internal structure other than the fact that it is based on the transformer architecture [10].

The only relevant extra information about this model comes from a comment of one of its creators in an interview where he indicates that this model uses the Mixture of Experts (MoE) technique supposedly with 8 models of 220 billion parameters each, increasing its number of parameters up to $8 \times 220 = 1760$ billion parameters. As of the date of this experiment, the GPT-4 model is globally considered to be the best public model in reasoning capabilities.

In the Table 1 and in the Figure 7 a comparison between the 4 models to be compared for the validation of the two hypotheses on the extraction of memorable personal data and their projection over time can be seen. It should be noted that the results used in this first comparison come mainly from the Huggingface leaderboard as the results presented in the companies' papers of these models may vary from the current tests [104]. The BBH (Big Bench Hack) benchmark results were not found on that leaderboard, so in this case the official data has been used. The benchmarks used to compare the models are those most closely related to the task described. Benchmarks related to other aspects such as coding or maths are not represented.

These benchmarks are: "MMLU" (Massive Multitask Language Understanding) [105] assesses the ability to comprehend language models in a wide range of knowledge and reasoning tasks, "HellaSwag" [106] challenges language models to complete complex and ambiguous scenarios, testing their understanding of context and narrative coherence, "ARC" (AI2 Reasoning Challenge) [107] assesses complex reasoning and problem solving in multiple-choice questions designed for high school students, evaluating the deep reasoning ability of models, and BBH (Big Bench Hack) [108] designed to test the ability of language models on tasks requiring creativity, real-world understanding, and logical reasoning, addressing limitations of existing benchmarks.

Benchmark	Gemma 7b	Mistral 7b	Llama2 13b	GPT 4
MMLU	66.03	64.16	54.8	86.4
HellaSwag	82.47	83.31	80.7	95.3
ARC	61.09	59.98	52.7	96.3
BBH	55.1	43.2	39.4	???

Table 1: Benchmarks comparison between models

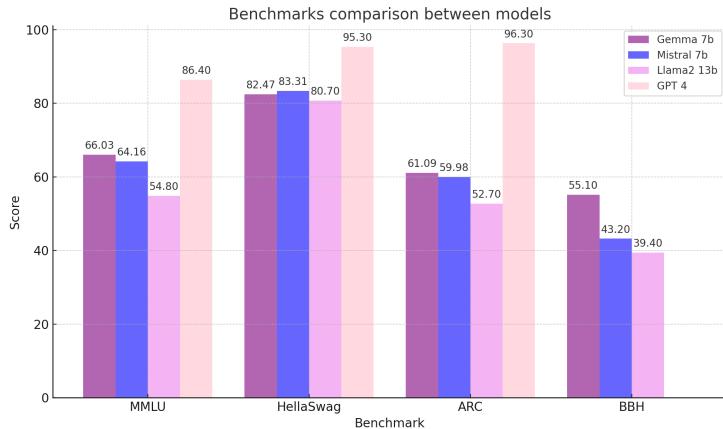


Figure 7: Benchmarks comparison between models

3.2.3 Design of the human evaluation test

The task to be carried out by a large language model is to extract data or facts of personal character that are worth to be remembered from a conversation for different temporal projections. This task has a very subjective character and depends a lot on the personality of each person and to what this person gives importance (See Section 2.2.4), so it is not a task that can be evaluated by the existing benchmarks that focus more on global aspects within the reasoning and understanding of the language.

In order to carry out the experiment and therefore the validation of both hypotheses, a dataset with conversations that has previously been manually annotated by humans is needed. They would have to have read and internalized the conversation in order to then be able to express which data or facts they consider memorable for each of the temporal projections.

A search process has been done and there is no such open source dataset. Some similar datasets have been found where summaries of conversations are made in order to identify the most important information. Also others where segregated memorable data are extracted as we desire, but they also add extra information that is not present in the conversation [109]. In order to contemplate this specific evaluation and the validation of the two hypotheses to be made in this project, it has been necessary to carry out a human evaluation to obtain valid data.

In order to carry out this it will be necessary first to design the test in such a way that it is easy for people to understand and at the same time try to reduce to the maximum possible problems such as the human bias that each user presents. This will be present in the results since each person has a different conception of what is considered memorable, but we will try to mitigate it as much as possible by interweaving the responses of different annotators.

Main idea of the test. The first aspect to consider in this test is that we not only want to extract the personal data that we consider memorable from a conversation, but also to project our self into a future situation and consider which of the initial data we should keep for different time intervals in the future. To give an example, when we have a conversation with a person, at the end of the conversation we will have in mind much of the present information that we have had during the conversation. This information could be considered to be that which is located in the Working Memory or Short-Term Memory as explained in the section on human working memory (Section 2.2.5).

After 1 hour, this information will be more scarce and we will only remember the things that have had more relevance according to our perspective, which would correspond to the fact that a piece of information has passed to the Long-Term Memory. After 1 week the information that is preserved from that conversation will be much less, so moving forward in time until after 2 months or 1 year where only the information that is very relevant is preserved or, possibly, all the information from that conversation has been forgotten.

Mitigate Bias. As mentioned above, the act of memorizing one fact or another and preserving it in memory for a longer period of time depends directly on the person's perception, and this is where the human bias mentioned above comes in. In order to reduce as much as possible the human bias, under the circumstances in which we find ourselves, it has been decided to make a cross evaluation of the different conversations in such a way that a single person is not in charge of extracting the memorable data of the same conversation over time, but that several people are interlaced in this selection according to the specified time.

Temporal Projections. The temporal projections referenced earlier about the estimation of the duration of the memorable data have been defined as 1 hour, 1 week, 2 months and 1 year. The first 1-hour projection is the one in which we make the equivalence with the transfer of data from working memory to long-term memory as in human memory theory [2.2.5], since these come from a very recent interaction but still persist for a period longer than the size of the working memory. We will also make the direct equivalence of the definition of memorable data with this 1-hour projection since it implies giving sufficient importance to these data to be remembered without entering into future estimations of their temporal duration.

Test Considerations. In the design of this test, it was decided that users should extract personal facts and events that they consider memorable rather than summarizing memorable facts for two reasons. The first is that memory functioning is based on data and experiences, not on summaries of conversations that may contain data related to various topics. Summaries would allow one to see the ability to pick out the most important ideas from the

conversation, but in the memory implementation these ideas or data have to be segregated because they may refer to different concepts and therefore have to be treated differently.

The second reason, and more important in this section, is that extracting the data in this segregated form allows the comparison to be more direct and efficient. Comparing two summaries could give rise to different opinions because the assessment of the summaries is more ambiguous, as they may be made up of different topics. Certain evaluation criteria should be specified, such as, for example, giving more importance to a data item that is mentioned for more characters in the summary than to another data item that is mentioned only once, but this does not always correspond to reality. However, comparing two segregated data has much less ambiguity and their similarity can be reasoned much more effectively even in the scenario where the information of one data item is reflected in more than one sentence.

Test for the LLMs. With the evaluation already designed for the human volunteers, we can adapt it to the LLMs to obtain from them the data to be compared. For these we will need to design an instruction and sequence of prompts to explain the task to be carried out in the best possible way. The temporal projections will be kept the same and the LLM will be asked to assign a temporal duration for each memorable data extracted from the available temporal projections (see Section 3.2.6). This task will be performed by each LLM for each of the available conversational data.

The operation of this whole test has been exemplified in the figure 8 which shows both how the LLMs extract the memorable data from the conversation for each of the temporal projections and the interlacing between the different users with the aiming of reducing the human bias as much as possible.

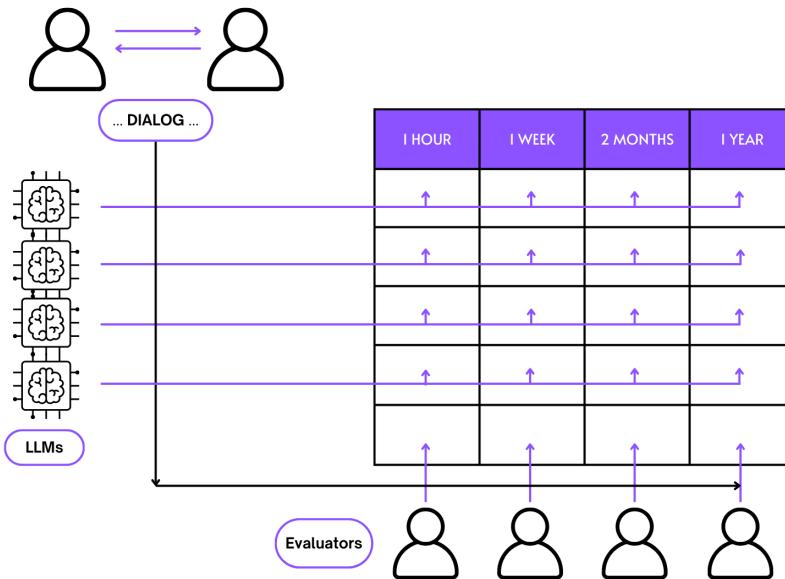


Figure 8: Diagram of test operation

3.2.4 Conversational Data

With the structure of the test already thought out, we are faced with the need to find the data corresponding to the dialogues between people in order to be able to evaluate the task. These data need to be conversations between real people in order to resemble a typical day-to-day conversation. It is intended to dispense with conversational data between people and bots as these generally do not resemble the reality of a dialogue between two people and we want to carry out this evaluation in situations as close to reality as possible.

Requirements. Thanks to the fact that the research field of large language models oriented to conversations is booming, there are several open source resources on which we can investigate. First, as mentioned above, we have made a discrimination of simulated dialogues in order to test the capabilities of the models on real study situations. On the other hand, the search has been restricted to English dialogues as well, since the multilingual skills of the open source LLMs used in this experiment are not good and much of their potential capability would be lost.

Another requirement is that the conversations are long enough to contain several different types of information and topics, so that the comprehension and understanding capabilities of the model and the people can be satisfactorily evaluated and useful results achieved. Much of the data available on the Internet is based on short conversations, question answering or even long conversations but on a single topic, so this is a major limitation in the search.

Research. After an extensive search we have found interesting databases such as one developed by the “Alexa” team, part of “Amazon”, called “Topical-Chat” [110]. This database contains over 8000 conversations between two real people averaging 20 turns and focused on a topic general enough to include different subtopics within it. These types of databases, while meeting the requirements mentioned above, focus on facts (most of which an LLM should already know as they belong to general knowledge) and not on information or experiences of a more personal nature.

Thus, as a last requirement that has been established for the search, observing that most of the conversation databases only talk about general knowledge, is that the conversations to be searched should be of a more personal nature. That is to say, for this test we want to use conversations where users talk about their personal experiences, life habits and attributes that are considered personal.

This is because the final project to be developed is focused on interacting with a human and remembering things about him to simulate showing more interest and seem more enjoyable. Therefore, this experiment seeks to compare the ability to extract memorable information related to a conversation where users are talking about their personal lives and their own past experiences. Factual information of general knowledge is considered to be outside the scope of this project as it should be embedded in the LLM and does not provide a sense of personalization to the user when talking to the chatbot.

With all the requirements mentioned above, an extensive search for databases to use has been carried out, but no public database has been found that meets the needs. All those that were explored such as "DailyDialog" [111], "EmpatheticDialogs" [112], "PersonaChat" [113], and the earlier mentioned Topical-chat [110] had some drawbacks. Some of these are the short duration of the conversation, the feeling of being simulated conversations that do not resemble day-to-day conversations, lack of first-person personal experiences, etc. Thus, it has been decided to change the paradigm and collect real conversations that follow the needs imposed for this experiment and therefore give us the most interesting results possible.

Simulation of Conversations. To collect these conversations, five related volunteers were sought out and asked to have a paired conversation about their life experiences independently. The requirement that they be related to each other stems from the nature of the friendship relationship between them. By having a friendship relationship, volunteers are more likely to express personal and in-depth information about their personal experiences, which is the ideal scenario to obtain with the conversational chatbot system.

Thus, the five chosen volunteers come from the same university residence and have friendly relationships with each other, favoring more personal results. This group of volunteers is composed of 3 men and 2 women between 23 and 25 years old. Each pair of this group of volunteers was asked to have a conversation about experiences and personal attributes of

each of them based on a very general topic. In this way, 10 conversations were obtained to form the corpus of data for the evaluation.

This number of conversations has been considered sufficient to get an idea of the performance of the models in this task of extracting memorable data. The number of people required to evaluate the conversations has also been a limitation when extracting more conversations, since this number could easily scale and we do not have a very large number of people for this evaluation. As well as the task of evaluating and comparing the results, which, as will be seen in Section 3.3, has a human annotation aspect and is a task of exponential growth of workload.

The following is an example of the beginning of a conversation to observe its structure [9]. Also, in the Table 2 the dataset is analyzed and summarized. All the entire conversational dataset can be found in the Annex C. This dataset of conversations fulfills the objective of having a long duration of near 20 turns on average, dealing with personal issues of the people involved and covering several different topics as a whole.

agent_1: Do you like animals?
agent_2: Not very much, I'm afraid of dogs for example.
agent_1: Well, I love them, I have a lot of pets.
agent_2: Do you have a dog?
agent_1: Yes, he is my best friend, if you want I can introduce him to you.
agent_2: I'm not sure, I'm afraid of him.
agent_1: My dog is not dangerous, he is very small too.
agent_2: What is his name?
agent_1: Yew, he looks like a cuddly toy, he has never attacked anyone and he is very good.
...

Figure 9: Example of conversation structure

Metric	Average	Range
Number of Turns	18.4	16 - 19
Total Words	211.0	167 - 235
Words per Turn	11.49	9.28 - 13.38

Table 2: Statistics of the conversations

3.2.5 Preparation of the environments of LLMs

Once the conversational data has been extracted and prepared, we are in a position to begin obtaining the results on which to subsequently evaluate. The first part of obtaining the results will be based on obtaining the responses of the LLMs to be compared for each of the extracted conversations and for the different time projections. That is why the first step is to collect the large language models and prepare them to pass the conversations to the LLMs.

The three open-source LLMs chosen (Gemma 7b, Mistral 7b and Llama2 13b) are models that require less resources when compared to other much larger models and therefore we will try to build them on a local machine using consumer graphics cards without having to make use of external platforms or APIs. For this we will use an environment with two 3090

graphics cards with 24GB of dedicated graphics memory each, which should be sufficient for this experiment.

Optimized Environment. In order to run the models locally, it is necessary to use optimization techniques to store the model weights in the graphics memory. These optimization techniques include, for example, weight quantization which is based on using model weights with lower precision (using weights in precision fp16 instead of fp32 for example) or model offloading between the GPU and CPU to move parts of the model to the CPU when the entire model does not fit in the GPU's graphics memory.

To implement these techniques, we are going to use a recent suite called "Ollama" [114]. This suite allows us to launch open source models locally. The differentiating factor of this suite, and why we have chosen to use it instead of writing the code manually, is that it is specially oriented to optimize to the maximum the management of the resources at the time of executing a large language model.

For each of the open source models offered in this suite (where we can find all the open source models to be compared in this experiment), the "Ollama" team has developed specific configurations to optimize the use of these models. Thus, even with a manual configuration by hand, to reach an equivalent optimization would require an effort that can be saved by making use of this open source suite dedicated specifically to this.

Another of the requirements that have been checked before using this suite is that it allows us to modify the instructions of the models since we are going to use our own instructions that will be developed later. This suite allows us to create a file called "Modelfile" for each of the LLMs to be used where we can specify both the custom instruction and other parameters such as temperature to adjust the desired level of creativity [115].

Proprietary LLM Environment. As for the proprietary GPT4 model, it is accessed either through its web interface directly from the ChatGPT [116] page or by using the API provided by the OpenAI team to access this model from a development environment such as a Python file. For either option, this model is only accessible for a fee.

In order to access the ChatGPT web interface, we need to have a ChatGPT Plus account with a current cost of \$20 per month per month. This payment allows us to use the GPT-4 model and several other functions found within the ChatGPT suite. As for the API, the cost of this depends on the number of tokens that are required to use and its price varies periodically. The current prices of the API can be seen in its official web page [117].

As for this experiment only this model is going to be compared with respect to the others and to the human evaluation, and as it is not planned to use this GPT4 model within the final system of long-term memory implementation in a LLM, it has been decided to pay the subscription for 1 month of the ChatGPT Plus account to obtain the necessary data for the comparison.

Recently the owners of this model have allowed to modify the instruction that is passed to GPT4 directly from the ChatGPT website, so to perform the experiment we have accessed the "Customize ChatGPT" section where we will specify the designed instruction and then we will enter the prompts directly in a normal conversation.

3.2.6 Prompt engineering on LLMs

Once we have the large language models prepared to make use of these and the conversational data, we need to know the format of how to explain the action to be performed to the LLM so that it understands it and the best possible results are obtained. Although this seems to be a trivial action in which by explaining it in one way or another the model should understand it and perform the corresponding action, in reality this is not true and even less so with small models such as those being used in this experiment.

Relevant Considerations. As seen in the section on Large Language Models and more specifically in the section on Prompt Engineering (Section 2.1.3), the design of the prompt that is passed to the LLM to perform an action must be studied in depth by experimenting and observing the results iteratively. Therefore, for the design of the sequence of prompts for this scenario of extracting the data and memorable facts of a conversation in function of the time projection, we have followed some guidelines on the good practices for the design of the prompt [20] [21] and iterated several times on the prompts themselves until the most satisfactory results were obtained.

Another important aspect is the instruction that is configured prior to the introduction of the prompt. This aspect is not included in the prompt engineering best practice guidelines since it is an aspect that cannot be customized in some of the most used proprietary models, but it also has a relevant importance and is aligned with the same philosophy of prompt optimization since it indicates how the model itself should behave and act. Thus, this will also be developed iteratively observing the results to keep the instruction that gives the best results.

Instruction. As explained, this parameter is found within the configuration of the parameters of the model itself in the “Ollama” suite and allows the LLM to have an idea of how to act and behave. Although it could be specified within the instruction itself the task to be performed by the LLM, following the good practice guide, it has been preferred to separate these concepts and in the instruction only indicate to the LLM how it should behave.

The instruction developed that gives the best results after many iterations is the one that explains to the model that it must simulate the thinking of a real person in an everyday situation where it is having a conversation with a friend and that it must perform the proposed action in the most concise way possible and in line with the reasoning that a person could have.

Prompts. In the design of the sequence of prompts for this specific task, an important key has been to explain in a very detailed and structured way the action required to be performed by the LLM.

First of all, the context of the test and the main action to be performed is well explained, putting in context that a conversation between two real people is going to be played and that its objective is to extract the data and facts that a human would consider relevant to memorize when this conversation is over. This has been specified because it represents a similarity with the human memory process of passing the data from the working memory to the long-term memory once an interaction is over.

In order to project these memories to different time periods, this action has been introduced in the next prompt of the interaction with the LLM following the Chain of Thought technique [23]. As explained in Section 2.1.3, this technique argues that for best results we should ask the model step by step how it performs a task and then separate these steps into several turns. In this way, it has been seen that it first extracts the memorable data and then assigns them the duration in time, so this last action has been separated in a second turn. In this, the action of considering which of these memories a human would remember between different time projections has been explained to him.

Unlike the test developed for volunteers, LLMs are not asked to extract personal memorable data for a temporal projection. They are asked to assign a temporal duration to each of the memorable data they have extracted after the first prompt from a series of indicated options (1 hour, 1 week, 2 months or 1 year). This has been done because it has shown the most consistent results. Otherwise, the LLMs were sometimes delusional in entering new data or not making an effective differentiation.

Relevant Insights. Along the iterative experimentation designing of different prompts, while observing the provisional results of the models, some relevant insights have been obtained

about the behavior of these models that are worth commenting on. A clear mismatch has been observed (as expected) between the small open source LLMs and the large GPT4 model both in understanding the instruction and sequence of prompts and in representing the results in the required format.

One of the ideas tested was to explain to the LLM that it must first put itself in the situation of “agent 1” to extract a list of memorable information about “agent 2”, and then put itself in the situation of “agent 2” by extracting the memorable information from “agent 1”. After trying many different ways, small open source LLMs have not been able to perform this task correctly while the GPT4 model understands the task and executes it as explained. Therefore, this idea was dropped.

For open source models it is necessary to be very careful in the design of both the instruction and the sequence of prompts. The slightest change in the prompts can lead to very different results. Thus, we cannot include too much information in the prompts because then the LLM starts to lose detail of what to do and goes beyond some of the explanations they have been given, so we have to be more concise with these models as opposed to the GPT4 model that understands and heeds everything that is indicated in both the instruction and the prompts.

Specifying a little more, within the open source models it has been observed that the 13 billion parameter llama2 model is the one that consistently produced the most coherent results at first sight with the given instructions versus the Mistral 7b model which was the one that varied much more in the form of its results even using the same instruction and prompts sequences and just doing the inference again. Even so, this does not give us information about their ability to solve the task, only information about the models themselves and how the 13 billion parameter llama2 model, despite obtaining worse results in the benchmarks, is the one that behaves more coherently.

For this experiment we wanted to use a common instruction and prompts sequence for all models, so that the way of requesting the results is standardized and therefore they are all in the same conditions. During the tests it has been observed that the open source LLMs generally did not behave consistently with the prompts designed for the other models, while the GPT4 model behaves consistently with all prompts and instructions.

Therefore, following the same premise of having a common instruction and prompts sequence for all models, we have tried to design an instruction and prompts sequence that works in the best possible way in all models. Omitting the GPT4 model since it obtains consistent results for all prompts, in each iteration of the instruction and prompt design we have tested it in all three models at the same time before performing the next iteration in order to obtain consistent results in all three models.

After many iterations, another insight into the behavior of these models has been obtained and is considered relevant. The difference in behavior of the models according to the instruction and the sequence of prompts introduced has been particularly influenced by both the way the data is requested (about the memorable information) and the way in which they are asked to represent it.

Therefore, the latter aspect (the way in which the results are represented) has been given less importance in favor of the model better understanding the task it has to perform and the data it has to obtain. This decision has been taken in order to have more reliable results considering that the objective of this experiment is to validate the two initial hypotheses about the ability of LLMs to extract memorable information regardless of the way it is represented.

The final evaluation by means of the score formula to be designed for this specific project requires a manual annotation since, as these results have different natures from one model to another (even from one conversation to another in some cases), it is necessary to analyze their content to know with certainty if they have the same information as the human evaluation

and if this information is dispersed in several sentences or not, thus giving them a score or another.

If the validation of the hypotheses is satisfactory and a model is chosen to be used in the memorization process, an effort will be made to optimize as much as possible both the instruction and the sequence of prompts in such a way that it not only gives good results, but also represents them in a coherent way among all the inferences in order to obtain the desired data from it and store them in the database.

Designed instruction and sequence of prompts. The following shows both the designed instruction (Figure 10) as well as the sequence of final prompts (Figure 11 and Figure 12) that have been used to perform this test by the LLMs. These have been obtained after many iterations of modifying both the instruction and the prompts and observing the results of the inference for the 3 open source models, seeing if they understood the task and the data requested.

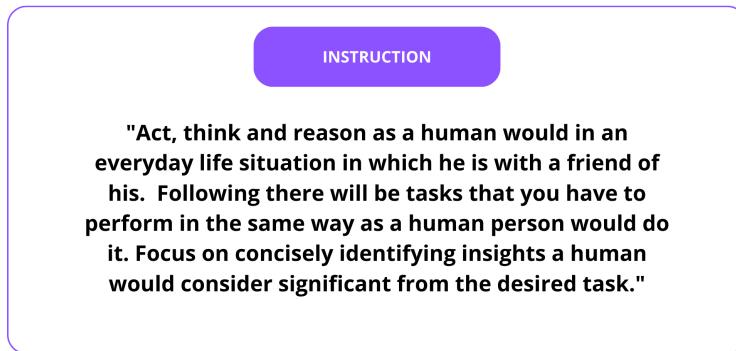


Figure 10: Instruction used for the comparison experiment

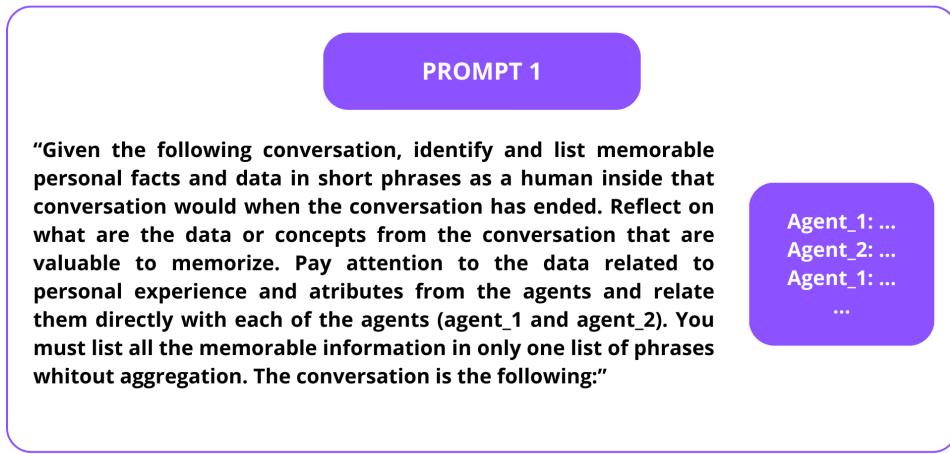


Figure 11: First prompt used for the comparison experiment

3.2.7 Human Evaluation

As previously mentioned, human evaluation is necessary in this scenario since the selection and evaluation of which data and facts are considered memorable and the projection of these

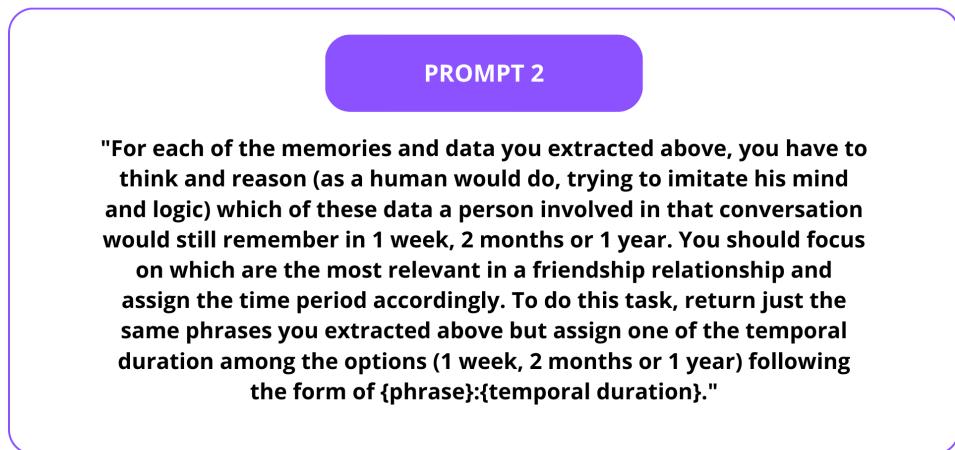


Figure 12: Second prompt used for the comparison experiment

in time is a very subjective process that depends directly on the perception of each person in each situation and on his personality and reasoning as it was seen in the explanation of the process of memory acquisition in the section 2.2.5.

Thus, with the structure of the test already defined previously, a human assessment has been elaborated that involves creating an environment in which volunteers will analyze some conversations and will have to express, according to their criteria, which personal data they consider relevant for different time projections.

Initial Consideration. It should be noted that these tests will help us to have a comparison of the reasoning of the model versus human reasoning at the moment of receiving the information from the dialogue, but these results will not reflect an absolute truth, especially in the temporal projections. This is due to the fact that the memory retention process and the recall process are chemical processes that cannot be equated to an evaluation of a memory at a precise instant towards a projection in time.

In order to obtain a database with golden-reference data, it would be necessary to study a series of human subjects during a period of time not less than the maximum time projection to be evaluated. In this study, it would be necessary to consider different aspects such as the way of establishing the communication between the subjects and the way in which the memories are evaluated since there are several problems such as the mere fact that if the patients to be studied know that they are under a memory study, their perception towards the initial conversation will be more biased and therefore they will remember more data and facts compared to a colloquial conversation of the day to day.

Likewise, this database would have a strong bias depending on the perception and personality of the participants to remember some data or others. This branch of research has an important weight in the understanding of the human mind and because of its complexity there are still no reliable studies that we can make use of. That is why for this scenario we will attend and compare the assessment of LLMs and humans in considering which of the data and facts of a conversation they would consider memorable and at what time projection.

Testing Scenario. Once the user accepts to participate in this test voluntarily, he/she will be sent three different conversations with a specific task for each of them. In the figure 13 we can see an example of the structure of the proposal that is sent to each of the people who participate in the test. The textual format of the final test developed can be seen in more detail in the Annex D.

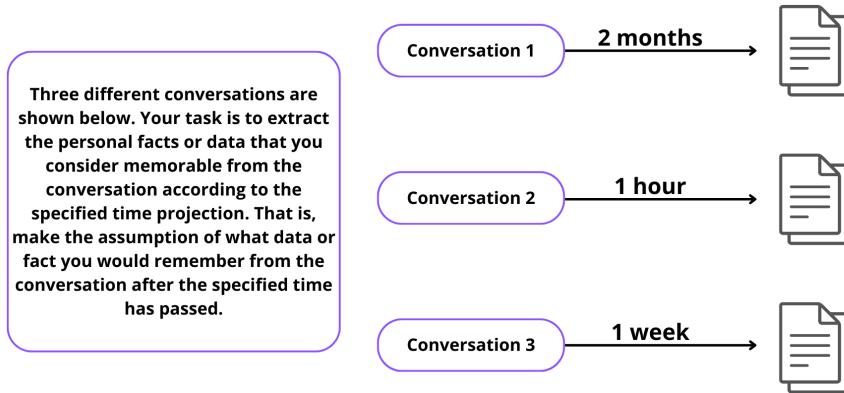


Figure 13: Example of test proposal to the human volunteers

As mentioned above, to mitigate the human bias of each of the volunteers performing the test, the user's answers are intertwined between the different conversations to be assessed for the different time projections. In this way, for the same conversation, each of the extracted memorable data for each time projections corresponds to an answer from different people.

The different time projections that the user has to think about are: 1 hour, 1 week, 2 months and 1 year. The 1 hour response will relate directly to the LLMs response to the first prompt about extracting the data or facts he/she considers relevant from the conversation. These responses to the 1-hour projection, making the equivalence with human memory, would represent those that make the passage from working memory to long-term memory and therefore, are memorable data or facts without depending yet on the temporal space.

This human evaluation has been planned sufficiently in advance to allow several weeks for volunteers to take the test. In this way, volunteers are encouraged to take the test when they have free time and do not feel pressured to do it in a hurry. In this way, more reliable results are achieved in a simple way.

Volunteers. Taking into account that 10 conversations will be used for this comparison experiment (see Section 3.2.4) and that for each conversation there are 4 time projections (see Section 3.2.3), we realize that we need 40 different responses. Therefore, taking into account that each volunteer is expected to respond to 3 tasks so that he/she does not feel overwhelmed, we need 14 volunteers, most of whom will carry out 3 tasks as shown in the Figure 13 and a small percentage will carry out 2 tasks to reach the 40 desired responses.

This set of 14 volunteers has been elaborated by grouping trusted people with the certainty that they will carry out the assigned task in an exhaustive manner. Unlike the case of conversational data generation, these volunteers are not required to be related to each other as the tasks are totally independent. The only requirement for this part is that the volunteers are not the same as those who carried out the generation of the conversational data to avoid possible alterations of the results.

The group of 14 volunteers consists of 4 women and 10 men between 23 and 30 years of age. They belong to different social groups and live in various cities and countries, so that the results are as universal as possible.

Results. The tests have been elaborated for each of the volunteers following the interlocking philosophy explained above to reduce the human bias in the answers. After this, the test

instructions and the three corresponding conversations were sent to each of the volunteers via the Internet.

The collection of the results was carried out over a period of three weeks. This was due to the fact that the instructions gave each volunteer a period of three weeks in order to prevent them from taking the test in a rushed manner and to allow them to take time out of their working day to take the test in a calm manner and to evaluate both the conversations and the answers.

All the answers have been compiled in the same file and will be used in the following section to carry out all the corresponding validations and evaluations.

3.3 Validation of memorizing process

Once we have all the necessary data for the validation collected, we start working with them. As the data collection and final evaluation is subjective to each person, it is necessary to verify and corroborate the data and results step by step in order to obtain at the end of the process results that are considered of real value.

This section has two final objectives. First, to try to validate the two initial hypotheses on whether LLMs have a capacity to extract memorable data and project it in time similar to that of a human person. Then, in the case that this validation is satisfactory, to obtain results that allow us to compare the performance of the models in these tasks.

For this purpose, we will use a final score formula that has been developed for this specific project that allows us to compare the responses of LLMs with those of humans, taking into account the nature of the memories, which are based on subjective aspects, but in order to take these results as true, it is first necessary to contrast this human analysis formula with metrics and techniques that have already been contrasted.

Therefore, first an Inter-Annotator Agreement (IAA) will be performed between several human subjects to check if there is a sufficiently valid consensus between their answers to be able to take the human answers as a valid source of information. If this is accomplished, it will verify both the use of the responses of the volunteers who have participated in the human evaluation discussed in the previous section and also enable the human aspect of the comparison between two sets of responses on memorable data.

After this, a cosine similarity based contrasted metric will be designed and executed using embeddings to compare two memorable datasets, each of the extracted datasets by the LLMs against the extracted datasets from the volunteers. Although, as will be discussed in more detail in the corresponding section, this metric will not cover all the aspects that we want to take into account in the analysis for this specific project, it can be used to have a basis of contrasted results. This base will allow us to check if there is a correlation between this analysis metric based on techniques verified by the community against the comparison metric designed based on a human analysis with a more subjective aspect.

Once these two analyses are done, we will explain the design and objective of the analysis formula designed for this specific project, which covers all the aspects that we want to take into account when comparing two sets of memorable data. We will analyze if there is a correspondence between the cosine similarity metric and the human analysis, and if this correlation exists, we will introduce two more penalty aspects that have been designed around the nature of this particular evaluation (Section 3.3.3).

To conclude this section, we will carry out the evaluation using the score formula designed where the results will be compiled both for the task of extracting personal memorable data and for the task of estimating the time projection of this memorable data.

3.3.1 Consistency among Users

As discussed throughout this project, in order to evaluate the ability of LLMs to extract memorable data, human data have been collected in this same task on the same data used by LLMs to compare the results on both sides and to observe the relationship between these results.

Thus, the first step to be able to analyze these results is to see if the answers given by the volunteers are valid. This is not an easy task because of the subjective aspect of these answers, since each person thinks in one way or another and can give more relevance to one aspect than another in a different way to another person.

A direct way to give validity to the responses of each of the volunteers in the human evaluation is to increase the number of people participating in the human evaluation in such a way that for the same scenario with the same conversation and the same time projection there are three or more responses. In this way, it would be possible to compare these three responses for each scenario and check if they have a sufficiently high correlation between them to verify their completeness and validity.

This has not been possible because of the need for many more human annotators than this technique would require, incurring more than three times the number of annotators that we could commit. Thus, it is useful to think of another way to be able to verify the consensus among several people on the data used and on the task of detecting and evaluating the data considered memorable.

It was therefore decided to use the Inter-Annotator Agreement (IAA) technique with three human annotators. For this purpose, these annotators will go turn by turn through each dialogue of the data used for the evaluation (10 dialogues) and will note for each of these turns how many memorable data they consider exist or are related to that turn.

Among these annotators, for this verification of the consensus between different annotators, there is also a meta-annotator who is intended to carry out the human comparison between the data sets in the final section of the evaluation (the author of this project himself). In this way, we also verify that the subjective aspect of this person has a consensus with other annotators.

Theory behind. Inter-Annotator Agreement (IAA) is a crucial metric in qualitative and quantitative research that measures the degree of agreement or consistency between different annotators who classify, code or label a data set. The objective is to assess the reliability of human annotations, ensuring that subjective judgments are as objective and reproducible as possible.

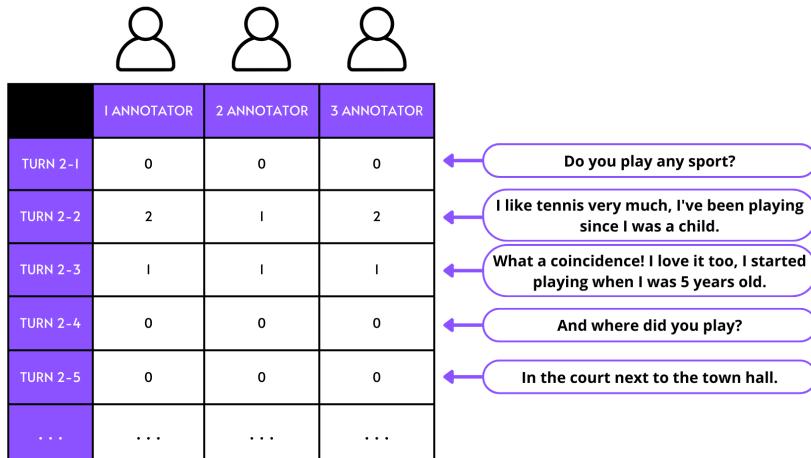
In order to carry out this technique, a statistical measure of inter-rater reliability is needed. In this case, Krippendorff's Alpha has been chosen to perform this statistical measure on any number of annotators and with any type of data developed by Klaus Krippendorff as a general tool to evaluate the consistency of encodings made by multiple annotators, considering the possibility of overlaps per annotator [118].

The Krippendorff Alpha is calculated from a data matrix where the rows represent the units of analysis (in this case, the turns of each dialogue) and the columns represent the annotators. Each cell contains the encoding given by an annotator to a specific unit, being in this case how many memorable data are found or relate to each turn of the dialogue.

Krippendorff's Alpha formula calculates the discrepancies between all encoding pairs for each unit of analysis. This means that it evaluates how much the encodings differ, not only whether they are identical or not. It also adjusts the observed discrepancies for the expected discrepancies by chance. This is crucial because a high degree of agreement could be achieved simply by chance, especially with a limited number of coding categories [118].

To interpret the alpha value obtained, a value of 1 indicates perfect agreement between the annotators, beyond what would be expected by chance. An alpha of 0 indicates no agreement beyond chance, suggesting that the encodings are essentially random. A negative value indicates worse agreement than expected by chance, which is an indicator of serious discrepancies between annotators.

Evaluation of Consistency. With the evaluation already designed we can move on to obtain the data and calculate the Krippendorff's Alpha. To obtain the data we have taken two more volunteers besides the author of this project. These have the task to go turn by turn of each one of the conversations and to assign a value of 0,1,2... to each one of the turns according to the amount of memorable data that they consider are within this turn or have relation with this one. This process can be seen in an abstracted form in the Figure 14.



	1 ANNOTATOR	2 ANNOTATOR	3 ANNOTATOR
TURN 2-1	0	0	0
TURN 2-2	2	1	2
TURN 2-3	1	1	1
TURN 2-4	0	0	0
TURN 2-5	0	0	0
...

Figure 14: Exemplification of the inter annotator performed

After having obtained all the data, we stored them in an Excel datasheet. Since excel does not include the necessary formulas to calculate the Krippendorff's Alpha, a python script has been developed to process this Excel file and calculate the Krippendorff's Alpha.

To do this, the pandas library has been used within python to import the excel file. After this, the data was processed to adapt it to the format required by the Krippendorff's Alpha formula. Once the data is ready, we have used a method included in the library called "krippendorff" that allows us to calculate the Krippendorff's Alpha [119].

Before obtaining the final result, a decision has been made whether to discard the turns where only "0" annotations indicate that the turn is not related to any memorable data. It has been decided to include these turns in the calculation of the Krippendorff's Alpha since the fact of valuing that a turn does not have any relation with a memorable data already supposes a subjective valuation and, therefore, a valid annotation, reason why these scenarios are considered as valuable data for the computation of the Krippendorff's Alpha.

Result. First the annotations of the three human persons have been obtained for this scenario. The dataset consists of 10 conversations with an average length of 20 turns per conversation, so each person had to annotate about 200 turns of conversation.

After this, the script for the calculation was executed with the annotated data and a Krippendorff's Alpha of 0.8199 was obtained. As discussed above regarding the interpretation of this alpha, this value implies a fairly reliable agreement among annotators, thus validating its consistency within the margin considered valid for the subjective aspect of this task.

Thus, after having validated the consistency among several annotators, we can verify the use of the data obtained by the volunteers in this task and make use of them in the following evaluations that are carried out. We also verify the consistency within the margin of error of the subjective aspect for the comparison by hand that will be carried out later between the sets of memorable data extracted by humans and LLMs.

3.3.2 Deterministic trusted semantic comparison

As will be seen in the next section (Section 3.3.3), for this project we wanted to design a new formula to calculate a score that allows us to validate the two hypotheses presented earlier (Section 3.1) and compare the performance between the different models tested for the task of extracting memorable data.

Although this formula has been developed taking inspiration from both other trusted metrics and theoretical aspects of human memory, in order to make use of this formula and have some certainty that the results it gives us are valid, it is necessary to contrast it.

In particular, there is a human comparative technique involved in the designed formula related with the comparison of two sets of extracted memorable data by hand (see Section 3.3.3) that has a more subjective aspect. In order to validate its use, it is required to contrast it with some non-subjective metric already established and tested by the community.

Therefore, a form of evaluation has been investigated and developed using contrasted and deterministic techniques that allows to obtain some results in the task of comparison of similarity between two sets of memorable data of personal character. These results will serve as a basis for, once the analysis has been carried out with the developed score formula, to check if there is a correlation between the subjective comparison and the deterministic comparison.

Initial considerations. For the development of this form of deterministic evaluation it must be taken into account that the same information can be expressed in different ways. The same information can be expressed in different words but mean the same thing. An example would be the sentences “My mother has 3 children” and “I have two brothers”, both sentences refer to the same information but the sentences syntactically do not resemble each other at all. Another example could be “Miguel is a fan of Real Madrid” and “Miguel likes football”, without meaning exactly the same, the information is closely related without using the same words.

Many techniques have been investigated including, for example, Jaccard similarity. This compares the similarity between two numerical sets, so it would be necessary to pass both texts through a tokenizer in order to calculate their similarity [120], thus losing the semantic meaning. The same information can be expressed in different words, so it is not a suitable evaluation technique for this case.

The BLEU score or the ROUGE score are widely used metrics to evaluate the text generated by an LLM. The BLEU Score [121] is more oriented to evaluate the results of translation tasks while the ROUGE Score [122] is used more for summarization tasks. The problem with this type of metrics is that they are based on the comparison of n-grams instead of comparing the meaning of the texts, so they again fail to maintain semantic meaning.

After much research, it has been decided that this deterministic evaluation technique involves the calculation of the semantic similarity through the cosine distance between the embeddings of the different memorable data extracted from both sets of responses. This comparison follows the 1-to-all structure to find the maximum semantic similarity value for each memorable data against the entire other memorable data set.

Theory behind. The cosine distance is used to compute this semantic similarity with the formula "similarity = 1 - cosine distance". This is a measure of similarity between two vectors in a multidimensional feature space that is widely used in fields such as Natural

Language Processing (NLP) or other data science fields. Unlike other distance metrics (such as Euclidean), cosine distance focuses on the angle between two vectors, rather than their magnitude. The resulting value varies between -1 and 1, where 1 indicates that the vectors are identical in orientation (maximum similarity), 0 indicates that they are completely orthogonal (no relationship), and -1 indicates that they are diametrically opposed. [78].

In order to calculate the similarity between two sentences or data sets, it is necessary to transform them into vectors in order to apply the cosine distance using embeddings. Embeddings are vector representations of words, phrases, or documents, which capture semantic meaning and syntactic relationships within a low-dimensional feature space. These vectors are generated using language models trained on large corpora of text, so that words or phrases with similar meanings are close to each other in the vector space.

Using these techniques, we can transform two sentences into vectors that capture the semantic meaning by using a pre-trained language model to generate embeddings. With these embeddings we can calculate the cosine distance and with the formula previously presented, being able to measure the similarity between the two embeddings and, therefore, capture the semantic similarity between the two sentences.

Relevant aspects. In order to perform this deterministic evaluation using the cosine similarity calculation it is not possible to take the two memorable data sets of humans and LLMs to be compared and transform them directly to embeddings. This is because within each memorable dataset there are data with very different meanings and if we tried to represent all of them together using a single vector of embeddings we may lose a lot of important information about them.

Another aspect to take into account is that the information of a memorable data in one set can be found scattered in several data of the other set. This is because the nature of memories varies greatly in the way the concept of memorable data is understood. An example of this scenario might be that one memorable piece of data is “John has a small, friendly dog named Yew” and in the other set of memorable data to be compared there are two sentences that are “John has a dog named Yew” and “John’s dog is small and friendly”. This prevents the use of techniques such as “Greedy matching” where a 1 to 1 pairing between data sets is implied, thus preventing the comparison of already matched data in the following iterations, making it impossible to search for sparse information [123].

The last aspect to consider is whether we consider the human’s response as the true ground truth or whether we give some value to the comparison in the opposite direction. As a general rule, one would have to take the human’s response as ground truth since the objective of the evaluation is to compare the ability to extract personal memorable data against that of a human. But since this is such a subjective task, it might be appropriate to give some weight to the opposite direction of the comparison as well where the LLM could detect some memorable data that the human person did not consider.

Technique of evaluation. Taking into account all the aspects discussed above, a technique has been designed to solve all of them to a certain extent. Some of these cannot be completely solved, especially the dispersion of the information, without other problems such as an altered weighting.

The technique designed for the comparison of two sets of memorable data follows the following structure. First of all, each of the memorable data in the form of a sentence is transformed into embeddings in a segregated manner. This is done in order to capture as much semantic information as possible from each of the data when passing them to the vector format.

For this transformation a public leaderboard has been used by HuggingFace called Massive Text Embedding Benchmark (MTEB) [124]. This leaderboard tests each of the language models oriented to transform data into embeddings against several benchmarks that measure

the quality of these embeddings in different aspects related to natural language processing. By analyzing this leaderboard, we have chosen the text encoder that obtains the best results in tasks related to this experiment.

The text encoder chosen to perform the task of transforming the sentences to embeddings has been the one called "mixedbread-ai/mxbai-embed-large-v1". [125]. As this is one of the best language models currently available for data to embeddings transformation in the specific summarization task (related to the selection of the most important data from a larger data set), we can have some certainty that the semantic meaning of each memorable data will be preserved in its embedding form.

When we have the memorable data in embeddings format we move on to the next step. Taking a set of memorable data, for each of these memorable data we calculate the cosine distance with each of the memorable data of the other set and we select the maximum similarity of all these comparisons, being this the similarity assigned to the memorable data initially selected. This process is performed iteratively with all the memorable data of the initial set and, then, it is also performed in the opposite direction taking the other set of memorable data.

At this point the semantic similarity scores are calculated for each of the memorable data. After this, the average of the cosine similarity considering human annotator as ground truth and the average of the cosine similarity considering LLM annotator 2 as ground truth is computed and its final value is weighted giving more weight to the comparison taking as ground truth the human responses (when comparing each of the data of the human dataset against all the data of the LLM dataset). Finally, a normalized value between 0 and 1 is obtained, indicating the similarity between the two sets of personal memorable data extracted to be compared.

Of the aspects mentioned above, the most complicated to resolve was the dispersion of the information. Initially we had thought of adding all the cosine similarity values for each data instead of taking the maximum to take into account the information dispersion. But this brought with it other problems in the weighting of the data.

In order to try to solve this aspect to be considered, the comparison has been made in both directions. Taking the example mentioned above, when comparing the data "John has a small and friendly dog named Yew" with the other data set, we will not obtain a semantic similarity equal to 1 even if the complete information is present in the data set, since only the maximum value is taken. But in the opposite direction of comparison, when comparing the data "John has a dog named Yew" and "John's dog is small and friendly", two semantic similarity values very close to 1 will be obtained.

In this way, the information dispersion factor is counteracted. There is no ideal way to perform this task automatically because of all the aspects that exist around the characteristics of the memories. But an attempt has been made to develop this technique taking into account all these aspects and, since the same evaluation technique is used for all the sets, the results can be taken as reliable to have a basis on which to later compare the results of the human similarity analysis and see if these results correlate.

The results of the semantic similarity in both directions are weighted with the factors of 0.7 for the similarity taking into account the human responses as ground truth and 0.3 for the other direction of the comparison. This weighting allows us to give more value to the human responses but also to take into account other aspects such as the dispersion of information and the subjective aspect of the human who may not always be right. It should be noted that these values have been proposed after reasoning about the situation and not after an effort to determine their optimum value as there is not a way of contrasting the results.

All this process has been developed in a python script where, through the nesting of 2 loops, all the cosine similarity scores between embeddings are calculated and the maximum is

chosen, saving it in a list to later calculate the average value for each set. Libraries such as "transformers" have been used to import the necessary methods to execute the language model and others related to the cosine distance calculation.

Comparison of results. Performing this evaluation for all scenarios, we found 40 data comparisons taking into account that there are 10 conversations on which to evaluate the ability of LLMs to extract personal data and 4 LLMs to compare against the human responses of the volunteers.

Once all the scores representing the semantic similarity between the data sets have been obtained, they have been stored in an excel file to be able to process them in the analysis. After this, the average of the scores for each of the 4 LLMs is calculated. These are shown in the following Table [3]

Deterministic Comparison	Gemma 7b	Mistral 7b	Llama2 13b	GPT 4
Average	0.8510	0.8615	0.8592	0.8744
Standard Deviation	0.0306	0.0377	0.0465	0.0332

Table 3: Deterministic semantic comparison between models

Observing the results obtained we see that all the models achieve very good results close to 1. This gives us an idea that the first hypothesis about whether LLMs can perform the task of extracting memorable data in a sufficiently human-like manner can be validated, which will be concluded later.

We see, as expected at first, that the model that achieves the best results is the GPT4 while the model that lags behind is the 7 billion parameter Gemma model. Both models, the 7 billion mistral and the 13 billion Llama2, achieve very similar results in the average between the two other models with more extreme results.

We have also computed the standard deviation of these values to get an idea of how the results vary between the different conversations used for the evaluation. We can see that these values are very low, being around 3.5%, which means that these results are very stable between different scenarios and therefore, they are reliable results.

We can also observe how the average values are very close to each other and fall within the confidence bands of all the models taking into account the standard deviation. Therefore, the final results are very similar and any of the models could be used with a similar result according to this deterministic semantic comparison.

3.3.3 Score Formula Design

After having verified the consistency in the ratings between different human annotators and having a deterministic basis for the semantic similarity between the human responses, it remains to design and calculate a scoring formula that addresses the nature of the memories in a more specific way and thus involves a manual comparison between the sets of responses.

Both the validation of the hypotheses and the final choice of the LLM to be used for this project are to be decided not only by the semantic similarity between the data sets, but also to take into account other aspects such as a penalty for the dispersion of information or a penalty when the LLM extracts data that the human does not consider memorable.

Therefore, a score formula has been designed to take into account all these aspects. This will consist of three terms where the first one will be related to the semantic similarity between data sets but with a manual comparison between them and the other two terms are penalties.

For this first term, we will look later if there exists a correlation with the results obtained in the calculation of the semantic similarity by means of the cosine distance to be able to contrast if the manual comparison is valid.

Relevant aspects. The main objective of this score has to be to assess that the responses of human persons are encompassed within the responses of LLMs, such that: $H \subseteq M$ where "H" represents the subset of the memorable data chosen by a human and "M" the memorable data extracted by the LLM. In an ideal case, this score should be maximized when $H = M$, indicating that the subset chosen by the human is equal to the subset chosen by the model.

It is easy to observe that when the subset of "H" is not contained within the subset of "M" it results in a penalty in the score, since it means that the LLM has not extracted the memorable data it required. On the other hand, the case where the subset "M" is larger than the subset "H" must also be penalized (in a lesser way), as this means that the LLM has extracted extra information that is not considered relevant by the human assessment. This is penalized in a minor way as it means extra information that will not worsen the final result, but to follow the premise of valuing the ability to select memorable memories as humans do, it does negatively affect the valuation.

Another aspect to be addressed is to reward the scenarios where the information is equally condensed in the human subset "H" and in the LLM subset "M". This scenario is rewarded because the processing of the answers in the database is done one by one, so it is not interesting that the information of a data extracted by the user is dispersed in several LLM data. The opposite case is also considered, penalizing when the LLM condenses in a data several data extracted by the human, being able to gather information that should be segregated in different data and therefore not being able to make a good management of this information.

On the other hand, it is necessary to assess not only the correspondence between the memories extracted by both parties, but also the similarity between them. That is, it is necessary to assess that the information extracted in the form of memory by a human is contained within the information extracted from the LLM. To give an example for the latter case, the memorable data "John has a dog" and "John's white dog is called Yew" can be valued as the same memorable data but one has much more relevant information than the other.

Score formula. Following these assessments, the following formula has been developed to calculate the score for each scenario:

$$\frac{1}{n} \sum_{i=1}^n v_i - \alpha * |m - n| - \beta * e$$

Where "n" represents the size of the subset of human responses, "m" represents the size of the subset of LLM responses, "v" represents the similarity rating between a person's response and the LLM response, "e" represents the number of data extracted by LLM that are unrelated to human responses, and "alpha" and "beta" are weighting parameters to adjust the importance of these penalty terms.

The calculation of "v" will be done by hand following the following philosophy. For each of the human rating responses, the information contained in that response will be internalized and that information will be searched among the subset of responses by the LLM "M", assigning a score between 0 and 1 depending on the amount of information preserved in the subset "M".

The first term of the formula tries to reward the similarity between the LLM responses and the human valuation responses. Thus, this term also penalizes the scenario discussed above where the subset of human rating responses "H" is not encompassed within the subset of LLM responses "M". This is achieved by dividing the summation by the size of the human response set.

In the case that a human rating response does not find equivalence among the LLM responses, the rating "v" of this one will be 0, so it will not add value within the summation as long as the division by the size of the set of people's responses remains constant, thus applying a penalty to the final result.

The second term is a regularization term that applies an extra penalty to the scenario discussed above on data condensation when the subset of the human rating responses is not encompassed within the LLM responses, but also applies a penalty in the case where the subset of the LLM responses is larger than the subset of the human responses. To contemplate both cases, the absolute value of the size difference is used, so that in both cases a penalty is applied and only in the case that both are of the same size, no penalty is applied.

The third term of this formula directly penalizes the scenario where the LLM extracts memorable data that is not related to any data belonging to the human data subset "H". As mentioned above, this scenario will not be penalized in great quantity since it is extra information, but it is penalized because it seeks to extract only memorable data and not a lot of information from the conversation.

Both the " α " parameter and the " β " parameter are in charge of applying the corresponding weighting to each of the penalty terms. For these two parameters, their values will be defined once the first term has been calculated and validated, so that they penalize this one but not to a great extent.

With this formula, it is possible to calculate a score value that allows us to compare the performance of each of the LLMs in comparison with the human responses, taking into account more relevant aspects about the memories and about the system to be developed compared to an automatic calculation such as the cosine distance only.

3.3.4 Score Formula Evaluation

Once the desired score formula has been designed for the final evaluation of this experiment with all the aspects that have been considered relevant, we will proceed to the calculation of this score for each of the scenarios of the evaluation of the LLMs versus the human responses. Then, will check if for the computing of the manual similarity between the two sets of memorable data there exist a correlation versus the evaluation with the cosine distance.

In order to obtain the data with this designed formula, the two sets of memorable data have been analyzed manually case by case and the different parameters of the designed formula have been obtained. Initially, for this section, we are only taking into account the extraction of memorable data without temporal projection (or only temporal projection of 1 hour as seen in Section 3.2.7). Then, there are a total of 40 cases to be analyzed or this evaluation (10 sets of responses from 4 different LLMs to compare against the human responses)

All these results have been stored again in an Excel file designed to allow the adjustment of the parameters " α " and " β " and to see the change in the results in real time. As well as being able to plot these results on a graph and observe the changes in the graph.

Correlation between deterministic and human similarity evaluation. To check whether there is a correlation between the results of the semantic similarity analysis between the data sets using the cosine similarity and the human manual analysis, it is first necessary to set the parameters " α " and " β " of the score formula to 0. In this way, only the first term, which is associated with the calculation of the similarity to be compared, influences the result.

To see if these two scenarios correlate, both scenarios have been plotted in bar graph format with their corresponding confidence intervals in order to draw conclusions. In these graphs we can see the results summarized where the y-axis is the score normalized to 1 and the x-axis represents the different LLMs. Below is first Figure 15 with the results of the cosine similarity analysis and then Figure 16 with the results of the manual hand similarity analysis.

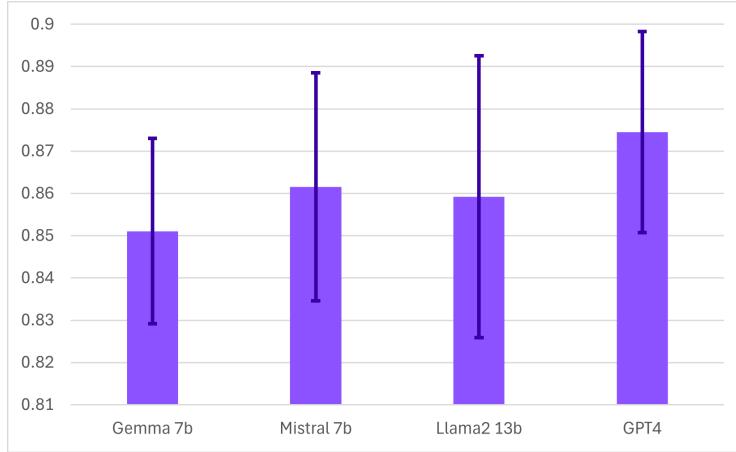


Figure 15: Deterministic Semantic Comparison

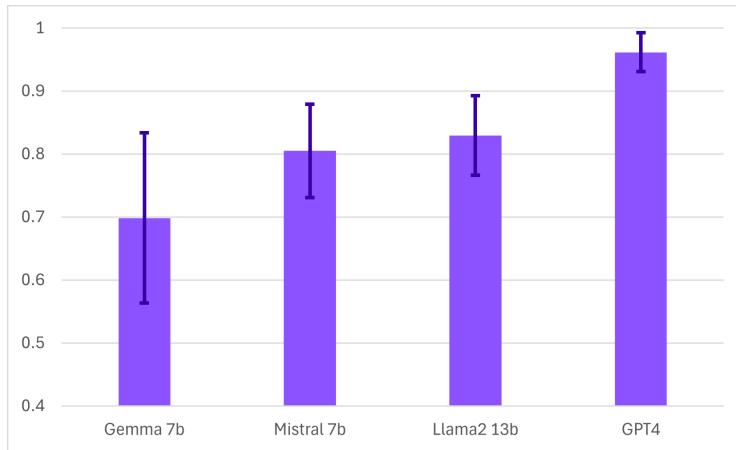


Figure 16: Manual Semantic Comparison

Observing both graphs we can see that they have a similar distribution taking into account the difference in the scale. In both cases the best performing model is the GPT4 and the worst performing is the 7 billion parameter Gemma. The models llama2 of 13 billion and mistral of 7 billion have very similar results in both cases.

The only noticeable difference is that in the deterministic evaluation the 7b mistral model scores slightly better than the 13b llama2, and in the manual evaluation it is the other way around. We also note that the scales are different, in the human evaluation there is a greater difference between the scores of the different models.

For a more in-depth analysis of the correlation of the data obtained, the Pearson correlation coefficient [126] will be used. This coefficient measures the linear relationship between two data sets and varies between -1 and 1, where 1 indicates a perfect positive correlation, -1 indicates a perfect negative correlation, and 0 indicates no correlation.

A python script has been developed that allows us to obtain this Pearson correlation coefficient using the python library "pearson". Thus, a Pearson correlation coefficient value of 0.9795 has been obtained for the correlation of the results of the manual semantic comparison and the deterministic semantic comparison, which indicates a very positive and almost perfect correlation.

Thus, taking into account both the distribution and the Pearson coefficient calculated, we can determine that both results have a sufficiently high correlation to validate and verify the use of the manual metric designed, and therefore we can trust the results it provides.

Manual Comparison evaluation. Once we have validated the use of the manual metric of comparison between sets of responses on the memorable data, we can begin with the analysis of the results of this manual comparison.

In Table 4 we can see the more detailed results of this formula when we have the parameters " α " and " β " set to 0 and, therefore, we are only evaluating how much of the memorable data extracted by the volunteer is contained in the data extracted by the LLM.

Manual Semantic Comparison	Gemma 7b	Mistral 7b	Llama2 13b	GPT 4
Average	0.6984	0.8051	0.8294	0.9615
Standard Deviation	0.1892	0.1037	0.0883	0.0428

Table 4: Manual semantic comparison between models

Looking at this table on the comparison of the information present we see that there is a significant difference between the models. GPT4 obtains an almost perfect score, meaning that practically all the information from the data extracted by the volunteer in the test is present in the data extracted by the LLM. Others like the Gemma 7b model obtains a score indicating that about 2/3 of the information extracted by the volunteer is in their extracted data.

We can also observe the standard deviation. It obtains higher values compared to the deterministic metric with an average of 10%. We can observe that the model Gemma 7b obtains the worst result with a standard deviation of 18.92% while the LLMs LLama 2 13b and GPT 4 obtain the best results with values lower than the average value.

These values are still high enough to have the intuition that it will be possible to make use of these LLMs in the learning process and that they offer added value over the system. The following analysis will decide which of these offers the best performance for this specific project scenario.

Score formula evaluation. With this preliminary analysis done, we move on to activate the other two terms of the formula that are in charge of penalizing certain undesired aspects for the implementation in this system, such as the dispersion of information or the excess of non-relevant information (see Section 3.3). After several tests and observations of the results about the importance of these terms, a value of 0.02 for " α " and 0.03 for " β " has been established.

These values are of such low value because the terms to which these weightings are applied are nor normalized or bounded, thus reducing their influence on the final formula. The value of " α " is lower than that of " β " because it has been decided to penalize more the excess of non-relevant information against the dispersion of information after having weighed it a lot.

Figure 17 shows the results when using new scores obtained by the formula. Table 5 shows the results in more detail, as well as the standard deviation also included. The reasoning about these results will be analyzed later on in Section 4.

Final Scores Comparison	Gemma 7b	Mistral 7b	Llama2 13b	GPT 4
Average	0.5974	0.6811	0.7554	0.7465
Standard Deviation	0.2126	0.1181	0.1128	0.1032

Table 5: Final Scores Task Comparison

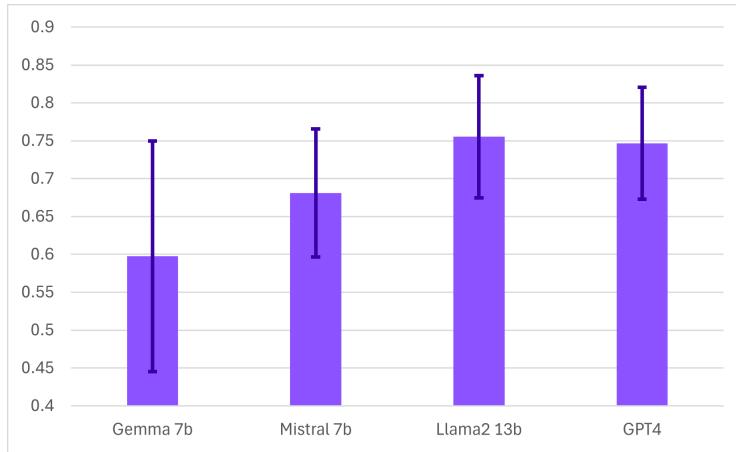


Figure 17: Final Scores Task Comparison

3.3.5 Score on time projections

Once we have verified and carried out the score calculation for the extraction of personal memorable data, we can move on to take into account the rest of the data collected from the volunteers by means of the human evaluation on temporal projections.

Recalling what has been explained during the project, the experiment carried out involved not only extracting personal memorable data from conversations, but also making a temporal projection and estimating the memorable data that would still be remembered for 1 hour, 1 week, 2 months and 1 year.

It is worth noting again the association that has been made previously to the 1-hour temporal projection with the definition of memorable data. Therefore, in this section we do not analyze the 1-hour temporal projection data since these have been previously analyzed in the previous section on the extraction of the memorable data itself.

Grouping Time Projections. In order to compare the memorable data for each of the time projections, the LLM's responses will be grouped in a decreasing order. That is to say, in the case of a 1-week time projection, the data that includes the projection of 1 week, 2 months and 1 year will be taken. In the case of a temporal projection of 2 months, data including the projection of 2 months and 1 year will be taken, and finally, for the projection of 1 year, only data including this same projection will be taken.

This is because the data collection for the LLM is different than for the human subjects in the test. Human subjects are asked to extract directly memorable personal data for a projection because, as discussed above, the responses are intertwined (Section 3.2.7). Therefore, following this data collection format, a person responding to a 2-months extraction does not have the data from a previous temporal projection extraction and cannot take it into account it.

The LLMs value each of the memorable personal data previously extracted by them and assign them a duration during which they believe the memorable data should be remembered. This is why, in order to match results and compare them in an optimal way, it is necessary to make this grouping, which means the same thing but in a different format, allowing the comparison.

If we wanted to avoid this aspect, the person responding to the memorable data for a time projection of 1 week, 2 months, or 1 year would have to have the memorable data for the 1 hour projection as base ground information and evaluate over it. This would have a

significant bias by directly influencing that person's response, so we have preferred to do this extra processing to obtain the best possible results.

Evaluation of time projections. Taking the data already collected on this issue from both LLMs and humans, there are 120 more cases where to manually apply the score formula developed and verified in the previous sections.

After having carried out all the comparisons and analysis of all the cases, the results obtained are shown below. In Figure 18 we can observe the value of the score as the time projection threshold growth. Table 6 shows data for the 1-week projection, Table 7 shows data for the 2-month projection and Table 8 shows data for the 1-year projection. The analysis of these results will be held in Section 4.

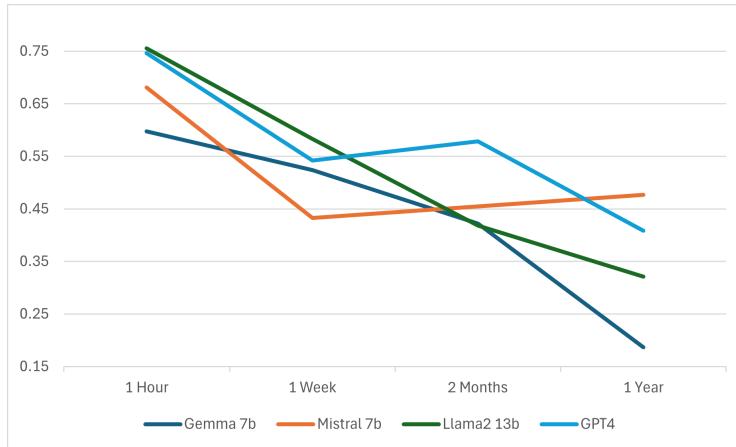


Figure 18: Scores over Time Projections

1 Week Projection Scores Comparison	Gemma 7b	Mistral 7b	Llama2 13b	GPT 4
Average	0.5237	0.4325	0.5829	0.5419
Standard Deviation	0.1052	0.1135	0.1078	0.1660

Table 6: 1 Week Projection Scores Comparison

2 Months Projection Scores Comparison	Gemma 7b	Mistral 7b	Llama2 13b	GPT 4
Average	0.4216	0.4546	0.4180	0.5783
Standard Deviation	0.2578	0.2154	0.2801	0.2443

Table 7: 2 Months Projection Scores Comparison

4 Evaluation Results

In the previous section a series of experiments and evaluations have been carried out with the objective of validating the two hypotheses that were raised during the design of the architecture and to compare the LLMs with each other in the tasks of extracting personal memorable data from a conversation and projecting them over time to estimate how long they should continue to be remembered.

Next, the results obtained will be analyzed and conclusions will be drawn from them. First, we will test whether the two hypotheses initially put forward can be validated. After that, the results of both the task of extracting memorable data and the task of estimating their temporal duration will be further analyzed.

1 Year Projection Scores Comparison	Gemma 7b	Mistral 7b	Llama2 13b	GPT 4
Average	0.1866	0.4766	0.3208	0.4083
Standard Deviation	0.3034	0.2147	0.1920	0.2522

Table 8: 1 Year Projection Scores Comparison

4.1 Validation of the Hypothesis

During the initial design of the proposed architecture, it was observed a lack of previous research on the part of memorization process in the works studied. Most of them stored all the information without filtering and focused on the investigation of a more advanced recall process.

This idea of storing all the information is not interesting because of the unnecessary expense of storing data that are not relevant and also because of the expense in resources associated with the recall process due to having to work with a much larger set of data.

Therefore, a more optimized memorization process was designed which involves the use of LLMs. In order to make use of this, it is necessary to validate two hypotheses raised about the ability of these LLMs to perform two tasks in relation to how a human person would do it. This two hypotheses were previously presented in Section 3.1 and the methodology for the validation of this hypotheses was explained and developed during Section 3 and Section 3.3.

First Hypothesis. The first hypothesis we sought to validate is the following: *Do current LLMs have the ability to perform the task of extracting personal memorable data from a conversation with results that can be correlated with those obtained by a human person?*

This hypothesis is considered to be validated from several points of view. First, with the deterministic semantic similarity using the cosine distance, values above 0.85 out of a maximum possible value of 1 have been obtained. These values indicate a very high semantic relationship between the responses of LLMs and humans.

On the other hand, using the score formula developed for this specific project that takes into account the nature of the memory with specific penalizing elements for a more complete result, sufficiently high values have also been obtained to validate the hypothesis in this scenario.

Specifically, the 13 billion parameter Llama2 model has obtained a 0.75 average score with a standard deviation of 0.11. This result is considered sufficient to validate the hypothesis to a sufficient degree to be able to make use of this LLM in the memory process of the system and have it add value to the system rather than subtract from it.

Second Hypothesis. The second hypothesis that we sought to validate in this experiment is the following: *Do current LLMs have the ability to estimate how long a memorable data is remembered and, therefore, to estimate its importance with results that can be correlated with the temporal estimation that a human person would perform?*

Looking at the results obtained in the previous section (Graph 18 and Tables 6|7|8), we see that these results are not good and get worse as the time projection increases. These results not only have an average value below 0.5 in most cases, but also have a very high standard deviation and, in some cases, even higher than the average value itself.

This indicates that the results are neither stable nor good enough to validate the second hypothesis in this scenario. Thus, it may require a more extensive analysis with more annotated data, further experimentation with prompt engineering and even a possible finetuning of the models to improve results. The results, although low and unstable, are promising.

Conclusion. Observing the results we have been able to observe how the first hypothesis could be validated in order to make use of LLMs in the task of extracting memorable personal data. On the other hand, the second hypothesis could not be validated due to the low and highly variable results obtained.

Therefore, for the design of the memorization process of the system to be developed, LLMs will only be used for the task of extracting personal memorable data and will not be used to obtain the temporal projections of these memorable data. This leads to a simultaneous adjustment in the recall process as these data are not available as will be explained in the section associated to the final design of the architecture.

4.2 Results of extracting memories

Starting with the analysis of the results, we will first focus on the task of extracting personal memorable data by observing the Figure 17 y la Table 5. The results of the score formula developed indicate that the LLM with the best score is the Llama2 with 13b parameters followed closely by the LLM GPT4. These are followed by the Mistral 7b model in third place and the Gemma 7b model in last place.

Looking deeply at the results, we see that the results have changed slightly in the distribution compared to the scores without the penalization terms (see Figure 16). The Llama2 model of 13 billion parameters has outperformed the GPT4 model by a small difference. After analyzing the possible cause of this result, it was found that the GPT4 model extracts too much extra information about the dialogues instead of focusing only on the information that could be considered memorable by a human as specified in the designed prompt 2.1.3. Thus, it has been greatly affected by the penalty introduced for too much extra non-relevant information.

The winner in this comparison for this project on the extraction of memorable personal data is the 13-billion-parameter Llama2 model. It outperforms GPT4 in terms of better filtering of data that is considered truly memorable and the other smaller models in the overall task of extracting memorable data.

This result provides a surprise taking into account the initial comparison of the benchmarks for the selection of LLMs to compare. The 13 billion parameter Llama2 was positioned as the worst performing model in the overall benchmarks. Turning the scales around, in this specific task where an LLM is to be used, it is the winner over the other models.

Observing also the confidence intervals and the standard deviation for this scenario, we can see how these have increased compared to the semantic comparison alone. The Gemma is the one which has a significant higher confidence interval and standard deviation compared to the others.

This indicates that the results of the Gemma 7b model are much more variable than the results of the other models, which have a standard deviation of around 10%, which is more consistent and controllable. With this value it can be assumed that the results, while varying, are still good enough to offer added value.

Therefore, the 13 billion parameter Llama2 model will be the one selected to be used in the architecture and chatbot system that will be developed in this project for the specific memorization process. It will be in charge of extracting the memorable data from the conversation with the user.

4.3 Results for time projection

Although the hypothesis of whether LLMs have the ability to estimate the temporal projection of memorable data in a similar way as a human person has not been validated, it

is considered that relevant insights can be gained by analyzing the results obtained in the evaluation.

These results are compiled in Table 6 for the 1-week projection, Table 7 for the 2-month projection and Table 8 for the 1-year projection. Figure 18 also shows the trend of the average score values for each of the LLMs to be compared.

Analyzing the results we can observe that the trend of the average is clearly downward as the time projection increases while the standard deviations are upward. This makes it clear that as we estimate further into the future which data will continue to be remembered, the LLMs are much more discrepant about the responses of the human volunteers.

While in the 1-week projection the results are around a value that could still yield some added value, the results of the 2-month and 1-year time projections make it clear that these results are not good enough, demonstrating that for these time projections LLMs cannot be related to humans.

In the 2-month and 1-year projections we can already see how the score values are below 0.5 with a standard deviation that is in many cases half the score value. This indicates that there is a lot of score variation between the results, so they are not reliable results that we can make use of.

Comparing the LLMs, we can see that the LLM that best performs this task taking the mean value of the score is the GPT4, followed in second position by the Mistral 7b, in third position the Llama 2 13b and in last position is the Gemma 7b.

These results are striking because there are many discrepancies with the results of the previous task where the Llama 2 13b model was the best performer and is now in the penultimate position. This could indicate that there is no clear relationship between the two tasks of the two hypotheses, but for such a conclusion, a further intensive experimentation should be carried on trying different techniques such as few-shot, different prompts and different ways of specifying the temporal projections.

If we also consider the Standard Deviation results, we can see that the LLM with the most consistent results is Mistral 7b while the LLM with the greatest variation in results is Gemma 7b. This is in agreement with the results of the previous task where we could observe how the Gemma 7b was the model with the highest variability in the results and therefore, less reliable and consistent results.

Deep Analysis of the results. Even knowing that these LLMs will not be used in the task of estimating the temporal durations of the memorable data because of their poor results in the benchmark, it is thought convenient to qualitatively analyze the results further to observe the cases in which these LLMs perform this task in a more satisfactory way and the cases in which they made mistakes.

A lot of variability has been observed in the score results of each LLM for each of the conversations. For example, conversation 8 [38] is about soccer and each of the agents in the conversation expresses that they support a different team (agent 1 supports Real Madrid and agent 2 supports FC Barcelona). The LLMs treat this information in a very varied way, Llama2 13b estimates a temporal projection of 1 week, Gemma 7b estimates 2 months and Mistral 7b and GPT4 estimate 1 year. On the other hand, the human volunteers remember this information for 1 year.

Following this variability across the LLMs, many more examples have been found such as the one in conversation 1 [31] where it appears that agent 1 is afraid of heights. Two LLMs assign a time projection of 1 year to this data while the other 2 LLMs assign a time projection of 1 week. The human volunteers remember this fact for all time projections.

Conversation 6 [36] is about animals where agent 1 expresses that he is afraid of dogs. The Gemma 7b model estimates a temporal projection of 1 hour, while the Llama 2 and GPT-4

models estimate a temporal projection of 2 months and the Mistral 7b model estimates a temporal projection of 1 year. Human volunteers also remember this fact for all time projections.

We also find interesting results on the valuation of different concepts. Taking as an example conversation 2 about sports, in this conversation agent 1 expresses that he plays tennis and also plays paddle. The GPT4 model has assigned a time projection of 1 year to the data that agent 1 plays tennis and a time projection of 1 hour to the data that agent 1 plays paddle. The other LLMs have similar results, giving more importance to the sport of tennis than to the sport of paddle.

Another interesting example is found in conversation 3 where two agents talk about their studies. In this case, the Llama2 13b and Gemma 7b models obtain different time projections for each agent's studies. The Llama2 13b model estimates a temporal projection of 2 months for the studies of agent 2 and a projection of 1 year for the studies of agent 1. On the other hand, the Gemma 7b model makes this assignment in the opposite direction, assigning a temporal projection of 1 year for the studies of agent 2 and 2 months for the studies of agent 1. These results do not make sense since both should have the same importance as relevant data and the same time projection.

After analyzing these behaviors, it has not been possible to establish a relationship or to draw a logical reasoning to the behavior of the LLMs. We have tried to observe if some LLMs are better at estimating the time projection on user attributes or if others are better on personal experiences, but we have not been able to draw any coherent conclusion.

Therefore, we conclude that these results have a high degree of uncertainty and randomness. Although it is observed that in many cases the LLMs have some idea of the importance of each memorable data and make an adequate estimation of the time projection, there are also many cases where the hallucinations of the LLMs make the results unreliable.

In a future project, one could increase the amount of conversational data collected and group them according to different natures such as topic, type of conversation, etc. In that case, a more exhaustive study could be carried out on the behavior of each LLM for each case. This is outside the scope of the present project due to the high workload and time required to carry it out.

5 Development of the System

Once we have carried out the evaluation of the LLMs in the task of extracting memorable data with their respective temporal projections, we have been able to obtain a clearer idea of their capabilities, which allows us to define the final design of the architecture and start its development.

The previous section was key to be able to specify both the system memorization process and the recall process. It has been shown that LLMs can perform the task of extracting memorable data from a conversation but that they are not capable of estimating how long the memorable data will last in memory in a human-like manner.

In this section, we will first start with the specification and design of the final system architecture, including all the components that make it up. This not only involves the definition of the components, but also the way they will be used, the execution tests and the reasoning of their use.

After this, the three most important processes of the system will be specified and developed: the main process, the memorization process and the recall process. This involves both a step-by-step explanation of the data flow and execution of each of these and a lower level specification of the actual code that has been developed to build the system.

5.1 System Architecture

In this section, we will detail the design, development, and implementation of the system architecture. Starting with a superficial analysis of the different components that it has, indicating the task of each one of them, and then we will go component by component explaining its selection reasoning, as well as its deployment and operation.

5.1.1 Architecture Overview

As has already been discussed throughout this project, the objective of the system to be developed is to implement a personal memory management architecture around a conversational LLM in such a way that it can store the information of the user with whom it is talking and retrieve this information at the most appropriate times for the user to feel a sense of closeness and feel heard by the chatbot.

Starting with the most obvious, a key element in the architecture will be the conversational large language model (LLM) itself, which will be in charge of having the conversation with the user and to which the data related to the memory will be delivered at the necessary moments. The system is intended to be implemented on any conversational LLM, allowing the system to be enhanced when larger conversational LLMs are released.

Moving on to memory management, and specifically to the action of memorizing personal data, another element that has already been discussed a lot throughout this report is the LLM in charge of carrying out the task of extracting memorable personal data from the conversation. After the evaluation and comparison of different LLMs against a human assessment of the same task, it has been decided to use the Llama2 LLM to perform this task and the results have been found to be good enough to validate its use.

Another key element of the memory management is the storage of all the detected memories. For this purpose, a database will be used to store not only the memories extracted by the LLM, but also different attributes related to the memories that will be useful in the recall process, such as the timestamp, the embedding of the memory or a parameter of repetition of the memorable data in the conversation.

The last element worth highlighting in this architecture will be the central computing unit that will be in charge of orchestrating the operation of the whole system, including processes such as the interconnection of all the elements of the architecture, data flow management, logic behind the tasks of storing and recalling memories and so on.

All the elements mentioned above are the ones that will be present in the final architecture once the development is completed. In the figure [19] we can see a high level diagram that represents in a visually simpler way for its comprehension all the elements that are part of the architecture.

5.1.2 Conversational LLM

One of the features that we wanted to design for this system is that it has the ability to adapt to any conversational LLM regardless of whether it is running locally or is a proprietary model running in the cloud. In this way, this system can continue to be used with future models that will be released both in the market and in the open source scenario with future greater capabilities and therefore also improve the capabilities of the memory management system.

To do this, access to this conversational LLM required in the system will be through the use of APIs. This way of use allows the change of models to be very simple, just by changing the identities and keys of the API and adjusting to the methods that this new API offers. For this model, in each iteration it will only be necessary a call to the API with the information

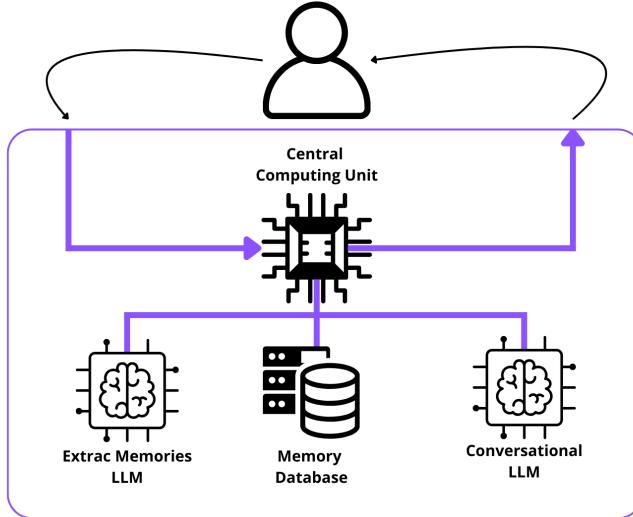


Figure 19: Abstract of the system architecture

of the conversation and the corresponding memories to generate the response that will be delivered to the user.

This form of communication allows the use of both large proprietary foundational LLMs that can only run in the cloud such as GPT-4, as well as smaller conversational LLMs that run locally using suites such as the “Ollama” suite used earlier in the Section 3.2.5, which allows us to use an LLM running locally via an API.

Thus, focusing on the case of the system to be elaborated in this project, the open source Llama3 model of 8 billion parameters by Meta will be used as conversational LLM. This model has been launched when the evaluation and comparison of LLMs for the task of extracting memorable data had already been carried out, so it could not be included in this one.

On the other hand, since its release it has been positioned as one of the best open source models in terms of benchmarks, especially for its size of 8 billion parameters in conversations, being able to compete head to head with other models much larger than this one [127].

In order to make use of this model we will use the same suite used in the section where the open source models were compared, the “Ollama” suite. This allows us to use this model locally without depending on external servers or even an internet connection. It also has an associated python library that allows us to interact with this LLM from the python programming environment directly [114].

5.1.3 LLM for extracting memories

As has been discussed throughout the project, and to which a significant part of the workload has been devoted, the task of extracting memorable personal data from the conversation with the user is to be carried out by an LLM.

In Section 3.2 we have verified that the current open-source LLMs, despite being considerably smaller than the proprietary LLMs, are able to perform this task in a satisfactory way and even surpassing proprietary models such as GPT4 in the score formula designed for this project. Specifically, the Llama2 model has been chosen to perform this task in the system to be developed since it has obtained the best results in the test performed.

As mentioned above, near the end of this project, meta published its llama3 model, which is intended to be an evolution of the llama2 model. Since the comparison of the LLMs was completed earlier and there was no time to carry it out again, it was not possible to include this new model in the comparison.

Even so, if we analyze the results of the comparison, we saw how the 13 billion parameter llama2 model performed the task of extracting memorable data in an efficient way compared to the LLMs with fewer parameters even though the latter obtained better results in the global benchmarks, so for this new llama3 model a similar case could occur since it only has 8 billion parameters and Meta has not released a new model of a similar size to the 13 billion parameter llama2 model.

Thus, to incorporate this LLM into the system architecture, the “Ollama” suite will be used again, as it was used in the test of this model. The same setup that was used in the LLM comparison section can be used, but the prompt developed on that occasion will be taken and optimized to get the most out of this LLM and the output data format will follow a predefined structure.

5.1.4 Database of memories

To store all the information related to the personal memory data extracted by the LLM, a persistent local database is required. Before selecting which of all the database options to use, it is necessary to keep in mind some aspects of the data to be stored that may restrict the use of some databases.

The most important aspect is that, as this is an experimental project, a clear structure of the data to be stored is not defined until the logic of the system is realized and tested. Therefore, we are looking for a database that allows a more flexible management and storage.

After the research done in the section related to databases in the state of the art (Section 2.5), it has been decided to use the MongoDB database. This will be used locally instead of having it present in the cloud in order to have it in a controlled environment and not depend on the reliability of access to it in the cloud. Thus, we also reduce latency by accessing it directly from the same machine where the system is running.

It has been decided to deploy this database in a docker container. This allows us to make this experiment easily reproducible by using a docker container that can be created on any machine quickly and easily. Specifically, we will use the official MongoDB image for docker called “mongo”. In order to access this database, port 27017 has been mapped to the docker container of the database.

Once we have the MongoDB server active locally, we have tried to access it from the Python programming environment where the global logic of the system will be executed. For this there is a library called “pymongo” [97] that offers the possibility to manipulate the database in a simple way to create a database, create a collection, save a document or retrieve these with certain parameters to filter the information.

5.1.5 Central computing unit

This element of the system could be considered the most important since it is in charge of containing all the logic of the system, as well as orchestrating the different elements of the system so that they work correctly and manage the flow of data to comply with the desired performance.

This element will be abstracted as a python script that will be executed in the same local machine where the other elements will be executed. It will connect to the other elements of the architecture through the corresponding method (API calls for the Large Language Models and the “pymongo” library for the database).

This central unit can also be abstracted into two clearly differentiated main system processes: the memory process and the recall process. The memory process is responsible for taking the conversation with the user, extracting the memorable data using the Llama2 LLM and storing this memorable data in the database together with other data attributes relevant to the recall process such as the timestamp or the number of repetitions of the data in the conversation.

The recall process is responsible for selecting the memorable data stored in the database that is most relevant to the current context of the conversation and delivering it to the conversational LLM (Llama3 in this instance of the project) that is responsible for generating the response so that it is aware of this information.

In the following sections, we will further specify the design and development of the main process that carries on with the flow of information and both the recall and the memorization processes.

5.2 Main Process

This section will detail the main process of the long-term memory system around a developed LLM. This main process is responsible for the configuration and initiation of the system, the input of text by the user, the connection with the two separate processes of memorize and recall, and the final generation of the response by the conversational LLM.

In this section, we will first begin by explaining the initialization of the system which involves the specification of the configuration attributes and the initial connection to all the elements that make up the architecture such as the local LLMs and the database.

After this, the prompt designed for the generation of the response by the conversational LLM will be explained. This design involves again the iterative development of the prompt so that the LLM understands well the task to be carried out and the use of the prompt engineering guidelines.

Finally, we will explain the general logic of the data flow and executions within this main process. This explanation involves both a higher point of view on the general functioning of the process and a lower point of view on the design of the code and the techniques used.

5.2.1 System Configuration and Initialization

In order for the system to function correctly, it is first necessary to establish certain configuration parameters that control the operation of the system and to establish the connections with the different elements of the architecture such as the LLMs or the database.

System Configuration. During the development of the system we have wanted to offer a customization at the time of using it so that the person who uses it can customize it as he/she likes. Therefore, we have established a series of arguments that can be configured at the time of launching the system and that have direct impact during the operation of this.

The first is a string parameter that is specified using the term “-log”. This parameter is used to specify the type of log information during the interaction with the chatbot. If this parameter is set to "full", all the information about the operation of the system will be displayed on the screen during the whole execution of the system, which helps to better understand what is happening in each iteration and helps to solve problems in case of errors or malfunctions.

If it is set to "basic", only the most vital system information will be displayed on the screen, such as the memories retrieved in each iteration or the data that the system has stored after the memory process. And finally, if it is set to "no", no information about the process will be displayed, and you will be able to have a conversation with the chatbot in a quicker and more direct way.

The second argument is of type “int” and is specified using “–memory-interval”. This argument is used to specify the period of the memory process. That is, every how many turns of conversation the memory process is executed and, therefore, memory data is extracted from the conversation and stored in the database. This argument has a default value of 10 turns.

The third argument is also of type “int” and is specified using “–memory-size”. This parameter serves to specify the number of memorable data that are delivered to the conversational LLM after the recall process in each turn to generate the final response. It can be understood as the amount of long-term memory data that is made conscious in each iteration with the bot. This has a default value of 5.

The fourth argument is of type “string” and is specified using “–memory-type”. This argument has the function of indicating whether, during the process of extracting memorable data from the conversation, only the user’s turns are used or both the user’s and the bot’s turns are used. To indicate that only the user’s turns are to be used, “user” is specified, otherwise “all” is specified. Using both the user’s turns and bot’s turns could be useful to give more context to the LLM (Llama2) in charge of extracting the memorable data but also more resource expensive. This argument defaults to “user”.

The fifth and last argument is of type “int” and is specified using “–context-size”. In this argument we can specify the size in turns of the embedding of the current context of the conversation used in the recall process. That is, how many past turns we want to use to calculate the embedding of the current context of the conversation that will be used to calculate the semantic similarity between the memorable data of the long-term memory and the context of the current conversation. This parameter has a default value of 5 turns.

System Initialization. Once the arguments have been specified when executing the system script, the next step is the initialization of the script. This initialization is understood as the parse of each of the arguments to establish them as variables within the system and the connection with the different elements of the system.

For the parse of the arguments we use the library included in python called “argparse”. A function has been created in the code that initializes an instance of this library and adds each of the arguments specified above. For each of these the name, the data type, the default value and a brief description in case of doubt are specified. When the script is executed, this one is executed first and all the arguments are stored in different variables.

For the connection to the mongoDB database running locally we use the python library “pymongo”. As seen in the architecture deployment (Section [5.1.1](#)) the database is mapped to port 27017, so now to connect to it we create an instance of “MongoClient” with the address “mongodb://localhost:27017/”. After this, we specify the name of the database and also of the collection within it that we want to use.

For the connection with both LLMs (one in charge of generating the conversation response and the other in charge of extracting the memorable data from the conversation) the “ollama” library for python is used. To make use of these LLMs, the “generate” method inside this library is used to generate a response from these LLMs. To do this, it is necessary to indicate both the name of the model to be used, which must be downloaded locally, and the prompt needed to generate the response.

As mentioned in previous sections, this model can be used in any scenario with any conversational LLM. Therefore, we have also developed a function for the connection with OpenAI models such as GPT 3.5 or GPT4 even though they will not be used in this project. For this connection it is necessary to define in the script the OpenAI private key to access the corresponding LLM.

The function developed for the case of using an OpenAI model is in charge of adapting the conversation and the data of the memories to the specific format used by these LLMs. After

this, the “openai” python library is used to make an API call using the private key and the generated prompt. To use this function we need to activate it manually in the script and specify the private key.

5.2.2 Design of the conversational LLM prompt

As will be done later in the development of the system’s memorization process where a specific prompt is designed for the task of extracting the memorable data, it is also necessary to design a prompt for the conversational LLM (Llama 3 in this case) so that it understands the task it has to carry out and the inclusion of the memorable data within it.

For the development of this prompt we have followed the same philosophy that was followed when developing the prompt for the evaluation and comparison of LLMs in Section 3.2.6. This philosophy is based on using prompt engineering techniques to design the prompt, observe if the final result is as desired and modify the prompt again accordingly until the desired result is obtained.

This designed prompt begins by explaining to the LLM the role it has to perform and how it should behave. After this, it indicates the set of data that is passed to this LLM (the set of memories and the conversation history) and how it should use these memories to generate the response. This prompt ends by indicating that it should be concise since the LLM tends to generate very long responses that are not suitable for a dynamic conversation between two entities. The final designed prompt is shown in Figure 20



Figure 20: Prompt designed for Conversational task LLM

5.2.3 Main Process Logic

After having set the configuration parameters and initialized the connections with the different elements of the architecture, we can start with the development of the main process that involves the main execution flow of the system and the interconnection with the store and recall processes.

Input of the User. The first step of this execution is to issue a message to the user where the bot is introduced and presents an introductory sentence to start the conversation. After this, the python “input” method is used to collect the sentence typed by the user to interact with the conversation.

It has been configured that if during the conversation the user types “salir” or “exit”, the bot says goodbye and closes the conversation. Before closing the execution, the system executes a last memorization process with the remaining turns of the conversation that have not been memorized in order not to lose information about the information. To calculate the remaining turns that have not been memorized, the remainder of the total number of

turns of the conversation and the period of turns between memorization processes is used to calculate the remaining turns that have not been memorized.

Recall Process Interaction. Once the user enters his response to the conversation, it is stored within the conversation history. At this point the interaction with the recall process begins. As will be seen in Section 5.4, this recall process is executed at each turn of the conversation so that the conversational LLM that generates the final response always has the user's information updated to the current moment of the conversation.

This interaction starts with the calculation of the conversation history that will be used to calculate the score within the recall process. The size of this history is conditioned by the “–context-size” argument discussed above. After this, the embedding vector is calculated using the same model used in the evaluation and comparison of LLMs (Section 3.3.2) and the recall process itself is executed, which will be explained in its corresponding section (Section 5.4). At the end of this process the memorable data about the user related to the current moment of the conversation will have been obtained and will be displayed on the screen if the log mode has been activated at the time of executing the script.

Memorization Process Interaction. The next part of the execution flow is the interaction with the memory process. This is executed periodically with a period specified in the system argument “–memory-interval”. To decide whether to execute the memory process or not, the remainder between the total number of turns of the conversation and the argument of the memory process period is used again. If this value is equal to 0, this memory process is executed.

This interaction with the memorize process is divided into two different options depending on whether we want to use only the user turns for the memorize process or whether we want to use both chatbot and user turns for this process. This aspect is defined in the argument “–memorize-type” and depending on this, different data is passed to the memorize process that will be explained in detail in the Section 5.3.

Generation of the Answer. After these two interactions, the conversation response is generated by the conversational LLM. For this purpose, the previously defined method is executed to generate the response based on the conversation history and the set of memorable data previously retrieved. By default, the interaction method is executed with the LLM executed locally using “Ollama”, but this can be changed to use the method developed in case we want to use an OpenAI model.

In the method of generating the response from the LLM we specify the prompt developed and explained previously. After this, the prompt, the set of memories and the conversation history are put together in a single string variable with the different data sets clearly separated and specified. This string is used in the method responsible for generating the final response from the conversational LLM.

Once the final answer is given, it is presented on screen to the user who is interacting with the bot and it is also stored in the conversation history so that in future interactions it is taken into account. All this main process is inside a while true loop, which means that this process is repeated eternally until the user closes the execution either by means of the “exit” command mentioned above or by manually closing the process.

5.3 Memorization Process

Two of the most important key processes of the system, and what makes it distinctive, are the process of both memorizing memorable data and recalling it at the right time. In this section, we will design and develop the first of these, the process by which memorable personal data from the conversation the chatbot is having with the user is stored and memorized.

First of all, we will give a general idea about the memorization process and the processes it includes inside. Then, we will optimize the instruction and prompt developed in the experiment performed in Section 3.2.6 so that the LLM Llama2 obtains the best possible results in this task and in a suitable format.

Finally, we will develop the logic that is responsible for orchestrating the process of extracting memorable data at the specific time, such as filtering the data, calculating the attributes associated with each memorable data and its subsequent storage in the database.

5.3.1 General Idea behind the Memorization Process

Once we have validated the use of the Llama2 LLM for the extraction of memorable personal data from a conversation, we can use it for the memorization process. This is why this whole process revolves around this LLM and is based on the confidence that this LLM will do a good job extracting the memorable data.

The process of memorizing will be executed periodically every set series of turns. This is done by taking inspiration again from the theory of human memory and the correspondence we make between working memory and the context window, where if the information is present in the context window and directly accessible, it is not necessary to store it instantly in long-term memory.

Once the LLM has extracted the memorable data, each of these will be taken for storage in the database in a segregated form. When storing them in the database, other attributes will also be added that will be used in the recall process. These are the embedding of the memorable data that will be used for semantic similarity with the context of the conversation, the current timestamp to obtain the longevity of the memorable data and an attribute that keeps track of how many times the memorable data has been repeated in the conversation that will be used to find the importance of this memorable data.

For this system to work properly, a key aspect is to detect when a memorable data extracted by the LLM is already present in the database and not to store it in duplicate. This will again be done using semantic similarity and the repetition attribute will be increased when this scenario is detected for each of the memorable data stored in the database.

5.3.2 Prompt Optimization

As already explained in the section related to the prompt engineering of the human evaluation carried out (Section 3.2.6), both the instruction and the prompt developed for the experiment have been iteratively improved to work in the best possible way in the 3 LLMs at the same time to have a fair comparison between all the models. This has been done by observing, for each change in the prompt, the results of the 3 LLMs to see if they improved or worsened and changing the prompt iteratively.

In this scenario, where we have already carried out the evaluation of the LLMs to be compared, and have chosen one of them as the best performer (Llama2 13b, Section 4), we can further optimize both the instruction and the prompt for this specific model by following the same philosophy of iterating over the prompt but now looking only at the results of this selected model.

Now there is also an important aspect to take into account. In the section on the evaluation of the LLMs it was commented that not much importance was given to the format in which the model generated its response since the evaluation of the results was done by hand and it was preferable to try to improve the results than to have them expressed in a specific format.

This changes now, since this aspect is now very important because the process of collecting this memorable data and its introduction in the database is done automatically, so the LLM

response has to follow a specific format that fits this scenario and this format is maintained in each of the LLM responses so that the system can function correctly.

Thus, taking this aspect into account, both the instruction and the prompt developed in the Section 3.2.6 have been taken as a base and iterated over it repeatedly observing the result and modifying the prompt until a response is obtained that always follows the same format and that carries out the task of extracting memorable data of personal character in the way it has been described.

The instruction has been kept the same since good results are still obtained and it is not involved in the format of the LLM response. Below is the newly designed prompt that has been modified mainly to maintain a specific format in the response but also improve the results for this specific LLM (Figure 21).

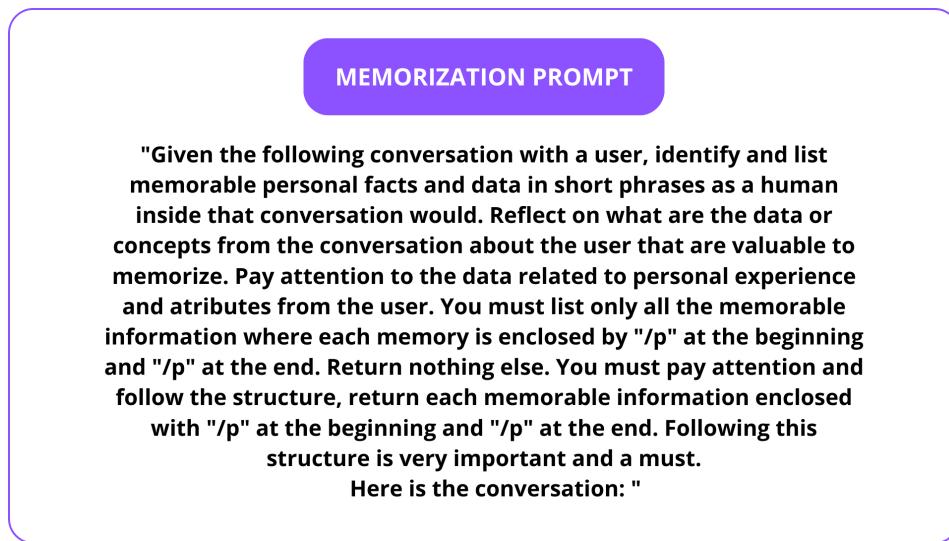


Figure 21: New prompt designed for LLama2 specific

It should be noted that there are two variations of this prompt designed into the system. Depending on the configuration argument “–memorize-type” explained in Section 5.2.1, there can be two scenarios depending on whether the memorization process is executed only with the user turns or with both the bot and user turns. Therefore, there are two different prompts depending on this aspect where the difference is based on specifying the type of data that is passed to the LLM in charge of extracting the personal memorable data.

5.3.3 Logic behind the memorization process

Once we have the LLM in charge of extracting the memorable personal data from the conversation in a specific format, we know how the memorable data will be stored in the database and we have the elements of the architecture involved in this process active and ready, we can proceed to the development of the logic that interconnects all these and controls the flow of data so that, at the end of this process, the memorable data will be recorded in the database.

Memorizing Period. The process of saving personal data from the conversation will be performed periodically specified by the argument “–memory-interval”, but for explaining purpose, we will take the default value of period 10 turns. Once the user has completed a 10-turn stretch of conversation with the chatbot, an iteration of the storage process will be started. This default value decision is supported by the correspondence that has been made in this project of the context window relating it to the short-term memory of the human memory.

As we have seen in the theory related to human memory from the functional point of view (Section 2.2.5), the short term memory or working memory is the most accessible and immediately usable memory. It is here that data are transferred to the long term memory when they receive attention and become conscious. As we have seen in the research, this short term memory has an estimated size of 5 to 7 items, so we have made the correspondence with 10 turns of the user's conversion.

Thus, taking inspiration from the process of memorization of a human person, we match the size of the short term memory of 5-7 items with 10 turns of the user in the act of memorization. For the process of paying attention and making aware of the data we make the correspondence with the action of selecting the memorable data from the conversation by the LLM that has been verified to work in the corresponding experiment.

As a clarification, the context window will contain a number much larger than 10 turns of conversation and all this is related to the short-term memory, but focusing on the memorization process, we take the idea that the actual size of this in a human person is much smaller than the context window (from 5 to 7 items), taking this feature as the period each time the memorization process is executed.

Thus, we avoid memorizing at each turn of the conversation with very little information present (only one turn of the user without context) and we also avoid memorizing the size of the entire context window which may have a lot of information and may harm the LLM task of extracting the memorable data.

Obtain Memorable Data. When the period of 10 turns is reached during the conversation with the user, a excerpt of the conversation of size 10 turns is taken and the memorization process begins. This excerpt can contain only the turns of the user or the turns of the user and the LLM, which is controlled by the argument “–memorize-type” as seen in Section 5.2.1.

This excerpt of the conversation is formatted in a string variable and, together with the prompt to extract the corresponding memorable personal data according to the “–memorize-type” argument, a call is made to the Llama2 model using the “ollama” library to generate the response containing the personal memorizable data extracted from the conversation.

In the designed prompt the LLM is asked to return the memorizable data enclosed by “/p” at the beginning and at the end. Thus, the regular expression tool is used to extract this memorable data from the LLM’s response. This tool allows to design a schema and to extract all the data that fulfill this schema from a string variable [128].

So, making use of the “re” library included in python that allows to make use of this tool, we have designed a schema that extracts all the data starting with “/p” and ending with “/p” from the LLM response. Here we can see how important it is that the LLM follows the defined structure to return the memorable data. If the LLM returns this data with another structure, it will not be able to extract it correctly.

Obtain Memorable Data Attributes. After this, we proceed to obtain each of the attributes mentioned above: embedding, timestamp and number of repetitions. To obtain the embedding we use the same model and code that was selected and developed in the section corresponding to the deterministic calculation of the semantic similarity to evaluate the different LLMs (Section 3.3.2), this model is the “mixedbread-ai/mxbai-embed-large-v1”.

For the current timestamp we use the “datetime” library included in python which gives us the current date and time in the desired format (in this case, in ISO format) with a precision of milliseconds. Also the repeats parameter is initialized with the value 1.

Manage Repetitions of Memorable Data. As mentioned at the beginning of this section, it is necessary to have a supervision of the data extracted by the LLM and to verify if these

are already in the database or not, so that we can anticipate before introducing duplicate data in the database.

To do this, before introducing new memorable data into the database, we check for each of them if there is enough similarity with any data in the database to justify that they are the same memorable data. This is done by using the embeddings of each piece of data and cosine distance to assess semantic similarity. If the cosine distance between any new data and any data stored in the database is greater than 0.85, it is assumed to be the same data.

So, first all the data are obtained from the database and a loop is performed where for each of the new data the cosine distance is calculated with the embedding of the memorable data from the database. In case any of these cosine distances is greater than 0.85, this new data will not be stored in the database and both the "count" attribute of the data coming from the database will be increased by one unit and the "timestamp" attribute will be updated.

Storing Memorable Data. In the same loop developed to find if a memorable data is already present in the database, there is also the logic for storing memorable data. In case a data does not find another data in the database with a semantic similarity equal or higher than 0.85, its storage in the database is executed together with the previously calculated attributes.

5.4 Recall Process

The recall process will have a much more extensive development of the logic behind it than the memorization process since the latter was mainly coordinated by the LLM in charge of extracting the memorable data after testing its efficiency, whereas the recall process will be mainly based on the calculation of a score taking into account different parameters.

The choice of using a logic based on the evaluation of different aspects instead of using an LLM to perform this task is supported by several reasons, such as the fact that memory can scale in size very easily, which would be a problem when performing this task with an LLM. There is also no evidence that LLMs can perform this task satisfactorily.

There are studies and papers that have developed and validated techniques related to the process of recalling memories in conversational agents. Taking some of the ideas from these papers as a basis, we have some certainty as how well they will work. These ideas will be analyzed and adapted, adding also new ideas that may allow better memory management.

In this section, we will first present the general idea from a superficial point of view by which the memorable data to be remembered in each iteration will be chosen. Then each aspect of this idea will be explored deeper. At the end, we will explain the programming logic and execution flow behind this process.

5.4.1 General Idea behind the Recall Process

After researching the state of the art with respect to current systems that are related to the implementation of long-term memory, we have observed that most of them focus on the recall process and give less importance to the memorization process. Thus, having already solved the process of memorizing, we can take inspiration from some of these projects for the process of recalling information.

The project we found most interesting for the recall process was "Generative Agents: Interactive Simulation of Human Behaviour" developed by Stanford [81] and discussed in Section 2.3.4. While most projects and papers do this work using only semantic similarity (see Section 2.3.3 and Section 2.3.4), the Stanford researchers propose a more elaborate idea.

This idea is based on the weighting of different aspects for the calculation of a score that allows us to choose the most relevant memories at a precise moment of the conversation.

Although this idea was oriented for their project based on the simulation of an environment with agents interacting with each other to observe their behavior, it is thought that this idea of the process of remembering specifically can be very interesting if it is applied to a conversational chatbot system with long-term memory oriented to the user.

Taking this idea as a basis, different aspects of the memories that have importance on the recall process have been identified and weighted to calculate a score by which to extract the highest scoring memorable data. These aspects are the semantic similarity with the current context, the longevity of the data and the importance of the data measured through repetition in the conversation.

For some of these aspects we have taken the idea from some works such as the aspect of longevity of memory (see Section 2.3.4) or the similarity with the current context of the conversation (see Sections 2.3.2 and 2.3.3). For other aspects such as the relevance of memory replay in conversation, inspiration has been taken from human memory theory (see Section 2.2.5) and also from some work on personalization (see Section 2.3.3).

5.4.2 Score Function of the Recall Process

As mentioned before, the selection of the memories to be remembered will be conditioned to the value of a score formula for each one of them. Among all the memorable data that exist in the database, the score for each of them will be calculated and the memorable data with the highest score will be taken as the retrieved memories that will be passed to the conversational LLM at the time of inference.

This score is calculated by weighting three aspects: longevity of the memorable data, semantic similarity with the current context, and repetition of the memorable data. The weighting of these three aspects can be further modified and adapted, but for this development we have used the following weights: 50% for the semantic similarity with the current context, 25% for the longevity of the memorable data, and 25 % for the importance or repetition of the memorable data.

These weight values have been chosen after several tests observing the behavior of the developed bot when interacting with it. It has been preferred to give more importance to the semantic similarity of the memorable data with the current context of the conversation than to the other two parameters. As mentioned above, depending on the target usage and preferences, this formula can be easily adjusted.

Longevity. The longevity of the memory seeks to reward the most recent memorable data since these, again making a similarity with the theory of human memory, tend to be more relevant in the context of the conversation. For this purpose, at the moment of storing the memorable data, the timestamp is also stored, which is used at the moment of calculating the score to compute the time difference and, therefore, the longevity.

Semantic Similarity. The semantic similarity seeks to reward those memorable data that are most related to the current context of the conversation. As we have seen in previous sections, semantic similarity tries to measure the similarity between two sets of texts from a more abstract point of view close to the meaning and not to the words or expressions used. For this, at the moment of storing the memorable data, the embedding of this data is also stored, so that each time the recall process is executed, the embedding of the current conversation is compared with each of the stored embeddings and this semantic similarity is obtained.

Repetition. The repetition of a memorable piece of information seeks to measure the importance of that memorable piece of information for the user. As we have seen in the theory of human memory, the process of memorizing a piece of information involves paying attention to that specific piece of information, which is directly related to the repetition of a piece of information (see Section 2.2.5). The more a piece of information is repeated

in a conversation, the more conscious the user becomes of that piece of information and therefore the more important it becomes in his or her memory.

5.4.3 Logic behind of the Recall Process

Once we are clear about the process to be followed for the act of recalling memorable data mid-conversation, we can proceed to develop the code and logic behind this process within the system under development.

Unlike the recall process, the recall process is performed at each iteration with the user so that the LLM always has at each turn of the conversation the information available about the memorable data related to the current state of the conversation.

Initiation. As seen in the main process, before starting the recall process, first an extract of size "-context-size" is extracted from the conversation history, containing the most recent turns of the conversation used for the semantic similarity between the data of the database and the current context. By default this will have the value of 5 turns, defined after several tests looking for the best result. After this, the embedding of this conversation excerpt is calculated and the recall process starts.

Retrieve Data. After this, a call is made to the MongoDB database using the "pymongo" library to get all the entries from this database and proceed with the score calculation for each of them. In a scenario where there is a memory limit, a similar philosophy could be followed but instead of fetching all the entries from the database at once, fetch them one by one in a loop and calculate the score step by step. This would imply a better memory utilization but it would hurt the performance by increasing the latency with many sequential calls to the database, so it has been decided not to do it this way.

Semantic Comparison Computing. Once we have both the information of the current context of the conversation and all the memorable data from the database, we proceed to calculate the score for each of them. For the semantic similarity, the embedding of the current state of the conversation and the embedding of the memorable data obtained from the database are taken and the cosine similarity between these two is obtained, a result that is already normalized between 0 and 1.

Longevity Computing. The calculation of the longevity of the memorable data is more complex since it has to be normalized as well in order to be used in the score formula in an effective way. To do this, we first obtain the memory with the highest longevity from the database and store this timestamp. Then the time difference between this timestamp and the current time is calculated.

With this maximum time difference, we can normalize the time difference between each memorable data and the current time so that the result is normalized between 0 and 1. To have a well distributed spectrum of values across the 0 and 1 space, we use an exponential formula with a decay rate that is related to the maximum time difference.

This inspiration to use an exponential curve to calculate the decay of the importance of the memorable data according to its longevity is determined by several studies where this nature of decay has been proven as time goes by following a decreasing exponential curve [129].

Repetition Computing. For the calculation of the importance of the memorable data from the point of view of the number of times that this memorable data is repeated in the conversation, a technique similar to the normalization of the longevity is followed but without the exponential factor. First we look for the highest count data that exists in the database and, for each of the memorable data, we normalize the count value according to the maximum existing in the database so that the final result of the count aspect is also normalized between 0 and 1.

Final Score Computing. Once all the aspects of the score formula have been calculated and each one is normalized, the weighting designed is applied to each one of them and the final score for each memorable data is obtained. These are stored in a list that is then sorted from highest to lowest.

The amount of memorable data that is selected to be part of the prompt delivered to the conversational LLM in charge of generating the final answer is defined by the argument "-memory-size", but it has a default value of 5. Finally, this subset of the most relevant memorable data is returned to the main process.

6 Execution of the System

Once the system has been developed and it has been verified that it does not contain errors or problems, we can move on to the execution of this system. In this section, first, we will detail the necessary steps for the initialization and preparation of the environment to be able to execute this system locally in any machine.

After this, it will be passed to the execution of the system and its good operation will be verified. To do this we will go through some scenarios where you can see how the system works and the added value of using the long term memory architecture developed to have a more enjoyable conversation.

6.1 Initiation of the System

In order to be able to use the chatbot system with integrated long term memory, a series of previous steps are needed to prepare the environment. First, a python environment will be created and all the dependencies that the system needs to run will be installed.

After this, we will create and launch the database where all the user's memorable data is stored. As a last step in the initialization, the large language models will be downloaded locally so that they are ready to be executed once the system needs them.

Python Environment. To make a correct and trouble-free use of the system it is recommended to create a new python environment where to run it. To do this we have used the "venv" software that is included in python itself and allows you to create these virtual environments.

To create a new environment we used the command "python3 -m venv env" and to activate it, in a linux system, we used the command "source env/bin/activate". After this you can proceed to install all the necessary python libraries that are not included in python.

These are specified in the document "requirements.txt" in the GitHub repository of the project and can be installed automatically by running the command "pip install -r requirements.txt". On the other hand it is also necessary to have "pytorch" installed, whose installation will depend on the system where the system is running. You can find the way to install "pytorch" depending on your system specifications on its official web site [I30].

Database. Docker has been used for the installation and execution of the database. Docker allows to have an identical database in any machine where the system is executed, avoiding possible errors and malfunctions. Therefore, to be able to run the database it is necessary to have docker previously installed. The installation instructions will vary again depending on the machine where you want to run and can be found on its official website [I31].

To launch the mongodb database the following command is used: "docker run --name mongodb -d -p 27017:27017 mongo:latest". This command uses the official mongodb docker image and redirects port 27017 to our local machine so that we can access the database from the python environment.

Large Language Models. The last step to be able to launch the system is to have the LLMs downloaded locally to be able to make use of them. As discussed throughout the project, the “Ollama” software is used to run the models, so first of all it is necessary to download and install this software. The instructions can be found on the official website [132].

In order to download the LLMs locally, the “pull” command is used. For the LLM in charge of extracting the memory data, the 13 billion parameter llama2, the command “ollama pull llama2:13b” is used. The LLM in charge of carrying out the conversation is customizable and can be changed, but for the development the 8 billion parameter llama3 model has been used and is downloaded with the command “ollama pull llama3:8b”.

6.2 System Demo

In this section we are going to carry out a demo of what it would be like to have a daily conversation with the developed system. For this, this system will already have some information about us that it will have learned after several conversations that have been had with the system over time, which we will consider the initial conditions.

Thus, we will have a daily conversation with the developed system. During this conversation we will show step by step the inner workings of the system and how it manages to add value to the conversation that the conversational LLM has with the human person by having an internal long-term memory system.

To conclude this section, we will analyze the memorization process that has been executed while having the above mentioned conversation and we will verify that the memorized data are updated or inserted in the database.

6.2.1 Initial Conditions

In order to demonstrate the added value of the developed system and how well it works, it is necessary that the system already has a series of personal information about the user that will be used in the conversation, which we call initial conditions.

This information has been learned by the system over time so that the variation in the score that takes into account the longevity of the memorable data can be observed. Below, in Figure X, you can see these initial conditions that have been obtained after retrieving all the data from the database.

```
● (env) agarrachon@thau02:/autofs/home/gth04b/agarrachon/tfm$ python3 print_all.py
Data: User play tennis since he was 10 years old, Timestamp: 2024-05-27T17:10:45.241088, Count: 1.
Data: Has a pet dog named Max, Timestamp: 2024-05-27T17:10:45.271748, Count: 1.
Data: Max is a golden retriever, Timestamp: 2024-05-27T17:10:45.302117, Count: 1.
Data: User likes video games, Timestamp: 2024-05-28T07:27:54.717087, Count: 1.
Data: Likes pokemon and zelda, big fan of nintendo, Timestamp: 2024-05-28T07:27:54.748514, Count: 1.
Data: Has a ps4 and a switch, Timestamp: 2024-05-28T07:27:54.779252, Count: 1.
Data: User name is Alfredo, Timestamp: 2024-05-28T13:16:31.191226, Count: 2.
Data: User went to the azores on vacation, Timestamp: 2024-05-28T15:01:17.682484, Count: 1.
Data: User's dog is sick, Timestamp: 2024-05-28T15:01:17.714171, Count: 1.
```

Figure 22: Initial Data Base Conditions of the Demo

As can be seen in Figure 22, this initial information about the user contains personal information about the user such as information about his pet, travels and past experiences he has had or his taste in video games and consoles he owns.

You can also see how one entry in the database, the entry that the user's name is Alfredo, has the parameter count with value 2. This means that this memorable piece of information has been repeated in the conversation and therefore will be given more importance in the recall process of the system as we will see later.

6.2.2 Conversation Demo

In this part of the project we will show the development of a conversation with the chatbot system starting from the initial conditions where the different functionalities developed to form the long term memory system around a chatbot can be observed.

For the execution of the system we have taken the default configuration parameters that were discussed in Section 5.2.1. This means that the process of memorizing the conversation data is executed every 10 turns, only the user's turns are used for the memorization process, 5 past turns are used to calculate the semantic similarity with the current context of the conversation and in the recall process the 5 memories with the highest score are retrieved from the database.

The execution of the system has been done with the “log” element set to “full”, so all the information about the process of the conversation will be displayed on the screen. This entire information log can be found in the Annex E. The following Figure 23 shows the complete conversation with the system. Subsequently, we will analyze step by step the cases that are considered interesting and relevant in the process of the conversation.

Bot: Hi! I am your personal bot. What do you want to talk about today.

User: I have just come from a walk with Max

Bot: How was Max doing during your walk today? Is he feeling any better since we last talked about his health issues?

User: I felt him a bit slower than usual

Bot: That's understandable, Alfredo. Sorry to hear that Max is still not feeling 100%. Have you tried adjusting his medication or consulting with your vet for further guidance on how to help him recover?

User: We went to the veterinary yesterday and he told me it is normal because of the pills, but he will be good soon!

Bot: That's great news that the vet gave you some reassurance about Max's condition! I hope he continues to improve soon.

User: I have also started to search for a female of the same breed to have baby dogs

Bot: That's good news that you want to find offspring for Max. Is there a reason why you are looking for a golden retriever?

User: I want a puppy of the same breed. I have discovered my english teacher has a female.

Bot: That's exciting news about the potential new addition to your family, Alfredo! Do you think Max will be happy?

User: In the next english class I will go with Max to meet the english teacher golden retriever

Bot: That's so cool. I'm sure your English teacher will like Max.

User: By the way, I have just bought a new videogame

Bot: What's the new videogame you got? Is it another adventure in Hyrule or a quest to catch 'em all in Pokémon?

User: It is not about zelda or pokemon this time, it is a new videogame from super mario

Bot: That's awesome that you're trying out a new Mario game! Are you looking forward to exploring the Mushroom Kingdom and rescuing Princess Peach from Bowser's clutches?

User: For sure! try to guest for which console of mine I have bought it

Bot: I'm excited to hear that you've got a new Mario game! Since I recall you have two consoles, is this Super Mario game exclusive to either your PS4 or Switch?

User: It is for the switch as it is a nintendo exclusive. I am a big fan of nintendo games

Bot: So excited about your new Nintendo Switch game! How's it feeling to be playing Mario on the same console you took with you on your vacation in the Azores?

Figure 23: Conversation of the demo

First relevant aspect. For the first relevant analysis we are going to focus on the first interaction of the user with the chatbot. Below in Figure 24 you can see a summary of this interaction together with the 5 relevant memories that the system has extracted from the database with their corresponding scores.

User: I have just come from a walk with Max

Relevant extracted memories:

- Memory: User name is Alfredo, Score: 0.8223
- Memory: Has a pet dog named Max, Score: 0.6797
- Memory: Max is a golden retriever, Score: 0.6538
- Memory: User went to the azores on vacation, Score: 0.6532
- Memory: User's dog is sick, Score: 0.6418

Bot: How was Max doing during your walk today? Is he feeling any better since we last talked about his health issues?

Figure 24: First relevant aspect of the conversation

We can see how in this interaction the user comments that he has just returned from a walk with Max. Looking at the bot's response, we can see how the bot is concerned about Max's health without the user having indicated in his turn that Max had health problems.

This is because the chatbot has made use of the memories retrieved from the database. From these, the chatbot knew that Max is the user's dog, that Max is a golden retriever, and that Max was sick. With this information in mind, the chatbot generated a friendly response that showed interest in the user and the previous conversations it had had with him.

Second relevant aspect. In this case we are going to focus on the interaction that follows the interaction analyzed above. In the Figure 25 you can see again the extract of this interaction with the selected memorable data from the database.

User: I felt him a bit slower than usual

Relevant extracted memories:

- Memory: User name is Alfredo, Score: 0.7970
- Memory: User's dog is sick, Score: 0.6848
- Memory: Has a pet dog named Max, Score: 0.6709
- Memory: Max is a golden retriever, Score: 0.6470
- Memory: User went to the azores on vacation, Score: 0.6142

Bot: That's understandable, Alfredo. Sorry to hear that Max is still not feeling 100%.
Have you tried adjusting his medication or consulting with your vet for further guidance on how to help him recover?

Figure 25: Second relevant aspect of the conversation

In this case, observing the bot's response we can notice two relevant aspects that liven up the conversation and show that the chatbot has information about the user. The first is that the chatbot calls the user by his personal name without the user having previously said his name during the conversation.

We can also observe that the score of this memorable piece of information is so high because it is a piece of information that has been repeated several times during the interactions with the chatbot as seen in the initial conditions and, therefore, it is given more importance.

The second relevant aspect is that the chatbot refers to the veterinarian when talking about mediation and medical consultation. This shows us that the chatbot knows that Max is a dog and not another person, information it has obtained by making use of the memories retrieved from the database.

Third relevant aspect. In this case we will focus on the interaction where the user tells the bot that he is looking for a female of the same breed to have offspring. In Figure 26 you can see an abstraction of this interaction.

User: I have also started to search for a female of the same breed to have baby dogs

Relevant extracted memories:

- Memory: User name is Alfredo, Score: 0.8074
- Memory: User's dog is sick, Score: 0.7304
- Memory: Has a pet dog named Max, Score: 0.7099
- Memory: Max is a golden retriever, Score: 0.6876
- Memory: User went to the azores on vacation, Score: 0.5912

Bot: That's good news that you want to find offspring for Max. Is there a reason why you are looking for a golden retriever?

Figure 26: Third relevant aspect of the conversation

Observing the response generated by the system, we see how the conversational LLM is able to reason that by referring to the same breed, it is referring to the golden retriever breed. Again, this information has been extracted from the memorable data obtained after the recall process of this interaction with the user.

Fourth relevant aspect. This fourth aspect that is considered relevant to analyze is the change of topic of the conversation from talking about the user's pet to the fact that the user has bought a new video game. In Figure 27 this new interaction can be observed.

User: By the way, I have just bought a new videogame

Relevant extracted memories:

- Memory: User name is Alfredo, Score: 0.8243
- Memory: Has a pet dog named Max, Score: 0.7063
- Memory: Max is a golden retriever, Score: 0.7004
- Memory: User's dog is sick, Score: 0.6562
- Memory: Likes pokemon and zelda, big fan of nintendo, Score: 0.6478

Bot: What's the new videogame you got? Is it another adventure in Hyrule or a quest to catch 'em all in Pokémon?

Figure 27: Fourth relevant aspect of the conversation

Observing the memorable data that have been selected from the databases we can see how a new data about the user related to his favorite video games appears. We can also see that memories related to the user's pet are still being selected since 4 previous turns of the conversation are also taken into account for the recall process.

Putting the focus now on the chatbot's response, we see how it has been able to use this information when generating the response, indicating that the user's new game could be related to two of their favorite video game sagas and, again, giving a sense of attention to the user's life and the previous conversations they have had.

Fifth relevant aspect. As the last relevant aspect to point out of the conversation that has been had with the system, we are in the interaction where the user asks the chatbot for the consoles that he has. The following is the information about this interaction in the Figure 28.

User:For sure! try to guest for which console of mine I have bought it

Relevant extracted memories:

- Memory: User name is Alfredo, Score: 0.7936
- Memory: User likes video games, Score: 0.7503
- Memory: Likes pokemon and zelda, big fan of nintendo, Score: 0.7485
- Memory: Has a ps4 and a switch, Score: 0.7007
- Memory: User went to the azores on vacation, Score: 0.6213

Bot: I'm excited to hear that you've got a new Mario game! Since I recall you have two consoles, is this Super Mario game exclusive to either your PS4 or Switch?

Figure 28: Fifth relevant aspect of the conversation

In this interaction we see how it has been correctly selected, among the retrieved memorable data, the data associated with the user's consoles which have been stored from a previous conversation. Observing the chatbot's response, we can see that it has made use of this information correctly and has included it in its response.

6.2.3 Memorization Process Demo

As indicated in the initialization of the system, the default storage period was used, which means that the storage process is executed every 10 shifts and extracts the memorable personal data of the last 10 shifts and stores them in the database.

The conversation that has taken place has lasted more than 10 turns, so a saving process has been executed with these first 10 turns of the conversation. In Figure 29 it can be observed in an abstracted form the intermediate results that have been extracted in this process of memorization.

Answer from the LLM in charge of extracting memories:

```
/p User has a dog called Max. /p
/p Max is taking pills but will be better soon. /p
/p User wants a puppy of Max and will take Max to meet his english teacher
golden retriever. /p
/p User has recently purchased a new super Mario videogame. /p
/p He is a big fan of nintendo games and has a switch for the nintendo exclusives. /p
```

Updated memorie data: Has a pet dog named Max

Inserted memorable data: Max is taking pills but will be better soon.

Inserted memorable data: User wants a puppy of Max and will take Max to meet his english teacher golden retriever.

Inserted memorable data: User has recently purchased a new super Mario videogame.

Inserted memorable data: He is a big fan of nintendo games and has a switch for the nintendo exclusives.

Figure 29: Memorization Process of the Demo Example

First we can see in the image the response of the LLM in charge of extracting the memorable personal data. This response has a specific format that is used to be able to extract in a segregated way each of the memorable data.

Among these extracted memorable data we see how the LLM has been able to synthesize the most valuable information of the conversation that has been had, such as that Max is taking pills, that the user wants a puppy of the same breed as his dog or that the user is a fan of nintendo.

The second part of the Figure 29 diagram represents the process of first comparing the new memorable data with the data in the database to see if that data already exists in the database and then updating that data or inserting it as new data.

We see how for the first memorable data extracted by the LLM, the user has a dog named Max, it has been correctly identified that this data was already in the database. So, instead of inserting it as a new data, the value of this data already present in the database is updated with an increment in the count value and an update in the timestamp of this memorable data.

As in the initial conditions section, we have retrieved after the conversation the memorable data from the database to observe how they have been updated and inserted correctly. This information can be seen in Figure 30

```
● (env) agarrachon@thau02:/autofs/home/gth04b/agarrachon/tfm$ python3 print_all.py
Data: User play tennis since he was 10 years old, Timestamp: 2024-05-27T17:10:45.241088, Count: 1.
Data: Max is a golden retriever, Timestamp: 2024-05-27T17:10:45.302117, Count: 1.
Data: User likes video games, Timestamp: 2024-05-28T07:27:54.717087, Count: 1.
Data: Likes pokemon and zelda, big fan of nintendo, Timestamp: 2024-05-28T07:27:54.748514, Count: 1.
Data: Has a ps4 and a switch, Timestamp: 2024-05-28T07:27:54.779252, Count: 1.
Data: User name is Alfredo, Timestamp: 2024-05-28T13:16:31.191226, Count: 2.
Data: User went to the azores on vacation, Timestamp: 2024-05-28T15:01:17.682484, Count: 1.
Data: User's dog is sick, Timestamp: 2024-05-28T15:01:17.714171, Count: 1.
Data: Has a pet dog named Max, Timestamp: 2024-05-28T20:24:41.120523, Count: 2.
Data: Max is taking pills but will be better soon. , Timestamp: 2024-05-28T20:24:54.718556, Count: 1.
Data: User wants a puppy of Max and will take Max to meet his english teacher golden retriever. , Timestamp: 2024-05-28T20:24:54.753681, Count: 1.
Data: User has recently purchased a new super Mario videogame. , Timestamp: 2024-05-28T20:24:54.785712, Count: 1.
Data: He is a big fan of nintendo games and has a switch for the nintendo exclusives. , Timestamp: 2024-05-28T20:24:54.820667, Count: 1.
```

Figure 30: Contents of the database after the demo conversation

In Figure 30 we can see how the new extracted personal memorable data has been entered into the database and how the data about the user's dog name has been updated with a new timestamp and its count parameter has been incremented.

7 Conclusions.

After having completed the development of this project, some interesting conclusions can be drawn both in the theories that have been explored to corroborate its proper functioning and in the final demo that has been developed and demonstrated.

During the state-of-the-art research, a lot of knowledge has been acquired and understood about the functioning of human memory and the most current innovative techniques for memory management and expansion in generative artificial intelligence chatbots. This knowledge has been used to reason and design the initial architecture of the chatbot that has been developed as the final system.

To finalize the design of the architecture and validate its use, two hypotheses about the capabilities of LLMs to extract and estimate the temporal duration of personal memorable data in a human-like manner have had to try to be corroborated. For this purpose, a dataset of 10 conversations involving 5 volunteers has been generated, a test with human annotation carried out by 14 people has been designed and carried out and another test with a specific prompts chain design for the 4 LLMs to be analyzed and compared has also been carried out.

Based on the data collected, human-to-human consistency has been validated using the inter-annotator agreement technique with volunteers involving a Krippendorff's Alpha of 0.8199. A specific score formula based on the nature of the memories has been designed to compare the LLMs' abilities in this task. This formula has been validated with deterministic techniques tested using a methodology involving the cosine distance, obtaining values of around 86% of similarity with the human results.

With these results, it has been possible to corroborate that LLMs have the ability to extract personal memorable data with results similar enough to those of humans to be able to use them in this task. On the other hand, it has been observed that in the task of estimating the temporal duration of the memorable data, they obtain more disparate results, indicating that their use in this task cannot be validated for this project and that a more exhaustive analysis is required. Using the score formula developed, it has been concluded that the Llama 2 13b model is the most optimal for performing the task of extracting personal memorable data, with results similar to GPT4 and better than Mistral 7b or Gemma 7b.

After these analyses, the final architecture of the chatbot has been designed and all the necessary logic for its proper functioning and interconnection of the different processes has been developed. All necessary logic for its proper functioning and interconnection of the different processes that form it has been developed. This final system includes a memorization process that involves the use of LLMs in this task (its use has been previously validated) and a recall process based on different characteristic parameters designed taking inspiration from state-of-the-art techniques and the theory of human memory.

Finally, the good performance of the developed chatbot has been demonstrated. In this demonstration we have analyzed the interactions and scenarios where the added value offered by this system over a generic chatbot has been demonstrated. It has been possible to verify how the chatbot is able to remember personal data about the user at the appropriate times to give the feeling of interest and closeness to the user. Also how the extraction of memorable data occurs periodically in the conversation, the processing and storage of these data.

7.1 Future Work.

There are some aspects that, due to the complexity of the project, could not be carried out in the time available and will require be investigated in the future. The first of these is a deeper analysis and research on the ability of LLMs to estimate the temporal duration

of memorable data. Although the results obtained in the tests carried out have not been sufficient to validate this capability of LLMs, they yield some promising results that leave the way open for further research and experimentation.

Another aspect that is considered interesting for future research is the ability of LLMs to estimate the importance of memorable data. While this project has directly related the importance of a memorable piece of information to the number of times it appears in the conversation, it is also considered interesting to explore the extent to which LLMs themselves give a value of importance to each memorable piece of information.

Thus, as less research work and more implementation work considered relevant is to take this developed system to a production platform where it can be tested with people from all over the world. This final objective is sought both to observe the results and people's evaluations of the system with the aim of improving it even more and to put into practice the initial objective of the project, to be able to offer a chatbot system that gives the feeling of closeness and interest to people who need it.

Índice de Tablas e Imágenes

- [1] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 Conference on Fairness, Accountability, and Transparency*, pages 610–623. ACM, 2021. doi: 10.1145/3442188.3445922. URL <https://doi.org/10.1145/3442188.3445922>.
- [2] Hongru Wang, Rui Wang, Fei Mi, Yang Deng, Zezhong Wang, Bin Liang, Ruifeng Xu, and Kam-Fai Wong. Cue-cot: Chain-of-thought prompting for responding to in-depth dialogue questions with llms, 2023.
- [3] Jimmy Wu, Rika Antonova, Adam Kan, Marion Lepert, Andy Zeng, Shuran Song, Jeannette Bohg, Szymon Rusinkiewicz, and Thomas Funkhouser. Tidybot: personalized robot assistance with large language models. *Autonomous Robots*, 47(8): 1087–1102, November 2023. ISSN 1573-7527. doi: 10.1007/s10514-023-10139-z. URL <http://dx.doi.org/10.1007/s10514-023-10139-z>.
- [4] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning, 2022.
- [5] Alireza Salemi, Sheshera Mysore, Michael Bendersky, and Hamed Zamani. Lamp: When large language models meet personalization, 2024.
- [6] Joe Dav. Zero-shot learning: The good, the bad, and the ugly, May 29 2020. URL <https://joeddav.github.io/blog/2020/05/29/ZSL.html>. Retrieved from <https://joeddav.github.io/blog/2020/05/29/ZSL.html>.
- [7] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units, 2016.
- [8] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google’s neural machine translation system: Bridging the gap between human and machine translation, 2016.
- [9] Michael Harris. Tokenization algorithms explained, March 26 2020. URL <https://towardsdatascience.com/tokenization-algorithms-explained-e25d5f4322ac>. Retrieved from <https://towardsdatascience.com/tokenization-algorithms-explained-e25d5f4322ac>.
- [10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.
- [11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.
- [12] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023.
- [13] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

- [14] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding, 2020.
- [15] Anshuman Tripathi, Jaeyoung Kim, Qian Zhang, Han Lu, and Hasim Sak. Transformer transducer: One model unifying streaming and non-streaming speech recognition, 2020.
- [16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.
- [17] Tony Z. Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. Calibrate before use: Improving few-shot performance of language models, 2021.
- [18] Albert Webson and Ellie Pavlick. Do prompt-based models really understand the meaning of their prompts?, 2022.
- [19] Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. Challenges and applications of large language models, 2023.
- [20] Elvis Saravia. Prompt Engineering Guide. <https://github.com/dair-ai/Prompt-Engineering-Guide>, 12 2022.
- [21] promptslab. Awesome-prompt-engineering. <https://github.com/promptslab/Awesome-Prompt-Engineering>, 2023.
- [22] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.
- [23] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.
- [24] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models, 2023.
- [25] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2023.
- [26] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 03 1960. ISSN 0021-9223. doi: 10.1115/1.3662552. URL <https://doi.org/10.1115/1.3662552>.
- [27] Anthropic. Claude 3 family, March 11 2024. URL <https://www.anthropic.com/news/clause-3-family>. Retrieved from <https://www.anthropic.com/news/clause-3-family>.
- [28] Google. Introducing gemini 1.5, google's next-generation ai model, 2024. URL <https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024/>. Acceso el 29 de febrero de 2024.

- [29] The New York Times. Interview with demis hassabis of google deepmind. <https://www.nytimes.com/2024/02/23/podcasts/google-deepmind-demis-hassabis.html>, Feb 2024. Acceso el 29 de febrero de 2024.
- [30] W. Ho, K. Dautenhahn, M. Lim, S. Enz, C. Zoll, and S. Watson. Towards learning 'self' and emotional knowledge in social and cultural human-agent interactions. *International Journal of Agent Technologies and Systems*, 1(3):51–78, 2009.
- [31] Arthur W. Melton. Implications of short-term memory for a general theory of memory. *Journal of Verbal Learning and Verbal Behavior*, 2(1):1–21, 1963. ISSN 0022-5371. doi: [https://doi.org/10.1016/S0022-5371\(63\)80063-8](https://doi.org/10.1016/S0022-5371(63)80063-8). URL <https://www.sciencedirect.com/science/article/pii/S0022537163800638>.
- [32] José María Ruiz-Vargas and José María López-Frutos. *Neurociencia Cognitiva*. 2014.
- [33] L.R. Squire. Memory systems of the brain: A brief history and current perspective. *Neurobiology of Learning and Memory*, 2004. doi: 10.1016/j.nlm.2004.06.005.
- [34] F. P. G. Maine de Biran. *The influence of habit on the faculty of thinking*. Williams Wilkins, Baltimore, 1929. First published in 1804.
- [35] L. R. Squire and S. Zola-Morgan. The neurology of memory: The case for correspondence between the findings for human and nonhuman primate. In J. A. Deutsch, editor, *The physiological basis of memory*, pages 199–268. Academic Press, New York, 1983.
- [36] J. O’Keefe and L. Nadel. *The hippocampus as a cognitive map*. Oxford University Press, Oxford, 1978.
- [37] E. Tulving, D. Schacter, and H. A. Stark. Priming effects in word-fragment completion are independent of recognition memory. *Learning, Memory and Cognition*, 8:336–341, 1982.
- [38] N. J. Cohen and L. R. Squire. Preserved learning and retention of pattern analyzing skill in amnesia: Dissociation of knowing how and knowing that. *Science*, 210: 207–209, 1980.
- [39] D. L. Schacter and E. Tulving. What are the memory systems of 1994? In D. L. Schacter and E. Tulving, editors, *Memory systems 1994*, pages 1–38. MIT Press, Cambridge, MA, 1994.
- [40] P. F. Gomes, C. Martinho, and A. Paiva. I’ve been here before! location and appraisal in memory retrieval. In K. Tumer, P. Yolum, L. Sonenberg, and P. Stone, editors, *Proceedings of the 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, pages 1039–1046, Taipei, Taiwan, May 2–6 2011.
- [41] Joana Campos and Ana Paiva. May: My memories are yours. In Jan Allbeck, Norman Badler, Timothy Bickmore, Catherine Pelachaud, and Alla Safanova, editors, *Intelligent Virtual Agents*, pages 406–412, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-15892-6.
- [42] E. Tulving. Episodic and semantic memory. In E. Tulving and W. Donaldson, editors, *Organization of Memory*, pages 381–403. Academic Press, New York, 1972.
- [43] E. Tulving. Remembering and knowing the past. *American Scientist*, 77:361–367, 1989.

- [44] E. Tulving. Concepts of human memory. In L. R. Squire, N. M. Weinberger, G. Lynch, and J. L. McGaugh, editors, *Memory: Organization and locus of change*, pages 3–32. Oxford University Press, 1991.
- [45] E. Tulving. On the uniqueness of episodic memory. In L.-G. Nilsson and H. J. Markowitsch, editors, *Cognitive neuroscience of memory*, pages 11–42. Hogrefe Huber Publishers, 1999.
- [46] E. Tulving and H.J. Markowitsch. Episodic and declarative memory: role of the hippocampus. *Hippocampus*, 8(3):198–204, 1998. doi: 10.1002/(SICI)1098-1063(1998)8:3<198::AID-HIPO2>3.0.CO;2-G.
- [47] William James. *The Principles of Psychology*. Henry Holt and Company, New York, 1890.
- [48] Donald O. Hebb. *The Organization of Behavior: A Neuropsychological Theory*. Wiley, New York, 1949.
- [49] Richard C. Atkinson and Richard M. Shiffrin. Human memory: A proposed system and its control processes. *The psychology of learning and motivation*, 2:89–195, 1968.
- [50] G. A. Miller, E. Galanter, and K. H. Pribram. *Plans and the Structure of Behavior*. Henry Holt and Co., 1960. doi: 10.1037/10039-000.
- [51] L. R. Peterson and M. J. Peterson. Short-term retention of individual verbal items. *Journal of Experimental Psychology*, 58:193–198, 1959. doi: 10.1037/h0049234.
- [52] M. Moscovitch. A neuropsychological model of memory and consciousness. In L. R. Squire and N. Butters, editors, *Neuropsychology of Memory*, pages 5–22. The Guilford Press, 2 edition, 1992.
- [53] S. Bluck. Autobiographical memory: Exploring its functions in everyday life. *Memory*, 11(2):113–123, 2003.
- [54] M.A. Conway. Sensory-perceptual episodic memory and its context: autobiographical memory. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 356:1375–1384, 2001.
- [55] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness, 2022.
- [56] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer, 2020.
- [57] Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers, 2020.
- [58] Krzysztof Choromanski, Valerii Likhoshesterov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers, 2022.
- [59] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity, 2020.
- [60] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020.

- [61] Beidi Chen, Tri Dao, Eric Winsor, Zhao Song, Atri Rudra, and Christopher Ré. Scatterbrain: Unifying sparse and low-rank attention approximation, 2021.
- [62] Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences, 2021.
- [63] Zhe Jia, Marco Maggioni, Benjamin Staiger, and Daniele P. Scarpazza. Dissecting the nvidia volta gpu architecture via microbenchmarking, 2018.
- [64] Andrei Ivanov, Nikoli Dryden, Tal Ben-Nun, Shigang Li, and Torsten Hoefer. Data movement is all you need: A case study on optimizing transformers, 2021.
- [65] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning, 2023.
- [66] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts, 2023.
- [67] Together Computer. Llama-2-7b-32k. <https://huggingface.co/togethercomputer/LLaMA-2-7B-32K>, 2024. 14/2/2024.
- [68] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Bin Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.
- [69] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context, 2019.
- [70] Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers, 2022.
- [71] Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. Accelerating large-scale inference with anisotropic vector quantization, 2020.
- [72] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus, 2017.
- [73] Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory, 2023.
- [74] OpenAI. Gpt-4, 2023. URL <https://openai.com/research/gpt-4>.
- [75] Kai Zhang, Fubang Zhao, Yangyang Kang, and Xiaozhong Liu. Memory-augmented llm personalization with short- and long-term memory coordination, 2023.

- [76] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. <https://github.com/huggingface/peft>, 2022.
- [77] Rishin Haldar and Debajyoti Mukhopadhyay. Levenshtein distance technique in dictionary lookup methods: An improved approach, 2011.
- [78] Baoli Li and Liping Han. Distance weighted cosine similarity measure for text classification, 2013.
- [79] Qingyue Wang, Liang Ding, Yanan Cao, Zhiliang Tian, Shi Wang, Dacheng Tao, and Li Guo. Recursively summarizing enables long-term dialogue memory in large language models, 2023.
- [80] Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems, 2024.
- [81] Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior, 2023.
- [82] James D. Hollan, Edwin L. Hutchins, and Louis Weitzman. Steamer: An interactive inspectable simulation-based training system. *AI Magazine*, 5(2):23–36, 1984.
- [83] Milind Tambe, W Lewis Johnson, Randolph M Jones, Frank Koss, John E Laird, Paul S Rosenbloom, and Karl Schwamb. Intelligent agents for interactive simulation environments. *AI Magazine*, 16(1):15, 1995.
- [84] John J. Horton. Large language models as simulated economic agents: What can we learn from homo silicus?, 2023.
- [85] LangChain Contributors. Langchain: Build context-aware reasoning applications. <https://github.com/langchain-ai/langchain>, 2024. GitHub repository.
- [86] Quick start: Langchain agents. https://python.langchain.com/docs/modules/agents/quick_start, 2024. Wiki.
- [87] Langgraph documentation. <https://js.langchain.com/docs/langgraph>, 2024. Wiki.
- [88] LangChain. Langgraph: Multi-agent workflows, 2024. URL <https://blog.langchain.dev/langgraph-multi-agent-workflows/>
- [89] Langroid Contributors. Langroid: Harness llms with multi-agent programming. <https://github.com/langroid/langroid>, 2024. GitHub repository.
- [90] Langroid Development Team. Task delegation - task collaboration via sub-tasks. <https://langroid.github.io/langroid/quick-start/multi-agent-task-delegation/#task-collaboration-via-sub-tasks>, 2024. Accessed: INSERT-DATE-OF-ACCESS.
- [91] joaomdmoura and CrewAI Contributors. crewai: Framework for orchestrating role-playing, autonomous ai agents. <https://github.com/joaomdmoura/crewAI>, 2024. Accessed: 2024-02-17.
- [92] Assembling and activating your crewai team. <https://docs.crewai.com/how-to/Creating-a-Crew-and-kick-it-off/#introduction>, 2024. Accessed: 2024-02-17.

- [93] Microsoft and AutoGen Contributors. Autogen: Enable next-gen large language model applications. <https://github.com/microsoft/autogen>, 2024. GitHub repository.
- [94] Kosovare Sahatqija, Jaumin Ajdari, Xhemal Zenuni, Bujar Raufi, and Florije Ismaili. Comparison between relational and nosql databases. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 0216–0221, 2018. doi: 10.23919/MIPRO.2018.8400041.
- [95] GeeksforGeeks. Types of nosql databases. <https://www.geeksforgeeks.org/types-of-nosql-databases/>, 2023. Último acceso: 2024-03-30.
- [96] MongoDB, Inc. Json and bson. <https://www.mongodb.com/json-and-bson>, 2023. Último acceso: 2024-03-30.
- [97] MongoDB, Inc. Pymongo 3.12 documentation. <https://pymongo.readthedocs.io/en/stable/>, 2023. Último acceso: 2024-03-30.
- [98] Google. Gemma: A leap forward in open model technology. <https://blog.google/technology/developers/gemma-open-models/>, 2024. Acceso el 29 de febrero de 2024.
- [99] DeepMind. Gemma report. Technical report, DeepMind, 2024.
- [100] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7b, 2023.
- [101] Meta and Microsoft. Meta and microsoft introduce the next generation of llama, July 2023. Disponible en: <https://about.fb.com/news/2023/07/llama-2/>
- [102] OpenAI. Gpt-4: Openai's generative pre-trained transformer 4. *OpenAI Blog*, 2023. URL <https://openai.com/research/gpt-4>
- [103] OpenAI, :, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madeleine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish

Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parasandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael Pokorny, Michelle Pokrass, Vitchyr Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotstetd, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2023.

- [104] Hugging Face H4. Open l1m leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard, 2024. Acceso el 29 de febrero de 2024.
- [105] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021.
- [106] Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019.
- [107] Arseny Moskvichev, Victor Vikram Odouard, and Melanie Mitchell. The conceptarc benchmark: Evaluating understanding and generalization in the arc domain, 2023.
- [108] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them, 2022.
- [109] Xinchao Xu, Zhibin Gou, Wenquan Wu, Zheng-Yu Niu, Hua Wu, Haifeng Wang, and Shihang Wang. Long time no see! open-domain conversation with long-term persona memory, 2022.

- [110] Alexa. Topical-chat: A knowledge-grounded human-human conversation dataset. <https://github.com/alexa/Topical-Chat/tree/master>, 2024. Acceso el 29 de febrero de 2024.
- [111] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. Dailydialog: A manually labelled multi-turn dialogue dataset, 2017.
- [112] Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. Towards empathetic open-domain conversation models: a new benchmark and dataset, 2019.
- [113] Pegah Jandaghi, XiangHai Sheng, Xinyi Bai, Jay Pujara, and Hakim Sidahmed. Faithful persona-based conversational dataset generation with large language models, 2023.
- [114] Desconocido. Ollama model library. <https://github.com/ollama/ollama?tab=readme-ov-file#model-library>, 2023. Último acceso: [Fecha de acceso].
- [115] Desconocido. Documentation for ollama model file. <https://github.com/ollama/ollama/blob/main/docs/modelfile.md>, 2023. Último acceso: [Fecha de acceso].
- [116] Chatgpt. <https://openai.com/blog/chatgpt>.
- [117] OpenAI. Openai pricing information. <https://openai.com/pricing>, 2023. Último acceso: [Fecha de acceso].
- [118] Klaus Krippendorff. Computing krippendorff's alpha-reliability. <https://www.asc.upenn.edu/sites/default/files/2021-03/Computing%20Krippendorff%27s%20Alpha-Reliability.pdf>, 2021.
- [119] Santiago Castro. Fast Krippendorff: Fast computation of Krippendorff's alpha agreement measure. <https://github.com/pln-fing-udelar/fast-krippendorff>, 2017.
- [120] Luciano da F. Costa. Further generalizations of the jaccard index, 2021.
- [121] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL <https://doi.org/10.3115/1073083.1073135>.
- [122] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL <https://aclanthology.org/W04-1013>.
- [123] Xing Gu and P. Rosenbaum. Comparison of multivariate matching methods: Structures, distances, and algorithms. *Journal of Computational and Graphical Statistics*, 2:405–420, 1993. doi: 10.1080/10618600.1993.10474623.
- [124] Hugging Face. Introducing mteb: A benchmark for evaluating multimodal models, 2022. URL <https://huggingface.co/blog/mteb>. [Online; accessed 1-April-2024].
- [125] MixedBread AI. MXBAI-Embed-Large-v1 model. Hugging Face Model Hub, 2023. URL <https://huggingface.co/mixedbread-ai/mxbai-embed-large-v1>. [Online; accessed 1-April-2024].
- [126] ScienceDirect. Pearson correlation - an overview. <https://www.sciencedirect.com/topics/computer-science/pearson-correlation>, 2023. Último acceso: 5 de abril de 2024.

- [127] Meta AI. Meta llama 3. <https://ai.meta.com/blog/meta-llama-3/>, 2024. Accedido: 2024-05-09.
- [128] Wikipedia. Regular expression. https://en.wikipedia.org/wiki/Regular_expression, 2024. Accedido: 2024-05-09.
- [129] Lloyd R. Peterson and Margaret Jean Peterson. Short-term retention of individual verbal items. *Journal of Experimental Psychology*, 58(3):193–198, 1959.
- [130] PyTorch. Get started locally with pytorch, 2024. URL <https://pytorch.org/get-started/locally/>. Accessed: 2024-05-23.
- [131] Docker. Install docker engine, 2024. URL <https://docs.docker.com/engine/install/>. Accessed: 2024-05-23.
- [132] Ollama. Download ollama, 2024. URL <https://ollama.com/download>. Accessed: 2024-05-23.

A Annex: Ethical, economic, social and environmental issues

A.1 Introduction

Technology has entered our lives in a way that would have been unthinkable years ago. It is present in every imaginable aspect from work to social life. Although it also has its negative sides and has caused problems in some areas such as reduced social interactions or new kinds of bullying, its main objective is to improve and facilitate people's lives.

In this project we wanted to develop a system that aims to help in the social field to many people. More specifically, we wanted to develop a chatbot with long term memory about the user that is able to use this information at the right time so that it gives the feeling of showing interest to the user and, therefore, feel closer and more enjoyable.

This system is directly oriented to people who are in a situation of social exclusion, introverted people and elderly people with a feeling of loneliness. Having a system, even an artificial one, to talk to and show interest towards the user can help in their social situation.

Regarding the impacts and problems related to the TFM, we have to take into account that this system is a software system, so the problems of raw materials, distribution, recycling and any aspect related to the physical format do not apply in this scenario.

The following is a list of the impacts (positive and negative) and problems that have been detected in this project, the most important of which will be analyzed later:

Positive

- Reduce Psychological harm
- Reduce Harms to society
- Reduce power utilization
- Social Inclusion
- Supportive of mental health
- Accessible to everyone
- Reduce waste of resources
- Maintenance, robustness
- Economic viability

Negative

- Information Privacy
- Risk of Integration and social acceptance

A.2 Description of relevant impacts related to the project

Next, the aspects and impacts (both positive and negative) that have been considered most relevant will be selected according to various criteria, such as their importance or the opportunity to address them. For each of these, a description of their significance and orientation for this specific project will be provided.

Reduce power utilization. This aspect focuses on the objective of seeking to reduce the power used as much as possible, which leads to a more optimized use of energy. It is included within the environmental objectives as one of the most important, since, by reducing the power used, many other aspects related to energy generation are improved.

The way to measure this aspect is based on the direct comparison of the energy consumption of a system with another system previously developed with the same objective and focus.

Supportive mental health. Mental health is one of the biggest health problems facing people today. Again, maintaining a healthy mental health is a desire for the whole population

but the people who suffer the most from this problem are the group of people suffering from depression.

To assess this impact, we would need a whole population to study and a group of psychologists to study their mental health at different periods. Although it seems to be a similar feeling to happiness, this is more complicated to quantify and experts are needed to verify it since, unlike happiness, the concept of mental health is not a feeling that can be expressed by one person and includes many aspects.

Accessible to everyone. This impact reflects that any system or technique developed during the project can be used by anyone. The mainly affected population group can be considered people with economic or physical difficulties, who are still able to make use of this chatbot system.

The form of evaluation of this impact is relatively simple compared to other impacts since it is necessary to certify that the developed product can be obtained, used freely and in any adapted device by anyone in the world.

Information Privacy. This impact reflects the importance of keeping information private in accordance with a set of regulations. The population group affected by this impact is defined as any person who has a relationship with the developed product and who, therefore, has data related to that person.

The most effective way to assess this impact is with continuous external monitoring of the data that has been collected. This external supervision would have to be carried out by specialized persons who follow the established information privacy rules and certify that the data is being processed correctly.

This impact has a series of laws and standards to comply with which are mainly related to the treatment and use of each person's personal information. There are also financial implications related to non-compliance with these laws with a series of associated financial penalties.

Risk of Integration and social acceptance. This impact is defined as the risk that society and the stakeholder group do not accept the product in a satisfactory manner and therefore reject it. The population group of interest in this case is defined as all the people to whom the product or system is oriented.

If we want to carry out the evaluation of this impact, a random sample of people within the population group of interest would have to be taken and a survey on the acceptance of the product would have to be carried out. In this way, it would be possible to know the percentage of the population within the target group that accepts the product.

A.3 Detailed analysis of some of the main impacts.

In this section we will take the main and most relevant impacts from the previous section and analyze them in more detail from the point of view of the development of this project.

Reduce power utilization. Throughout the development of the system, priority has been given to techniques that optimize energy use and reduce waste of resources. Among some of the aspects taken advantage of are: use of pre-trained models, use of small size models with lower associated consumption, use of optimization techniques to reduce the necessary resources, etc.

One way to measure this aspect, as we have mentioned before, is based on analyzing pre-existing systems with the same objective: chatbots. Looking for example at ChatGPT, this is based on models trained for this specific system and of large size that require a lot of resources and energy to be used, with a consumption of power and resources much higher than the developed system.

Supportive mental health. In a similar way to the previous impact, having a close entity to express oneself to can help the mental health of the user. People could express their feelings and anxieties to the system knowing that they will not be judged and at the same time they will be listened to and the system will show interest in their problems.

Measuring this impact, as discussed above, is more complicated than the previous impact. This would require a series of psychological experts to analyze the users who are using the developed chatbot and measure if their mental health suffers any modification.

Accessible to everyone. One of the main objectives of the project is to develop it in a way that it can be used freely by anyone and even on domestic hardware without the need to depend on private companies with products under subscription.

The measurement of this impact is simpler since it is enough to certify at the end of the project that all the software is open source for free use by anyone and that it can be run on domestic hardware without depending on large companies and private systems.

Information privacy. In the context of the project, information privacy is a key aspect since the system is directly oriented to store the user's personal information to be used in future interactions.

As mentioned in the previous section, to evaluate this impact, a set of people in charge of this supervision and of the compliance with the current regulations is needed. On the other hand, another form of justification is the demonstration that each person's data does not leave his or her own local environment and therefore no other person has access to it, as it is the case for this project.

Focusing also on the data generation and evaluation carried out, it was ensured that no personal or private data was included in the collection of the conversational data and the volunteers' responses to the tests were anonymized.

Risk of integration and social acceptance. The developed system, interpreted as an artificial chatbot that tries to simulate the feeling of closeness and interest towards the user, can have a negative and bad acceptance by the population when trying to imitate human characteristics in a system that is not human.

In order to evaluate this impact, a survey would have to be carried out on a random sample of the population. In this process, the developed chatbot system would be explained to them, they would be given the ability to use it and a survey on its acceptance would be conducted.

A.4 Conclusions

After having analyzed the most important ethical, legal, economic, social and environmental impacts and aspects related to the project, we can draw a series of conclusions about the treatment of these and how they have affected the development of the project, as well as whether they have been fulfilled at the end of the project.

On the part of the impacts of Right to pursue happiness and Supportive mental health, we have continued with the idea of developing the chatbot with the ability to remember the personal data that the user has communicated to the chatbot in previous interactions and make use of these at appropriate times to show the feeling of closeness and interest.

As mentioned in the Future Work section, we have not had the necessary time to carry out the subsequent evaluation of the system and show if this feeling is fulfilled or not, so it is a pending evaluation that could be carried out in the future based on the developed system.

As for the Accessible for everyone aspect, we can certify that this is fulfilled since the system has been oriented and designed around this requirement. The chatbot system runs on open source models that are free to use and small enough to run on home hardware.

We can also assure the information privacy aspect as the system is intended to be used locally by the user himself. All processing and storage of the user's personal data is executed in the local environment and is not accessible by any other entity.

As for the last relevant aspect selected for analysis, the Risk of integration and social acceptance, it cannot be assessed manually without conducting a survey among a random sample. Therefore, the final validation of this aspect is pending as future work within the final human evaluation of this system by the global population.

B Annex: Financial Budget

In this annex we have developed the economic budget for the development of the chatbot system carried out in this project. Next, in the table 9, we show the analysis carried out with the final result of the total economical budget needed.

LABOR COST (direct cost)	Hours	Price/hour	Total
	750	30	22.500
COST OF MATERIAL RESOURCES (direct cost)			
Personal computer (software included).....	1.549,99	6	5
Remote lab machine	3200,00	6	5
TOTAL COST OF MATERIAL RESOURCES			474,99
GENERAL EXPENSES (indirect costs)	15 %	over DC	2.321,25
INDUSTRIAL PROFIT	6 %	over DC + IC	928,50
SUBTOTAL BUDGET			26.224,74
VAT APPLICABLE			5.507,19
TOTAL BUDGET			31.731,93

Table 9: Economical budget

C Annex: Examples of conversational dataset

agent_1: Do you like animals?
agent_2: Not very much, I'm afraid of dogs for example.
agent_1: Well, I love them, I have a lot of pets.
agent_2: Do you have a dog?
agent_1: Yes, he is my best friend, if you want I can introduce him to you.
agent_2: I'm not sure, I'm afraid of him.
agent_1: My dog is not dangerous, he is very small too.
agent_2: What is his name?
agent_1: Yew, he looks like a cuddly toy, he has never attacked anyone and he is very good.
agent_2: We can try, bring him if you want.
agent_1: Where does your fear of dogs come from?
agent_2: When I was 7 years old a wild dog attacked me when I was playing in the park.
agent_1: That's too bad, but that shouldn't make you afraid of dogs, most dogs are very good.
agent_2: Maybe, but I've been afraid of them ever since.
agent_1: Do you have any pets?
agent_2: I had a turtle, but a year ago it jumped off the balcony and we haven't found it again.
agent_1: And how come it was able to jump over the balcony if turtles don't jump.
agent_2: He climbed over the railing and jumped.
agent_1: What a pity.

Figure 31: First example of Conversational Dataset

agent_1: Last year I went on a climbing trip to the mountains.
agent_2: How cool, did you like the experience?
agent_1: Very much, but I would not repeat the experience because I am afraid of heights.
agent_2: That's too bad, I'm sure you didn't have a good time.
agent_1: Not at that moment, I just wanted to finish.
agent_2: Did you have any trick to avoid the fear of heights?
agent_1: To look up all the time and think that it is an experience that I had to live once my life.
agent_2: Well, two years ago I went to ride jet skis, you might like that experience better.
agent_1: Maybe. And what did you think, would you repeat the experience?
agent_2: Without hesitation, I had a great time and it was a lot of fun.
agent_1: Next time you go, let me know and let's go together, I want to try it.
agent_2: Perfect. The only problem is that it is a bit expensive.
agent_1: That's ok, I have some savings that I have been saving for occasions like this.
agent_2: Then no problem, I have vacation in August this year, we could go around that time.
agent_1: Fine with me! Let's meet then to go jet skiing in August.
agent_2: Great!

Figure 32: Second example of Conversational Dataset

agent_1: Nice to meet you, what are you studying?
agent_2: I am studying a degree in computer science, I started it 5 years ago.
agent_1: How interesting, that field is being very relevant in the last years.
agent_2: Yes, that's why I chose it, to find a job easier.
agent_1: And where do you study?
agent_2: At San Pablo University, it is located north of the city.
agent_1: I've been told that this university is very good, I'm sure it was hard for you to get in.
agent_2: Yes, I had to prepare for an entrance exam for 1 year. I even had to attend classes.
agent_1: It was really hard then.
agent_2: And you, what are you studying?
agent_1: I'm studying a master's degree in marketing, I've been studying for 1 year and a half.
agent_2: In which university?
agent_1: It is not very well known, it is the University of San Lorenzo, it is in the south of the city.
agent_2: Yes, I know which one it is, a friend of mine is studying a degree in philosophy there.
agent_1: Oh yes, what is your friend's name? Maybe I know her
agent_2: My friend's name is Lorena Garcia.
agent_1: Ah yes! I know who she is, we have some friends in common.
agent_2: What a coincidence, well, when I see her I will tell her about you.
agent_1: I'm sure she remembers me!

Figure 33: Third example of Conversational Dataset

agent_1: Do you play any sport?
agent_2: I like tennis very much, I've been playing since I was a child.
agent_1: What a coincidence! I love it too, I started playing when I was 5 years old.
agent_2: And where did you play?
agent_1: In the court next to the town hall.
agent_2: I used to play in the park next to the river, but I also went there some times.
agent_1: If you want to play a game some day.
agent_2: I don't have the tennis racquets anymore, but if you want to play paddle tennis.
agent_1: It also works for me, although I have only played a match.
agent_2: No problem, I'll teach you, I've been playing for 2 years.
agent_1: Perfect, if you want let's go to the paddle courts next to the town hall.
agent_2: Fine with me, if you want I'm a member of the paddle courts.
agent_1: Wednesday of next week is good for you?
agent_2: Yes, now I call the paddle club and reserve.
agent_1: I injured my ankle 1 month ago, I hope it won't be a problem.
agent_2: Do you have an ankle brace?
agent_1: Yes, I have already practiced a little sport and nothing has happened to me.
agent_2: Then there should be no problem, see you there.
agent_1: Perfect!

Figure 34: Fourth example of Conversational Dataset

agent_1: Do you have any hobbies?
agent_2: Now I am very busy with my studies, but when I have some time I like to play videogames.
agent_1: I also like video games, two weeks ago I bought the PS5.
agent_2: That's great! Have you bought any game with the console?
agent_1: The Elden Ring game, I really wanted to play it.
agent_2: I love that game, but it is very complicated, I played it when it came out 2 months ago.
agent_1: I will start to play it now, if I have any problem I will ask you for help.
agent_2: Of course, I will be happy to help you.
agent_1: Do you have a favourite videogame saga?
agent_2: I love the Pokemon series. I played the first game when I was 9 years old.
agent_1: Me too, my favourite game of the series is Pokemon Diamond.
agent_2: I loved that game, but the one I like the most is Pokemon Emerald.
agent_1: It's also very cool, recently a new one has been released and it has very good reviews.
agent_2: Oh yeah, what's it called?
agent_1: Pokemon Arceus, I recommend it.
agent_2: I'll play it when I finish the elden ring.
agent_1: Perfect, you will tell me what you think, I liked it a lot.
agent_2: Yes, let's stay in touch.

Figure 35: Fifth example of Conversational Dataset

agent_1: The other day I had a very strange experience.
agent_2: Tell me about it.
agent_1: I was at home and I heard my grandmother's voice.
agent_2: What's so strange about it?
agent_1: That my grandmother has been dead for 5 years.
agent_2: How strange, do you believe in ghosts?
agent_1: Not much, but this experience makes me doubt.
agent_2: I do believe in ghosts, my aunt is a psychic, I can ask her for help.
agent_1: Well, I would be interested if she could help me.
agent_2: She is the best psychic in town, she has already helped many people to talk to the dead.
agent_1: How much does she charge?
agent_2: 100 euros per session.
agent_1: Yes, let her know and have her come to my house as soon as possible.
agent_2: I need some information to give her. What is your grandmother's name and what did she do for a living?
agent_1: Her name was Carmen and she was a farmer.
agent_2: And what did she die of?
agent_1: She was run over by my grandfather with the tractor, my grandfather's eyesight was failing a little.
agent_2: What a shame. Well, I'll talk to my aunt and we'll solve this problem as soon as possible.
agent_1: Perfect! I'll be waiting for you.

Figure 36: Sixth example of Conversational Dataset

agent_1: Yesterday I went grocery shopping and all the vegetables were on sale.
agent_2: Oh yeah? which one did you go to?
agent_1: The one next to the church.
agent_2: Do you know which vegetables are better now?
agent_1: It is tomato season, they have brought them very good.
agent_2: That's good, I'll buy some, too bad it's not melon season.
agent_1: The melon season starts in May.
agent_2: How come you know so much about vegetables?
agent_1: When I was a child I grew up in the country with my grandfather who had a garden.
agent_2: And you grew a lot of vegetables?
agent_1: A lot, my grandfather was a farmer, we cultivated all year round.
agent_2: What an envy, I grew up in the city and did not set foot in the countryside.
agent_1: What did your parents do?
agent_2: My parents were bankers and they did not like the countryside.
agent_1: A pity
agent_2: But when I buy a house I want it to have a yard to cultivate my own plants.
agent_1: Then ask me for help when you need it, I can advise you.
agent_2: Perfect! when I have my garden I'll let you know
agent_1: Great.

Figure 37: Seventh example of Conversational Dataset

agent_1: Did you watch the Real Madrid game on Tuesday?
agent_2: The Champions League match?
agent_1: Yes
agent_2: I was watching Manchester City who were playing at the same time.
agent_1: So you like soccer right?
agent_2: Yes, I watch all the games of my team.
agent_1: What team are you?
agent_2: FC Barcelona
agent_1: Well, I'm from the rival team, Real Madrid FC
agent_2: How do you see the season?
agent_1: I see us strong in the Spanish league, but not so much in the Champions League.
agent_2: Well, I think FC Barcelona will not win anything this year.
agent_1: I heard that the club has money problems.
agent_2: Yes, but they want to sell players to get money and prepare for the next season.
agent_1: The next round of the Champions League is coming soon, if you want we can watch it together.
agent_2: Sure, you can come to my house, I have a new TV set.
agent_1: Perfect, I'll bring the drinks!
agent_2: Great

Figure 38: Eighth example of Conversational Dataset

agent_1: Next week is the week of offers in stores?
agent_2: Ah yes, in which stores are there offers?
agent_1: Mostly in clothing stores, but also in some electronics stores.
agent_2: It's great for me, I love fashion and clothes.
agent_1: I don't like it very much, but I need to buy some things.
agent_2: If you want I can go with you and help you to buy some things
agent_1: Do you know a lot about clothes?
agent_2: Yes, I know most of the brands and I know which ones are of good quality.
agent_1: And to combine clothes can you help me too?
agent_2: Of course!
agent_1: Great. I need mainly pants and some sweatshirt
agent_2: The Levis pants are of very good quality, I have several of them
agent_1: I had some but they broke when I went to climb a mountain.
agent_2: Normal, they are good pants but they are not made for the mountain.
agent_1: Do you know any brand of sport pants?
agent_2: Yes, if you want next week we go to the offers and I show you
agent_1: Perfect! Thank you very much
agent_2: No problem.

Figure 39: Ninth example of Conversational Dataset

agent_1: Do you like to travel?
agent_2: Yes, I love it, I want to take advantage now that I am a student to travel as much as possible.
agent_1: Where have you been recently?
agent_2: I've been to Rome and Florence recently.
agent_1: A tour of the most important cities in Italy, no?
agent_2: Just, I also wanted to go to Venice but I did not have the time
agent_1: And what did you think of them?
agent_2: Very nice, especially Florence, I think it is the best city in Italy.
agent_1: I want to go to Florence, I've been waiting for it.
agent_2: And you? What was the last city you have traveled to?
agent_1: I was in Tokyo recently
agent_2: How cool, I really want to go to Tokyo, how did you like it?
agent_1: I thought it was very nice but also very big, you do not have time to see it all.
agent_2: Is it true that it is very similar to the anime?
agent_1: Yes, the scenery is practically the same.
agent_2: And the clash of cultures? They say that the culture is very different
agent_1: Yes, the culture of Japan is very different from the European one.
agent_2: I want to go next year, I will ask you to help me.
agent_1: Great, I will help you in everything you need, I already know how to go there.

Figure 40: Tenth example of Conversational Dataset

D Annex: Task delivered to human annotators

The task consists of 3 conversations and each of them has a time projection assigned to it. The task is to read each one of them and think what personal data of that conversation would be memorable in the assigned time projection. That is, if you were inside that conversation and the two agents were people close to you, what do you think would be the data that you would still remember from that conversation after some time has passed, being the time the temporal projection itself.

Here is an example for a time projection of 1 week

Agent_1: I sprained my ankle playing paddle tennis last week.

Agent_2: What degree was it?

Agent_1: I didn't go to the doctor, they told me that ice would be enough.

Agent_2: I studied medicine and I think you should go to the doctor.

Agent_1: I will try to go, don't you have an extra crutch?

Agent_2: I have two, they belong to my grandfather.

An example of answers for 1 week could be:

- **Agent_1** sprained his ankle a week ago playing paddle tennis.

- **Agent_2** is studying medicine

- **Agent_2** is on crutches

For a time projection of 1 hour you might remember that **Agent_1** did not go to the doctor or that **Agent_2**'s crutches are his grandfather's. For longer projections like 2 months or 1 year you might not remember where or when he sprained his ankle or that **Agent_2** has crutches. For longer projections like 2 months or 1 year you might not remember where or when he sprained his ankle or that **Agent_2** has crutches.

So, this is a demonstrative example and the reasoning is up to you and should not be influenced by the example. It serves only to demonstrate the structure.

Here are the 3 conversations with their respective time projections. There is no hurry to carry out the task, it will be done when you have some free time and without any pressure.

Thank you very much.

Figure 41: Example of the Task delivered to the volunteers

E Annex: Log information of the example of execution

```
(env) agarrachon@thau02:/autofs/home/gth04b/agarrachon/tfm python3 main.py --log full  
/autofs/home/gth11a/agarrachon/env/lib/python3.10/site-packages/huggingface_hub/file_download.py:1132:  
FutureWarning: 'resume_download' is deprecated and will be removed in version 1.0.0.  
Downloads always resume when possible. If you want to force a new download, use  
'force_download=True'. warnings.warn()
```

Hi! I am your personal bot. What do you want to talk about today.

You: I have just come from a walk with Max

START RECALL PROCESS

Max Count: 2

Oldest Timestamp: 1716822645.241088

Current timestamp: 1716910454.181997

Max time difference: 87808.94090914726

SCORES FOR THE DATA START

Data: User play tennis since he was 10 years old, Time Score: 0.9048374180359595, Count Score: 0.5, Cosine Score: 0.5377047192414868, Weighted Score: 0.6200617141297333

Data: Has a pet dog named Max, Time Score: 0.9048374496300698, Count Score: 0.5, Cosine Score: 0.6569817028706495, Weighted Score: 0.6797002138428422

Data: Max is a golden retriever, Time Score: 0.9048374809242046, Count Score: 0.5, Cosine Score: 0.6053428685030533, Weighted Score: 0.6538808044825778

Data: User likes video games, Time Score: 0.9594162510140234, Count Score: 0.5, Cosine Score: 0.5444514184954667, Weighted Score: 0.6370797720012392

Data: Likes pokemon and zelda, big fan of nintendo, Time Score: 0.9594162853516263, Count Score: 0.5, Cosine Score: 0.5194241525809847, Weighted Score: 0.6245661476283989

Data: Has a ps4 and a switch, Time Score: 0.9594163189366447, Count Score: 0.5, Cosine Score: 0.47516421886788596, Weighted Score: 0.6024361891681042

Data: User name is Alfredo, Time Score: 0.982544330630168, Count Score: 1.0, Cosine Score: 0.6534215499520083, Weighted Score: 0.8223468582417583

Data: User went to the azores on vacation, Time Score: 0.9896038872858395, Count Score: 0.5, Cosine Score: 0.5616695540185029, Weighted Score: 0.6532357488307113

Data: User's dog is sick, Time Score: 0.9896039229970093, Count Score: 0.5, Cosine Score: 0.5389472015218075, Weighted Score: 0.6418745815101561 SCORES FOR THE DATA END

Relevant memories:

Memory: User name is Alfredo, Score: 0.8223

Memory: Has a pet dog named Max, Score: 0.6797

Memory: Max is a golden retriever, Score: 0.6538

Memory: User went to the azores on vacation, Score: 0.6532

Memory: User's dog is sick, Score: 0.6418

END RECALL PROCESS

Bot: How was Max doing during your walk today? Is he feeling any better since we last talked about his health issues?

You: I felt him a bit slower than usual

START RECALL PROCESS

Max Count: 2

Oldest Timestamp: 1716822645.241088

Current timestamp: 1716910471.419886

Max time difference: 87826.1787981987

SCORES FOR THE DATA START

Data: User play tennis since he was 10 years old, Time Score: 0.9048374180359595, Count Score: 0.5, Cosine Score: 0.5094147686087047, Weighted Score: 0.6059167388133422

Data: Has a pet dog named Max, Time Score: 0.9048374496238688, Count Score: 0.5, Cosine Score: 0.639566641815093, Weighted Score: 0.6709926833135137

Data: Max is a golden retriever, Time Score: 0.9048374809118613, Count Score: 0.5, Cosine Score: 0.5917659264855242, Weighted Score: 0.6470923334707275

Data: User likes video games, Time Score: 0.9594052219659894, Count Score: 0.5, Cosine Score: 0.48173039121870376, Weighted Score: 0.6057165011008492

Data: Likes pokemon and zelda, big fan of nintendo, Time Score: 0.9594052562964582, Count Score: 0.5, Cosine Score: 0.47511897046368157, Weighted Score: 0.6024107993059553

Data: Has a ps4 and a switch, Time Score: 0.9594052898744987, Count Score: 0.5, Cosine Score: 0.4220075606094742, Weighted Score: 0.5758551027733618

Data: User name is Alfredo, Time Score: 0.9825284445145566, Count Score: 1.0, Cosine Score: 0.6028893115690159, Weighted Score: 0.7970767669131471

Data: User went to the azores on vacation, Time Score: 0.9895864940377955, Count Score: 0.5, Cosine Score: 0.4836410261832487, Weighted Score: 0.6142171366010732

Data: User's dog is sick, Time Score: 0.9895865297413287, Count Score: 0.5, Cosine Score: 0.6249361546739722, Weighted Score: 0.6848647097723183

SCORES FOR THE DATA END

Relevant memories:

Memory: User name is Alfredo, Score: 0.7970

Memory: User's dog is sick, Score: 0.6848

Memory: Has a pet dog named Max, Score: 0.6709

Memory: Max is a golden retriever, Score: 0.6470

Memory: User went to the azores on vacation, Score: 0.6142

END RECALL PROCESS

Bot: That's understandable, Alfredo. Sorry to hear that Max is still not feeling 100%. Have you tried adjusting his medication or consulting with your vet for further guidance on how to help him recover?

You: We went to the veterinary yesterday and he told me it is normal because of the pills, but he will be good soon!

START RECALL PROCESS

Max Count: 2

Oldest Timestamp: 1716822645.241088

Current timestamp: 1716910512.035844

Max time difference: 87866.79475617409

SCORES FOR THE DATA START

Data: User play tennis since he was 10 years old, Time Score: 0.9048374180359595, Count Score: 0.5, Cosine Score: 0.47278152571739507, Weighted Score: 0.5876001173676875

Data: Has a pet dog named Max, Time Score: 0.9048374496092675, Count Score: 0.5, Cosine Score: 0.7013936229227237, Weighted Score: 0.7019061738636787

Data: Max is a golden retriever, Time Score: 0.9048374808827973, Count Score: 0.5, Cosine Score: 0.63773108254128, Weighted Score: 0.6700749114913394

Data: User likes video games, Time Score: 0.9593792529060501, Count Score: 0.5, Cosine Score: 0.4500404386977709, Weighted Score: 0.589865032575398

Data: Likes pokemon and zelda, big fan of nintendo, Time Score: 0.9593792872197209, Count Score: 0.5, Cosine Score: 0.4464665115944626, Weighted Score: 0.5880780776021615

Data: Has a ps4 and a switch, Time Score: 0.9593793207813317, Count Score: 0.5, Cosine Score: 0.3995532190777261, Weighted Score: 0.564621439734196

Data: User name is Alfredo, Time Score: 0.982491033540339, Count Score: 1.0, Cosine Score: 0.6152483929709741, Weighted Score: 0.8032469548705718

Data: User went to the azores on vacation, Time Score: 0.9895455402135923, Count Score: 0.5, Cosine Score: 0.46173705741155113, Weighted Score: 0.6032549137591736

Data: User's dog is sick, Time Score: 0.9895455758991448, Count Score: 0.5, Cosine Score: 0.7163711893975937, Weighted Score: 0.7305719886735831 SCORES FOR THE DATA END

Relevant memories:

Memory: User name is Alfredo, Score: 0.8032

Memory: User's dog is sick, Score: 0.7305

Memory: Has a pet dog named Max, Score: 0.7019

Memory: Max is a golden retriever, Score: 0.6700

Memory: User went to the azores on vacation, Score: 0.6032

END RECALL PROCESS

Bot: That's great news that the vet gave you some reassurance about Max's condition! I hope he continues to improve soon.

You: I have also started to search for a female of the same breed to have baby dogs

START RECALL PROCESS

Max Count: 2

Oldest Timestamp: 1716822645.241088

Current timestamp: 1716910538.749593

Max time difference: 87893.50850510597

SCORES FOR THE DATA START

Data: User play tennis since he was 10 years old, Time Score: 0.9048374180359595, Count Score: 0.5, Cosine Score: 0.47554592830953657, Weighted Score: 0.5889823186637582

Data: Has a pet dog named Max, Time Score: 0.9048374495996713, Count Score: 0.5, Cosine Score: 0.7175751026622613, Weighted Score: 0.7099969137310485

Data: Max is a golden retriever, Time Score: 0.904837480863696, Count Score: 0.5, Cosine Score: 0.6729212024309102, Weighted Score: 0.6876699714313791

Data: User likes video games, Time Score: 0.9593621861172695, Count Score: 0.5, Cosine Score: 0.4420512738340673, Weighted Score: 0.585866183446351

Data: Likes pokemon and zelda, big fan of nintendo, Time Score: 0.959362220419901, Count Score: 0.5, Cosine Score: 0.43759602014947097, Weighted Score: 0.5836385651797107

Data: Has a ps4 and a switch, Time Score: 0.9593622539707144, Count Score: 0.5, Cosine Score: 0.40077298071485756, Weighted Score: 0.5652270538501074

Data: User name is Alfredo, Time Score: 0.9824664473838757, Count Score: 1.0, Cosine Score: 0.6236390650059803, Weighted Score: 0.807436144348959

Data: User went to the azores on vacation, Time Score: 0.9895186258017071, Count Score: 0.5, Cosine Score: 0.4376580233801851, Weighted Score: 0.5912086681405193

Data: User's dog is sick, Time Score: 0.9895186614754432, Count Score: 0.5, Cosine Score: 0.7162236135153693, Weighted Score: 0.7304914721265454

SCORES FOR THE DATA END

Relevant memories:

Memory: User name is Alfredo, Score: 0.8074

Memory: User's dog is sick, Score: 0.7304

Memory: Has a pet dog named Max, Score: 0.7099

Memory: Max is a golden retriever, Score: 0.6876

Memory: User went to the azores on vacation, Score: 0.5912

END RECALL PROCESS

Bot: That's good news that you want to find offspring for Max. Is there a reason why you are looking for a golden retriever?

You: I want a puppy of the same breed. I have discovered my english teacher has a female.

START RECALL PROCESS

Max Count: 2

Oldest Timestamp: 1716822645.241088

Current timestamp: 1716910623.977165

Max time difference: 87978.73607707024

SCORES FOR THE DATA START

Data: User play tennis since he was 10 years old, Time Score: 0.9048374180359596, Count Score: 0.5, Cosine Score: 0.4846785442152717, Weighted Score: 0.5935486266166258

Data: Has a pet dog named Max, Time Score: 0.9048374495690946, Count Score: 0.5, Cosine Score: 0.7379442787086505, Weighted Score: 0.7201815017465989

Data: Max is a golden retriever, Time Score: 0.9048374808028329, Count Score: 0.5, Cosine Score: 0.6899807675636422, Weighted Score: 0.6961997539825293

Data: User likes video games, Time Score: 0.959307807522068, Count Score: 0.5, Cosine Score: 0.46139850047673264, Weighted Score: 0.5955262021188833

Data: Likes pokemon and zelda, big fan of nintendo, Time Score: 0.9593078417895271, Count Score: 0.5, Cosine Score: 0.47149079748533074, Weighted Score: 0.6005723591900471

Data: Has a ps4 and a switch, Time Score: 0.9593078753059391, Count Score: 0.5, Cosine Score: 0.4072147668941278, Weighted Score: 0.5684343522735487

Data: User name is Alfredo, Time Score: 0.9823881115926395, Count Score: 1.0, Cosine Score: 0.633597324040344, Weighted Score: 0.8123956899183319

Data: User went to the azores on vacation, Time Score: 0.9894328721721197, Count Score: 0.5, Cosine Score: 0.44088515246810983, Weighted Score: 0.5928007942770849

Data: User's dog is sick, Time Score: 0.989432907808209, Count Score: 0.5, Cosine Score: 0.6938353490766676, Weighted Score: 0.719275901490386

SCORES FOR THE DATA END

Relevant memories:

Memory: User name is Alfredo, Score: 0.8123

Memory: Has a pet dog named Max, Score: 0.7201

Memory: User's dog is sick, Score: 0.7192

Memory: Max is a golden retriever, Score: 0.6961

Memory: Likes pokemon and zelda, big fan of nintendo, Score: 0.6005

END RECALL PROCESS

Bot: That's exciting news about the potential new addition to your family, Alfredo! Do you think Max will be happy?

You: In the next english class I will go with Max to meet the english teacher golden retriever

START RECALL PROCESS

Max Count: 2

Oldest Timestamp: 1716822645.241088

Current timestamp: 1716910694.443031

Max time difference: 88049.2019431591

SCORES FOR THE DATA START

Data: User play tennis since he was 10 years old, Time Score: 0.9048374180359595, Count Score: 0.5, Cosine Score: 0.4942990300556948, Weighted Score: 0.5983588695368373

Data: Has a pet dog named Max, Time Score: 0.9048374495438586, Count Score: 0.5, Cosine Score: 0.7236529760038932, Weighted Score: 0.7130358503879113

Data: Max is a golden retriever, Time Score: 0.9048374807526005, Count Score: 0.5, Cosine Score: 0.7301411575572269, Weighted Score: 0.7162799489667635

Data: User likes video games, Time Score: 0.9592629293074183, Count Score: 0.5, Cosine Score: 0.484220014187688, Weighted Score: 0.6069257394206986

Data: Likes pokemon and zelda, big fan of nintendo, Time Score: 0.9592629635458514, Count Score: 0.5, Cosine Score: 0.5223293552790245, Weighted Score: 0.6259804185259751

Data: Has a ps4 and a switch, Time Score: 0.9592629970338733, Count Score: 0.5, Cosine Score: 0.4271603565221469, Weighted Score: 0.5783959275195418

Data: User name is Alfredo, Time Score: 0.9823234630581618, Count Score: 1.0, Cosine Score: 0.6570340266604184, Weighted Score: 0.8240978790947496

Data: User went to the azores on vacation, Time Score: 0.9893621023319956, Count Score: 0.5, Cosine Score: 0.4556343969581823, Weighted Score: 0.6001577240620901

Data: User's dog is sick, Time Score: 0.9893621379370185, Count Score: 0.5, Cosine Score: 0.5883061762790698, Weighted Score: 0.6664936226237895

SCORES FOR THE DATA END

Relevant memories:

Memory: User name is Alfredo, Score: 0.8240

Memory: Max is a golden retriever, Score: 0.7162

Memory: Has a pet dog named Max, Score: 0.7130

Memory: User's dog is sick, Score: 0.6664

Memory: Likes pokemon and zelda, big fan of nintendo, Score: 0.6259

END RECALL PROCESS

Bot: That's so cool. I'm sure your English teacher will like Max.

You: By the way, I have just bought a new videogame

START RECALL PROCESS

Max Count: 2

Oldest Timestamp: 1716822645.241088

Current timestamp: 1716910718.452508

Max time difference: 88073.2114200592

SCORES FOR THE DATA START

Data: User play tennis since he was 10 years old, Time Score: 0.9048374180359595, Count Score: 0.5, Cosine Score: 0.5118821049908091, Weighted Score: 0.6071504070043945

Data: Has a pet dog named Max, Time Score: 0.9048374495352692, Count Score: 0.5, Cosine Score: 0.7102215593637119, Weighted Score: 0.7063201420656733

Data: Max is a golden retriever, Time Score: 0.9048374807355035, Count Score: 0.5, Cosine Score: 0.6984059655923132, Weighted Score: 0.7004123529800325

Data: User likes video games, Time Score: 0.9592476550626778, Count Score: 0.5, Cosine Score: 0.5477771810976251, Weighted Score: 0.638700504314482

Data: Likes pokemon and zelda, big fan of nintendo, Time Score: 0.9592476892912322, Count Score: 0.5, Cosine Score: 0.5661680059982518, Weighted Score: 0.647895925321934

Data: Has a ps4 and a switch, Time Score: 0.9592477227695919, Count Score: 0.5, Cosine Score: 0.47260022991236367, Weighted Score: 0.6011120456485798

Data: User name is Alfredo, Time Score: 0.9823014602901362, Count Score: 1.0, Cosine Score: 0.6575713469766888, Weighted Score: 0.8243610385608785

Data: User went to the azores on vacation, Time Score: 0.9893380163040543, Count Score: 0.5, Cosine Score: 0.46445971416304355, Weighted Score: 0.6045643611575353

Data: User's dog is sick, Time Score: 0.9893380518985043, Count Score: 0.5, Cosine Score: 0.5678933311196532, Weighted Score: 0.6562811785344527

SCORES FOR THE DATA END

Relevant memories:

Memory: User name is Alfredo, Score: 0.8243

Memory: Has a pet dog named Max, Score: 0.7063

Memory: Max is a golden retriever, Score: 0.7004

Memory: User's dog is sick, Score: 0.6562

Memory: Likes pokemon and zelda, big fan of nintendo, Score: 0.6478

END RECALL PROCESS

Bot: What's the new videogame you got? Is it another adventure in Hyrule or a quest to catch 'em all in Pokémon?

You: It is not about zelda or pokemon this time, it is a new videogame from super mario

START RECALL PROCESS

Max Count: 2

Oldest Timestamp: 1716822645.241088

Current timestamp: 1716910736.781345

Max time difference: 88091.54025697708

SCORES FOR THE DATA START

Data: User play tennis since he was 10 years old, Time Score: 0.9048374180359595, Count Score: 0.5, Cosine Score: 0.5317588678840469, Weighted Score: 0.6170887884510133

Data: Has a pet dog named Max, Time Score: 0.9048374495287154, Count Score: 0.5, Cosine Score: 0.6706674684492792, Weighted Score: 0.6865430966068184

Data: Max is a golden retriever, Time Score: 0.9048374807224578, Count Score: 0.5, Cosine Score: 0.6475755256576248, Weighted Score: 0.6749971330094269

Data: User likes video games, Time Score: 0.95923600047057, Count Score: 0.5, Cosine Score: 0.6779947682301726, Weighted Score: 0.7038063842327288

Data: Likes pokemon and zelda, big fan of nintendo, Time Score: 0.9592360346915868, Count Score: 0.5, Cosine Score: 0.7013639742011277, Weighted Score: 0.7154909957734605

Data: Has a ps4 and a switch, Time Score: 0.9592360681625742, Count Score: 0.5, Cosine Score: 0.5638779582073776, Weighted Score: 0.6467479961443323

Data: User name is Alfredo, Time Score: 0.982284671779342, Count Score: 1.0, Cosine Score: 0.6342170315456483, Weighted Score: 0.8126796837176596

Data: User went to the azores on vacation, Time Score: 0.989319638259718, Count Score: 0.5, Cosine Score: 0.501402561094726, Weighted Score: 0.6230311901122925

Data: User's dog is sick, Time Score: 0.9893196738461011, Count Score: 0.5, Cosine Score: 0.5455541356644317, Weighted Score: 0.6451069862937411

SCORES FOR THE DATA END

Relevant memories:

Memory: User name is Alfredo, Score: 0.8126

Memory: Likes pokemon and zelda, big fan of nintendo, Score: 0.7154

Memory: User likes video games, Score: 0.7038

Memory: Has a pet dog named Max, Score: 0.6865

Memory: Max is a golden retriever, Score: 0.6749

END RECALL PROCESS

Bot: That's awesome that you're trying out a new Mario game! Are you looking forward to exploring the Mushroom Kingdom and rescuing Princess Peach from Bowser's clutches?

You: For sure! try to guest for which console of mine I have bought it

START RECALL PROCESS

Max Count: 2

Oldest Timestamp: 1716822645.241088

Current timestamp: 1716910783.559792

Max time difference: 88138.31870412827

SCORES FOR THE DATA START

Data: User play tennis since he was 10 years old, Time Score: 0.9048374180359595, Count Score: 0.5, Cosine Score: 0.4989408307623914, Weighted Score: 0.6006797698901856

Data: Has a pet dog named Max, Time Score: 0.904837449512001, Count Score: 0.5, Cosine Score: 0.46475189061921196, Weighted Score: 0.5835853076876062

Data: Max is a golden retriever, Time Score: 0.9048374806891877, Count Score: 0.5, Cosine Score: 0.43887094416725486, Weighted Score: 0.5706448422559244

Data: User likes video games, Time Score: 0.9592062784947706, Count Score: 0.5, Cosine Score: 0.7710418396996003, Weighted Score: 0.7503224894734928

Data: Likes pokemon and zelda, big fan of nintendo, Time Score: 0.9592063126965651, Count Score: 0.5, Cosine Score: 0.7675454115503669, Weighted Score: 0.7485742839493248

Data: Has a ps4 and a switch, Time Score: 0.9592063461487516, Count Score: 0.5, Cosine Score: 0.6718253134384613, Weighted Score: 0.7007142432564185

Data: User name is Alfredo, Time Score: 0.9822418574680365, Count Score: 1.0, Cosine Score: 0.596179970383924, Weighted Score: 0.7936504495589711

Data: User went to the azores on vacation, Time Score: 0.9892727704197604, Count Score: 0.5, Cosine Score: 0.4981455093109811, Weighted Score: 0.6213909472604306

Data: User's dog is sick, Time Score: 0.9892728059855713, Count Score: 0.5, Cosine Score: 0.45202031987660707, Weighted Score: 0.5983283614346964

SCORES FOR THE DATA END

Relevant memories:

Memory: User name is Alfredo, Score: 0.7936

Memory: User likes video games, Score: 0.7503

Memory: Likes pokemon and zelda, big fan of nintendo, Score: 0.7485

Memory: Has a ps4 and a switch, Score: 0.7007

Memory: User went to the azores on vacation, Score: 0.6213

END RECALL PROCESS

Bot: I'm excited to hear that you've got a new Mario game! Since I recall you have two consoles, is this Super Mario game exclusive to either your PS4 or Switch?

You: It is for the switch as it is a nintendo exclusive. I am a big fan of nintendo games

START RECALL PROCESS

Max Count: 2

Oldest Timestamp: 1716822645.241088

Current timestamp: 1716910896.312092

Max time difference: 88251.0710041523

SCORES FOR THE DATA START

Data: User play tennis since he was 10 years old, Time Score: 0.9048374180359595, Count Score: 0.5, Cosine Score: 0.462767811524281, Weighted Score: 0.5825932602711303

Data: Has a pet dog named Max, Time Score: 0.9048374494717861, Count Score: 0.5, Cosine Score: 0.4356293998296891, Weighted Score: 0.5690240622827911

Data: Max is a golden retriever, Time Score: 0.9048374806091399, Count Score: 0.5, Cosine Score: 0.4169811875581133, Weighted Score: 0.5596999639313416

Data: User likes video games, Time Score: 0.9591347714899837, Count Score: 0.5, Cosine Score: 0.7259657544197284, Weighted Score: 0.7277665700823601

Data: Likes pokemon and zelda, big fan of nintendo, Time Score: 0.9591348056455345, Count Score: 0.5, Cosine Score: 0.7472641661235684, Weighted Score: 0.7384157844731678

Data: Has a ps4 and a switch, Time Score: 0.9591348390524909, Count Score: 0.5, Cosine Score: 0.6834648127867256, Weighted Score: 0.7065161161564855

Data: User name is Alfredo, Time Score: 0.9821388543069329, Count Score: 1.0, Cosine Score: 0.5527685546878414, Weighted Score: 0.7719189909206539

Data: User went to the azores on vacation, Time Score: 0.9891600159602179, Count Score: 0.5, Cosine Score: 0.46923244721705026, Weighted Score: 0.6069062275985796

Data: User's dog is sick, Time Score: 0.9891600514765404, Count Score: 0.5, Cosine Score: 0.42145466698115425, Weighted Score: 0.5830173463597123

SCORES FOR THE DATA END

Relevant memories:

Memory: User name is Alfredo, Score: 0.7719

Memory: Likes pokemon and zelda, big fan of nintendo, Score: 0.7384

Memory: User likes video games, Score: 0.7277

Memory: Has a ps4 and a switch, Score: 0.7065

Memory: User went to the azores on vacation, Score: 0.6069

END RECALL PROCESS

START MEMORIZATION PROCESS

Turns of the conversation to memorize: ["You: I have just come from a walk with my dog Max", "You: I felt him a bit slower than usual", "You: We went to the veterinary yesterday and he told me it is normal because of the pills, but he will be good soon!", "You: I have also started to search for a female of the same breed to have baby dogs ", "You: I want a puppy of the same breed. I have discovered my english teacher has a female.", "You: In the next english class I will go with Max to meet the english teacher golden retriever", "You: By the way, I have just bought a new videogame", "You: It is not about zelda or pokemon this time, it is a new videogame from super mario", "You: For sure! try to guest for which console of mine I have bought it", "You: It is for the switch as it is a nintendo exclusive. I am a big fan of nintendo games"]

Answer from the LLM in charge of extracting memories:

/p User has a dog called Max. /p

/p Max is taking pills but will be better soon. /p

/p User wants a puppy of Max and will take Max to meet his english teacher golden retriever.
/p

/p User has recently purchased a new super Mario videogame. /p

/p He is a big fan of nintendo games and has a switch for the nintendo exclusives. /p

Extracted Memories:

User has a dog called Max and is slower than usual.

Max is taking pills but will be better soon.

User wants a puppy of Max and will take Max to meet his english teacher golden retriever.

User has recently purchased a new super Mario videogame.

User is a big fan of nintendo games and has a switch for the nintendo exclusives.

START OF UPDATING/STORING MEMORIES

Updated memorie data:: Has a pet dog named Max

Inserted memorable data: Max is taking pills but will be better soon.

Inserted memorable data: User wants a puppy of Max and will take Max to meet his english teacher golden retriever.

Inserted memorable data: User has recently purchased a new super Mario videogame.

Inserted memorable data: He is a big fan of nintendo games and has a switch for the nintendo exclusives.

END OF UPDATING/STORING MEMORIES

END MEMORIZATION PROCESS

Bot: So excited about your new Nintendo Switch game! How's it feeling to be playing Mario on the same console you took with you on your vacation in the Azores?