Politecnico di Torino

POLITECNICO DI TORINO

Master's Degree in Computer Engineering
A.Y. 2024/2025
Degree Session July 2025

Generative Artificial Intelligence for Insurance: developing a chatbot for Multi-Document Summarization of Insurance Products

Supervisor

Prof. PAOLO GARZA

Company Tutor

ALESSANDRO DANESI

Candidate

IVAN BRANCATI

Table of Contents

List of Tables

List of Figures

Acronyms

A

Artificial Intelligence (AI)

B

Bag of Words (BoW)

BERT for Summarization (BERTSUM)

Bidirectional Encoder Representations from Transformers (BERT)

Bilingual Evaluation Understudy (BLEU)

C

Chain-of-Event (CoE)

Chain-of-Thought (CoT)

Chinese Call Center Textual Corpus (CCTC)

Convolutional Neural Network (CNN)

F

Facebook AI Similarity Search (FAISS)

G

Generative Adversarial Network (GAN)

Generative Artificial Intelligence (GenAI)

Generative Pre-trained Transformer (GPT)

Grammatical Error Correction (GEC)

H

Hidden Markov Model (HMM)

I

Information Retrieval (IR)

Inverse Document Frequency (IDF)

L

Large Language Model (LLM)

Latent Dirichlet Allocation (LDA)

Latent Semantic Analysis (LSA)

Long Short-Term Memory (LSTM)

Longformer-Encoder-Decoder (LED)

M

Maximum Marginal Relevance (MMR)

Multi-Document Summarization (MDS)

N

Named Entity Recognition (NER)

Natural Language Generation (NLG)

Natural Language Processing (NLP)

Natural Language Understanding (NLU)

Non-Negative Matrix Factorization (NMF)

Q

Question Answering (QA)

R

Recall-Oriented Understudy for Gisting Evaluation (ROUGE)

Recurrent Neural Network (RNN)

Reinforcement Learning from Human Feedback (RLHF)

Retrieval-Augmented Generation (RAG)

T

Term Frequency (TF)

V

Variational Autoencoder (VAE)

W

Wikipedia Current Events Portal (WCEP)

Chapter 1

Introduction

Generative Artificial Intelligence (GenAI) represents the leading technology today. It has rapidly evolved in recent years and is still gaining outstanding advancements, especially with the success of **Large Language Models (LLMs)**, such as OpenAI's ChatGPT, which can perform almost every task.

The term Artificial Intelligence dates back to 1956 and was coined by John McCarthy. He formed a small group of scientists and organized the *Dartmouth Summer Research Project on Artificial Intelligence* [1], which can be considered the birth of this field of research. Alan Turing, back in 1950, already prepared the ground for this new theme in his work *Computing Machinery and Intelligence*[2]. Although the beginning of this area of study originated in the 1950s, the results in the next 40 years can be considered disappointing, producing only rudimentary systems to solve mathematical problems, play checkers, or translate texts. Only with the 1990s and the increasing hardware computing power did the interest in **Machine Learning** grow again, paving the way to the **Deep Learning** explosion of the 2010s. The year 2019 was the beginning of what can be defined as the Large Language Models and GenAI era: OpenAI published its LLM GPT, showing the extraordinary capabilities of these language models and what they can generate.

The growing impact of GenAI on society is evident in almost every field, but particularly in **Natural Language Processing (NLP)**, one of the fields where it has shown the most remarkable results. NLP is a core subfield of Artificial Intelligence (AI) that focuses on enabling machines to understand and process human language. It has applications across a wide range of domains and industries.

Among the various NLP techniques, **Multi-Document Summarization (MDS)** stands as one of the most relevant, especially in today's data-driven world, where individuals often deal with many documents containing large volumes of information on a specific topic. Despite its potential infinite applications, only a

few domains have been widely explored. Among them, **News Summarization** has emerged as one of the most extensively studied, and numerous works have been published about it.

On the other hand, MDS in the **insurance sector** remains quite unexplored, even though customers often deal with extensive and complex documentation. This could represent a huge barrier since these documents could be very complex for non-experts, proving a big need for automatic tools with the power to simplify and clarify insurance documents.

The main objective of this thesis is to demonstrate that MDS techniques can be effectively adapted and applied within the insurance domain. The thesis will first analyze the theoretical aspect of this topic, and then move to the creation of a real MDS pipeline, integrating it into a functional chatbot capable of summarizing, comparing, and responding to user-specific questions about insurance products.

Another contribution of this research is the creation of a custom dataset to test MDS in the insurance domain, made by information documents about insurance products from the Unipol group, retrieved from their website. The proposed solution utilizes LangChain APIs, in combination with OpenAI's GPT LLM, and tailored prompts to be finally deployed through a Streamlit-based web application. The resulting chatbot also utilizes the **Retrieval-Augmented Generation (RAG)** technique to provide accurate responses by directly accessing data from the dataset documents, instead of responding with information only from the LLM knowledge base.

This thesis is composed of five chapters:

- The first chapter - this one - is an introduction to the themes faced in the thesis.

- The second chapter, the **State of the Art**, provides an overview of the currently most relevant techniques in the technologies involved in the thesis. The first theme is Natural Language Processing, describing the most meaningful NLP tasks, the most commonly used techniques, and the most advanced models. Next, GenAI is explored, starting with a quick overview of its history and then describing the top-performing models, but also listing the advantages, limitations, and applications of this technology. After that, the focus moves on Large Language Models, especially on GPT, exploring how they work, the most used architectures and models, and their applications, advantages, and challenges. Finally, the attention moves towards the state of the art of the specific tasks applied in the thesis; therefore, Multi-Document Summarization, Retrieval-Augmented Generation, Financial-related MDS works, and the

Chain-of-Event prompting technique are explored to provide overviews on these topics.

- The third chapter introduces the **Use Case**, explaining why a Multi-Document Summarization pipeline is fundamental in the insurance field and defining the motivations and challenges its development faces. It first presents the details of the initial experiments conducted in the financial domain, before introducing the evaluation metrics and financial dataset employed. Later, the final product is analyzed at a high level.

- The fourth chapter details the **Design and Implementation** of the final solution, covering both the high-level system architecture and the technical details of its implementation. The design strategy lists the architecture and the technologies utilized to produce it, displaying the operations flow and summarization pipeline produced, and finally provides details about the user Interface design. The implementation section covers all the details about the summarization pipeline, prompts, and RAG. Moreover, it also explores chatbot interface implementations.

- The fifth chapter presents the **Experimental Evaluation**, including comparisons between different configurations and the analysis of the obtained results. Initially, the experimental setups that led to the final solution are presented, and finally, the results are compared with all the utilized evaluation metrics.

- The sixth chapter is the **Conclusion** of the thesis and analyzes its outcomes, taking stock of what has been done, but also exploring possible evolutions and directions for future refinements and developments.

Chapter 2

State of the Art

Artificial Intelligence (AI) has become our most widely discussed topic. This technology is revolutionizing many aspects of our lives more than anything else, and its applications are visible across multiple sectors and fields.

One of the implementations where AI makes a significant difference is *Natural Language Processing* (NLP). This chapter will begin with an analysis of the state of the art in NLP and its related tasks. Then Generative Artificial Intelligence will also be considered, together with the accelerated advancements it already made, especially in *Large Language Models* (LLM), with a specific focus on the *Generative Pre-trained Transformer* (GPT), a model developed by OpenAI which rapidly became the most recognized GenAI model.

The chapter will then explore the field of *Multi-Document Summarization* (MDS), the central topic of this thesis. Lastly, *Retrieval Augmented Generation* (RAG), a technique that has gained increasing attention in recent years and that is widely adopted, will be discussed.

2.1 Natural Language Processing

According to Liddy (2018)[3], «Natural Language Processing is a theoretically motivated range of computational techniques for analyzing and representing naturally occurring texts at one or more levels of linguistic analysis to achieve human-like language processing for a range of tasks or applications.»

NLP is a discipline focused on developing machines capable of processing human language, in both its written and spoken forms, as well as data that resembles it. This science aims at facilitating interactions between humans and machines, reducing the gap between two traditionally distinct disciplines: Linguistics and Computer Science.

This topic has long been studied within computer science, but it has gained even

more importance today as Machine Learning and AI drive technological evolution. Human-machine interactions are becoming increasingly frequent, with users now expecting machines to respond as naturally as human beings would.

NLP can generally be divided into two main categories, though in practice these often overlap or are used together depending on the task:

- **Natural Language Understanding (NLU)** aims to facilitate machines to interpret human language in a similar way to how people do. It faces significant challenges due to the intrinsic ambiguity and complexity of natural language;
- **Natural Language Generation (NLG)** focuses on creating human-readable text from structured or unstructured data. It makes possible for machines to generate natural language responses that make sense to users.

In simple terms, NLU is about understanding language, while NLG is about creating it in a way that feels natural to human users.

NLP's importance has risen largely because of the massive volume of data generated across all the communication channels that are available today, much of which exists in natural language form. While these resources have great potential for a variety of applications, their vast numerosity makes manual processing almost impossible.

2.1.1 Key NLP Tasks

Language-related tasks play a crucial role in everyday life, influencing diverse areas as medicine, sales, and even web browsing. The following subsections will explore some of the most common tasks that leverage NLP, providing a detailed analysis of each.

Sentiment Analysis

Sentiment Analysis is a NLP technique used to determine the tone or the emotion communicated in a certain text. Typically, it takes a piece of text as input and outputs a probability distribution indicating the likelihood of the text expressing a positive, neutral, or negative sentiment. The input can vary widely, including articles, reviews, or social media posts, and the analysis may also identify more complex emotions such as anger, happiness, or sadness.

Various models can be employed for this task. Supervised models are trained on annotated datasets to recognize specific words or patterns linked to sentiments. Lexical models, on the other hand, rely on predefined dictionaries on positive and negative words. Additionally, a hybrid approach combining Machine Learning techniques and rule-based methods can be utilized. The features used to determine

the sentiment probability can either be manually generated, such as word n-grams or TF-IDF, or derived from Deep Learning models that capture both long-term and short-term dependencies in the text.

Sentiment Analysis has numerous real-world applications, including:

- **Social Media Analysis**: Sentiment Analysis monitors public opinion about a brand, event, or public figure. For example, platforms like X (formerly Twitter) leverage this technology to identify trends and monitor public sentiment;

- **Customer Support**: analyzing the tone of customer interactions enables companies to tailor their responses and provides more personalized support, improving customer satisfaction;

- **Products and Services Reviews**: businesses use Sentiment Analysis to understand customer perception of their products (see Figure 2.1). For instance, Amazon applies this technique not only to measure customer opinions but also to identify fake reviews;

- **Toxicity Classification**: this specialized application of Sentiment Analysis identifies content that may be violent, offensive, or contain inappropriate language. The output will be the probability of the text belonging to one of the toxicity classes. Toxicity Classification is crucial in the modern age to moderate online interactions and create healthier digital communities.

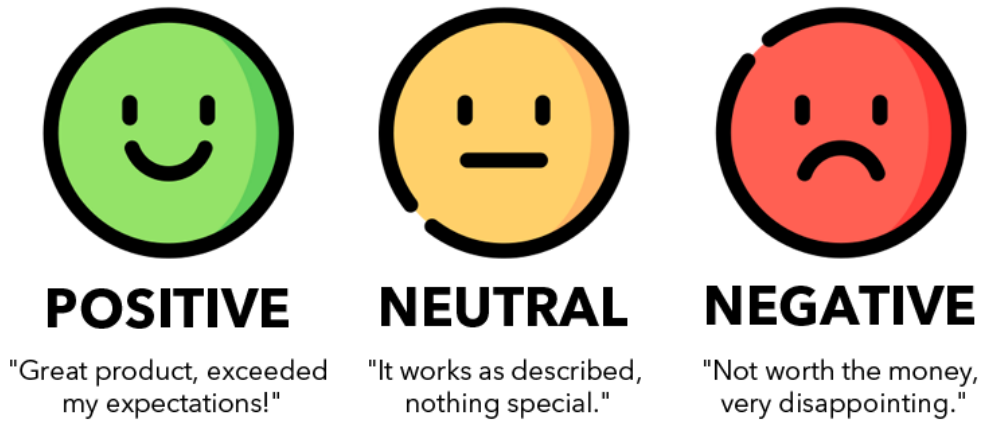


Figure 2.1: An example of sentiment analysis of some reviews.

Machine Translation

Machine Translation is a NLP task that focuses on automatically converting text from one language to another, ensuring that the original meaning is preserved while producing fluent and natural-sounding text in the target language. The objective is to achieve translations without human intervention, recognize patterns, and address challenges related to language ambiguities and complex linguistic rules. Many systems can automatically detect the source language, without requiring manual input, simplifying the translation process.

Various widely used tools leverage Machine Translation, with Google Translate being the most popular automatic translation service worldwide. Social media platforms like Facebook and Skype also utilize this technique to enable real-time conversations between users speaking different languages.

Named Entity Recognition

Named Entity Recognition (NER) is a NLP task in which the objective is to assign named entities in a text to predefined categories, such as quantities, personal names, locations, time expressions, or organizations. The input usually is a stream of text and the output is made by the entities, their categories, and also their starting and ending positions in the text. This task is very helpful in understanding quickly what the main topic of the text is, and this could be fundamental in activities such as summarizing news articles.

Although it may seem easy to distinguish the words in the text that are proper names, sometimes this is not true; for example, capital letters could be a hint to deal with names, but they are used also in other places, like at the beginning of a paragraph, and many languages do not use capitalization for names at all.

NER is generally based on statistical models, but also Machine Learning techniques or Neural Networks could be involved. It can be divided into three phases:

- **Tokenization**: the text is split into single words or tokens;
- **Tagging**: labels are assigned to words or tokens, to define if they are entities or not;
- **Classification**: the entities are assigned to the predefined categories.

Text Classification

Text Classification is a fundamental task in NLP, where the goal is to automatically assign a text to a specific label or a category based on its content. There are three possible types of classification:

- **Binary Classification**: the text is assigned to one of two possible categories;

- **Multiclass Classification**: the text is classified into one of multiple possible categories;
- **Multilabel Classification**: the text can belong to multiple categories simultaneously, as the categories are not mutually exclusive.

Text classification can utilize traditional Machine Learning approaches, such as Support Vector Machines, Naïve Bayes, and Random Forests. More advanced techniques include Deep Learning models, like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), as well as innovative Transformer Models, such as Bidirectional Encoder Representations from Transformers (BERT) or Generative Pre-trained Transformer (GPT).

Spam Detection is a binary text classification task where models analyze not only the content of emails but also additional details such as the subject of the email and the sender's address to classify emails as either spam or legitimate. Leading email providers implement these detectors to improve user experience by filtering out unwanted advertisements or phishing attacks.

Topic Modeling

Topic Modeling is a NLP technique used to uncover hidden topics within large collections of unstructured text. This unsupervised task takes a collection of documents as input, groups words into coherent topics and generates a list of these topics as its output.

The most widely used technique for Topic Modeling is Latent Dirichlet Allocation (LDA), which treats each document as a collection of topics and each topic as a collection of words. An alternative approach is Non-Negative Matrix Factorization (NMF), which decomposes the text data into matrices representing the relationships between words and topics.

Unlike classification, Topic Modeling does not rely on predefined labels; instead, it discovers patterns and themes within large collections of documents, where topics are not known in advance. A practical application of Topic Modeling can be found in the legal field, where lawyers use it to efficiently search through vast amounts of legal documents to identify relevant evidence or themes related to a case.

Text Generation

Text Generation is a core application of NLP that focuses on producing text in natural language. The objective is to produce content that is both grammatically and semantically accurate, resembling what a human might write. Generated text can have various forms, including articles, programming code, social media posts, and more.

Various technologies are employed for text generation, including RNNs, Long Short-Term Memory (LSTM) networks, and Transformer models. In today's digital landscape, many companies rely on automatic text generation to produce content for their websites or social media platforms, which is why it has become an essential tool to meet their content creation demands.

One notable application of Text Generation is Autocomplete, where the system predicts the next word or completes a partially typed word, facilitating faster and more efficient text input for the user. Modern chat applications like WhatsApp and Telegram utilize autocompletion models to assist users, while Google employs this technology to predict search queries.

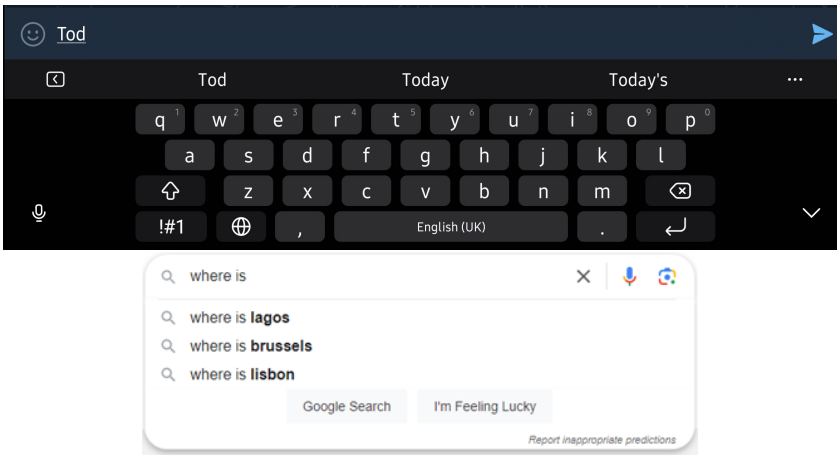


Figure 2.2: The use of autocomplete features in the Telegram messaging application and the Google search bar.

Chatbots

While chatbots can be viewed as a specific case of Text Generation, their growing notoriety has led them to be considered a distinct NLP task. A Chatbot is a NLP application that interacts with humans automatically, trying to reply to users' questions most similarly to what a person would do.

The term "chatterbot" was invented by Michael Mauldin in 1994 with the creation of Verbot. However, the concept of chatbot dates back to 1950 when Alan Turing introduced the Turing Test in his paper "Computing Machinery and Intelligence"[2]. Notable early examples include ELIZA, created by Joseph Weizenbaumin in 1966, and Dr. Sbaitso, developed in 1991 by Creative Labs (see Figure 2.3). Recent advancements have brought high-quality chatbots like OpenAI ChatGPT, Microsoft Copilot, and Google Gemini into the spotlight, making them largely recognized and utilized.

It is possible to distinguish two categories of chatbots:

- **Rule-based chatbots**: these chatbots operate following some predefined rules, responding only to some specific inputs. Their biggest limitation is the inability to understand complex language or variation in user queries;
- **Artificial Intelligence chatbots**: leveraging Machine Learning, Deep Learning, and NLP models, these chatbots excel at understanding user queries and continuously improve through learning, adapting their responses to better meet user needs over time.

Chatbots have a wide range of applications, including customer service to address user issues quickly, answering product-related queries on e-commerce websites, providing support or entertainment in everyday situations, and even offering preliminary medical advice based on symptoms.

This topic will be explored in greater detail in the following chapters, as the final goal of this thesis is to develop a chatbot designed to assist users in making informed insurance product choices.

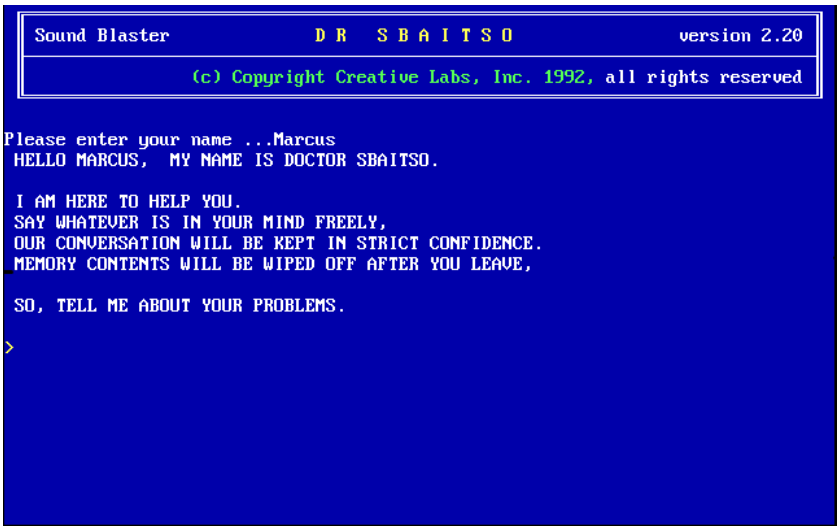


Figure 2.3: Dr. Sbaitso, one of the first chatbots, was developed by Creative Labs in 1991. It could be used on computers running MS-DOS operative system.

Summarization

Summarization is a NLP task that takes one or more blocks of text as input and generates a condensed version that highlights the most important information, facilitating a clearer understanding of the original content.

This is also a very employed NLP task, and it can be categorized into two main types:

- **Extractive Summarization**: this method selects and combines the most important sentences from the original text to create the summary. Since sentences remain unaltered, the original meaning is preserved; however, the result may occasionally appear fragmented;
- **Abstractive Summarization**: in this approach, the summary is obtained by rephrasing the original content, instead of just extracting sentences. This method is closer to the way a human would summarize a text, resulting in more fluent and natural-sounding content, but it may occasionally introduce semantic errors if the original meaning is misinterpreted.

This task has numerous applications and relies on a diverse range of technologies, which will be analyzed in greater detail in the next chapters, as Summarization is one of the key topics of this thesis.

Question Answering

Question Answering (QA) is another important NLP task that involves providing accurate responses to questions posed in natural language. The objective is not only to generate answers but also to grab relevant information from various sources, such as documents and databases, to construct a complete response.

QA can be split into two main categories:

- **Closed-domain QA**: this type of QA focuses on questions within a specific domain, often presented as multiple-choice problems where the model has to pick the correct answer from a given set of options;
- **Open-domain QA**: this approach handles questions on a wide range of topics, requiring the model to find relevant information from external sources, such as web pages. The challenge is greater because the model must first understand the context of the question before formulating an appropriate answer.

Multiple technologies, including Transformer models, RNNs, and LSTM networks, are employed to develop QA systems.

Information Retrieval

Information Retrieval (IR) is a subfield of NLP that focuses on analyzing large quantities of data, such as files, web pages, scientific papers, or databases, to retrieve a set of documents that are relevant to a user-specific query.

IR can be divided into several phases:

- **Pre-processing**: this phase involves preparing the text by applying techniques such as tokenization, lemmatization, and stop-word removal to transform the raw text into a structured format;
- **Indexing**: the text is organized using an inverted index, mapping keywords to the documents in which they appear, enabling efficient search and retrieval;
- **Query Analysis**: in this phase the user's query is analyzed to understand its intent and match it effectively with relevant terms in the dataset;
- **Document Retrieval**: the system identifies and retrieves the documents that are the most relevant to the user's query;
- **Ranking**: retrieved documents are sorted using ranking algorithms to emphasize the most important results, ensuring that the most relevant information is displayed first;
- **Post-processing**: the search results are refined by removing duplicates, highlighting relevant keywords, and summarizing each document to enhance the user's search experience.

These steps are fundamental to applications like Search Engines and Content Recommendation Systems. Search engines, such as Google, which is surely the most known, let users search for information using natural language, efficiently retrieving accurate results. Content Recommendation Systems analyze user behavior and preferences to suggest personalized content, enhancing user engagement. Today, many e-commerce websites and streaming platforms rely heavily on these systems to provide a more tailored experience (see Figure 2.4).

Grammatical Error Correction Models

The final task analyzed is Grammatical Error Correction (GEC), which focuses on processing sentences that may contain grammatical errors and providing their corrected versions as output. GEC plays an important role today, given the vast amount of text produced daily. These models are essential in reducing the effort required to ensure grammatical accuracy. GEC can correct not only grammar errors but also punctuation, spelling, and word choice to ensure that the most contextually appropriate terms are used.

Typically, the GEC process is seen as a sequence-to-sequence task and can be divided into three main steps:

- **Error Detection**: identifies grammatical errors present in the analyzed text;
- **Error Correction**: once errors are detected, the system suggests improvements by correcting the mistakes or restructuring the sentence;

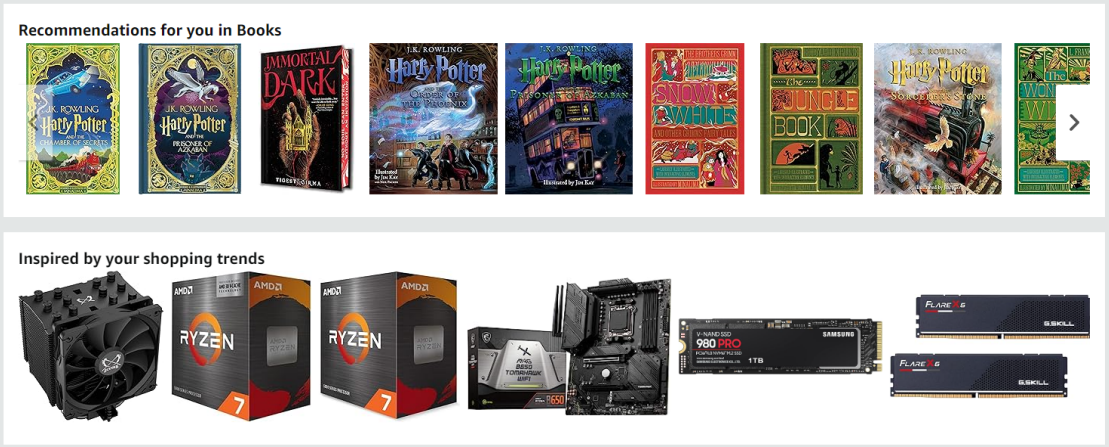


Figure 2.4: Amazon Recommendation System, based on users' views and bought items.

- **Contextual Understanding**: advanced models analyze the context of the text to provide more accurate corrections while preserving the meaning of the phrase.

GEC techniques vary widely, ranging from models that utilize predefined grammar rules and dictionaries to more advanced approaches based on statistical models, Machine Learning, Deep Learning, and Transformer-based architectures.

These models have a big spectrum of applications, including Writing Assistants that improve the quality of user-generated text, Language Learning Tools like Grammarly that help non-native speakers avoid errors, and Data Cleaning tools that eliminate textual inaccuracies that could lower analytical outcomes.

2.1.2 NLP Process Flow

NLP models depend on their ability to identify relationships within text. These models typically follow three key phases: Data Preprocessing, Feature Extraction, and Modeling.

Data Preprocessing

Textual data is often not in a format suitable for direct input into a model. Therefore, preprocessing steps are necessary to transform raw data into a format that the chosen model can efficiently process. This improves data quality, reduces noise, and enhances the performance of subsequent feature extraction and modeling operations.

Various techniques can be utilized for this purpose:

- **Text Normalization**: input text is normalized by removing tags, accented characters, special characters, and extra white spaces. Contractions are expanded and all the text is converted to lowercase to ensure uniformity and prevent words from being treated differently due to case variations;
- **Tokenization**: this process splits the text into smaller units called tokens, which could be words, phrases, or parts of words. Tokenization is essential for breaking down the text into manageable units that can be processed by models. The result is a set of numerical tokens that represent words and can be easily used in various models. For example, the sentence "The autumn breeze whispered through the amber leaves." becomes ["The", "autumn", "breeze", "whispered", "through", "the", "amber", "leaves", "."];
- **Stemming and Lemmatization**: these techniques reduce words to their base or root forms. Stemming cuts word suffices (e.g., "runner" and "running" are reduced to "run"), which can sometimes produce imprecise results. Lemmatization, which relies on a dictionary, identifies the base form of words while considering their context (e.g., "better" becomes "good"). Lemmatization is generally preferred as it produces real words with proper meanings, instead of Stemming, which sometimes can map words into roots with no meaning or not conventional;
- **Stop Word Removal**: Stop Words (e.g., "the", "at", "on") are commonly used but often carry little contextual value. Depending on the task, these words may introduce noise, and thus, removing them can improve the model's focus on more meaningful terms;
- **Sentence Segmentation**: large texts can be split into sentence-level units, each conveying a specific meaning. This task may seem simple but can be challenging in languages where punctuation does not consistently denote sentence boundaries.

Feature Extraction

Most NLP models cannot operate directly on raw text and require features to be extracted. Feature Extraction involves transforming textual data into numerical representations that preserve relevant information while simplifying the complexity of the input.

Various techniques can be employed for feature extraction:

- **Bag of Words (BoW)**: this common technique represents text as a set of words (or n-grams), counting how often each word appears but disregarding word order. Each document is transformed into a vector of word(or n-gram)-frequency pairs, but this can sometimes result in very large vectors;

- **Term Frequency-Inverse Document Frequency (TF-IDF)**: this method refines BoW by assigning weight to each word based on its importance in the document. The Term Frequency (TF) measures how often a word appears in a single document, while Inverse Document Frequency (IDF) reduces the importance of common words across multiple documents. The TF-IDF score is computed as follows:

$$w_{x,y} = \mathrm{tf}_{x,y} \times \log\left(\frac{N}{\mathrm{df}_x}\right) \tag{2.1}$$

  where $\mathrm{tf}_{x,y}$ is the frequency of x in y (TF), df_x is the number of documents containing x, and N is the total number of documents;

- **Word Embeddings**: unlike BoW and TF-IDF, Word Embeddings capture semantic relationships between words by representing them as vectors in a lower-dimensional space. Words with similar meanings are represented by similar vectors, which allows the model to understand complex meanings such as synonyms, analogies, and usage contexts. Popular word embedding techniques include Word2Vec, GloVe, and FastText;

- **Sentence Embeddings**: rather than focusing on individual words, Sentence Embeddings represent entire sentences or documents. This approach captures the global context of the text, making it ideal for tasks requiring deeper language understanding. Well-known sentence embedding techniques include Universal Sentence Encoder and BERT.

Modeling

Once the data have been preprocessed and relevant features extracted, they can be fed into NLP architectures to perform the selected task. These models can be divided into three main categories:

- **Machine Learning and Statistical Methods**: these early models in NLP rely heavily on manual feature engineering. Algorithms such as Logistic regression, Naïve Bayes, and Support Vector Machines are commonly used for tasks like text classification, while hidden Markov models are employed for named entity recognition;

- **Deep Learning techniques**: these models revolutionized the approach to NLP by learning more abstract representations of text and by working without using extracted features. Convolutional Neural Networks (CNNs) excel at identifying patterns in text for tasks like classification, while Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks handle sequential tasks, such as translation and text generation, by capturing long-term dependencies in sequences;

- **Transformer Models**: currently the state-of-the-art in NLPs, transformers use attention mechanisms to process text efficiently. Notable models include BERT, ideal for language comprehension and classification tasks; GPT, known for text generation and next-word prediction based on the context; and T5 (Text-to-Text Transfer Transformer), which unifies various NLP tasks by treating them all as text generation problems.

2.1.3 State-of-the-Art NLP Models

As stated by Sawicki et al. (2023) [4], «Nowadays, NLP is one of the most popular areas of artificial intelligence. Therefore, every day, new research contributions are posted and it is rather difficult to capture the current "state of the field"». However, in this paper, the authors analyze the most common tasks explored in NLP research, collecting 4576 articles.

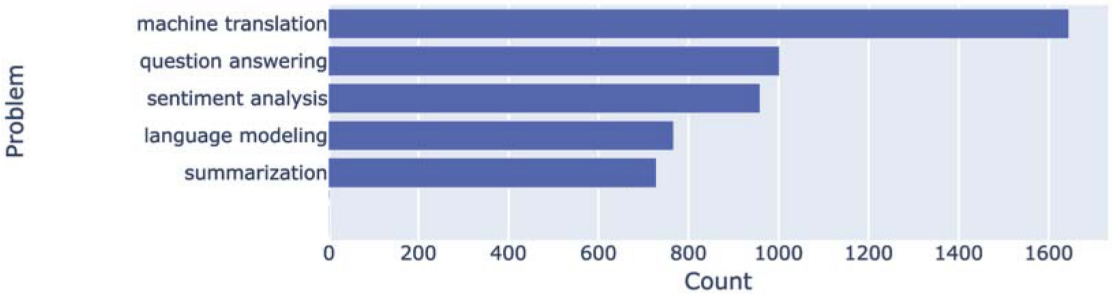


Figure 2.5: Published papers in the NLP field until 2021, divided by tasks.

As shown in Figure 2.5, the most popular tasks in NLP literature are Machine Translation (particularly from English to Chinese), Question Answering, Language Modeling, and Text Summarization.

Khurana et al. (2023) [5] also discuss the evolution of NLP, emphasizing the developments over the last 30 years. Figure 2.6 illustrates key milestones in NLP's development over recent decades. In 2001, the focus shifted toward Neural Language Modeling, where models predicted word occurrence by studying its probability based on prior words (n-grams). By 2008, Multitask Learning was introduced, leveraging convolutional models for tasks like speech generation and named entity recognition. In 2013, Word Embedding gained prominence, enabling models to capture word semantics, and Neural Networks were applied to NLP, marking a revolutionary shift. Following this, Sequence to Sequence models like CNNs, RNNs, and LSTMs became crucial, particularly in translation and text generation tasks. The breakthrough of the Attention Mechanism in 2015 paved the way for modern models, including Transformers and BERT, which remain foundational to today's NLP landscape.

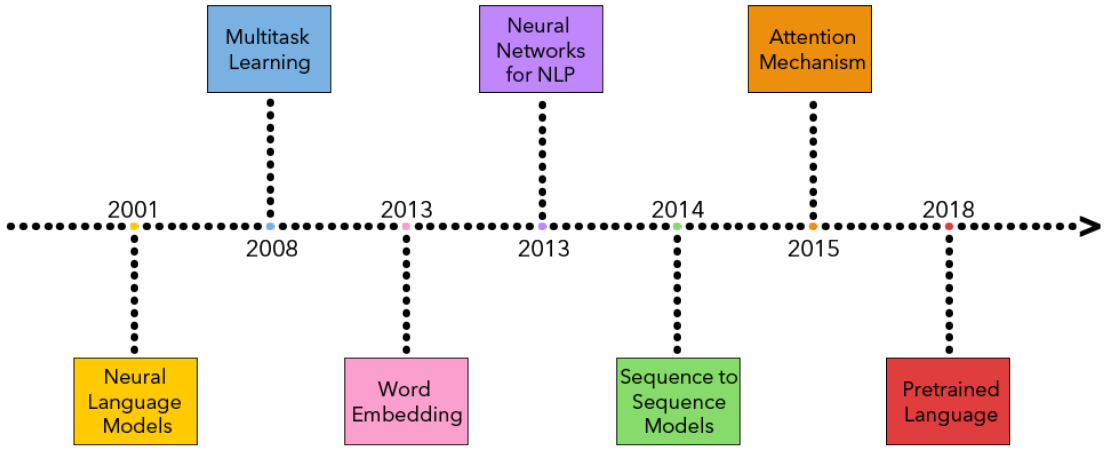


Figure 2.6: Latest key milestones in NLP's development.

Machine Learning models are generally categorized as:

- **Generative methods**: these generate new data samples by modeling the joint probability distribution; Examples include Naïve Bayes Classifiers and Hidden Markov Models (HMMs). They are more accurate for NLP tasks than Discriminative models;
- **Discriminative methods**: these models are based on observations, and directly map inputs to outputs by focusing on conditional probability, distinguishing between classes rather than generating data.

Naïve Bayes Classifiers

Naïve Bayes classifiers are based on probability theory and Bayes Theorem. They have been used in tasks like segmentation and translation, as well as in more specific applications such as identifying opinions and facts in documents or Question Answering systems. One limitation of these models is that, when encountering new words during test time that were absent in the training data, they may assign zero probabilities, which can negatively affect performance.

Hidden Markov Models

HMMs are another class of probabilistic models that represent systems evolving over time, where the true states (hidden states) are not directly observable. The

observable outputs are probabilistically dependent on these hidden states. In NLP, HMMs are commonly used for speech recognition and tasks like part-of-speech tagging and text segmentation.

Neural Networks in NLP

As previously mentioned, a major revolution in NLP techniques occurred in 2013 with the introduction of Neural Networks. These networks enhanced NLP tasks by enabling models to learn multi-level features. Word Embeddings allowed words to be represented as vectors, where distances between them reflected semantic similarity. Neural Networks brought advancements not only to word embeddings but also to tasks like information retrieval, text summarization, classification, machine translation, sentiment analysis, and speech recognition.

Initially, research focused on feedforward and CNNs, but RNNs and LSTMs later became critical for capturing word contexts in sentences and predicting words or sentence topics. Unlike HMMs, which predict hidden states, Neural Networks can anticipate future states.

The paper "Attention is All You Need" [6] by Vaswani et al. was a game changer, introducing the Attention Mechanism that led to the development of Transformer models, which are the core of NLP architectures. The major difference between Transformer models and their predecessors is their ability to process multiple sequences of text in parallel, thanks to the attention mechanism, rather than sequentially.

BERT

BERT (Bidirectional Encoder Representations from Transformers) is an architecture that leverages the Attention mechanism. It is a bidirectional model that reads context in both directions, considering words to the left and right of the target word. This enables BERT to achieve a more comprehensive understanding of context compared to unidirectional models, making it highly effective in language comprehension tasks.

BERT is pre-trained on a large corpus of data, mostly drawn from Wikipedia and books, using two unsupervised tasks:

- **Masked Language Model**: during training, 15% of the words in a sentence are masked, and the model must predict the hidden words based on their surrounding context. This helps BERT learn bidirectional word representations, as it considers both the preceding and following words to make accurate predictions;
- **Next Sentence Prediction**: the model receives two sentences and must predict whether the second sentence logically follows the first. This helps BERT

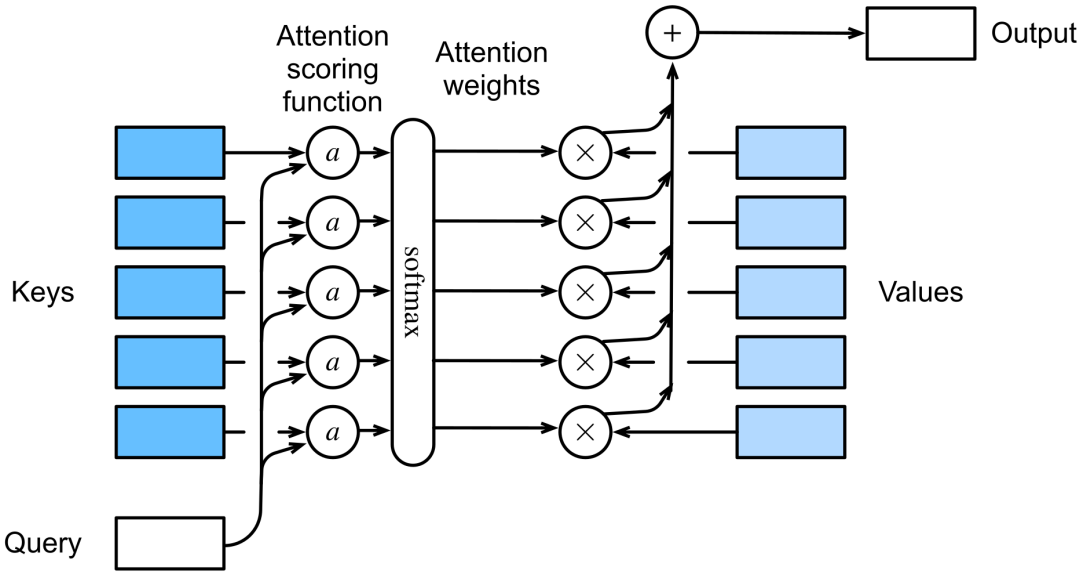


Figure 2.7: Overview of Attention mechanism.
Source: Wikipedia

better capture relationships between sentences, improving its performance in tasks such as question answering and text inference.

After pre-training, BERT can be fine-tuned for specific tasks using relevant datasets. This flexibility is one of BERT's major advantages, as it can be adapted to a wide variety of NLP tasks by simply modifying the final classification layer of the network, known as the "classification head".

BERT comes in two forms: the Base model, with 12 Transformer layers and 110 million parameters, and the more powerful Large model, with 24 Transformer layers and 340 million parameters. Several variants of BERT have been developed over time, the most notable being DistilBERT, RoBERTa, and ALBERT.

BERT excels in a wide range of NLP tasks, such as Text Classification, Named Entity Recognition, Question Answering, Paraphrase Detection, and Linguistic Acceptability. However, it has some limitations, the most significant being the considerable computational resources required for both training and inference. Another limitation is that the artificial masking used during pre-training differs from natural language comprehension, where all words are visible.

GPT

Another family of NLP models is GPT (Generative Pre-trained Transformer), developed by OpenAI. Like BERT, it is based on the Transformer architecture,

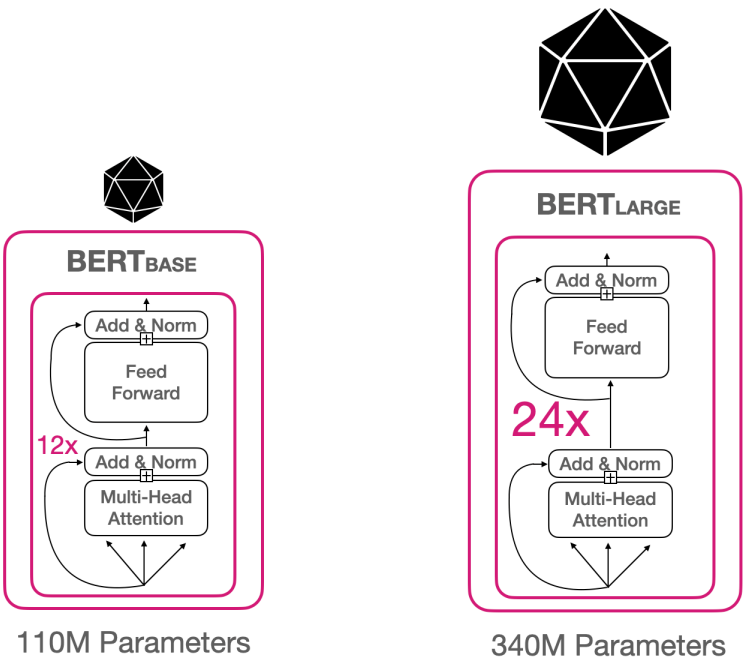


Figure 2.8: Comparison between BERT Base and Large sizes and architectures.[1]

but it handles text differently. GPT specializes in Automatic Text Generation, producing high-quality, contextually coherent text, which has led to its extraordinary success across various NLP applications.

Unlike BERT, GPT employs a unidirectional model, processing text sequentially. It generates the next word based only on the preceding words, making it ideal for text-generation tasks. GPT uses causal attention, where each word in the input sequence attends only to preceding words, ensuring a natural and coherent flow in the generated text.

GPT is pre-trained on vast unsupervised text data (books, Wikipedia articles, web sources, etc.), focusing on sequential text generation. The model is trained using the next word prediction technique: given a context, GPT predicts the next word in the sequence. One key to GPT's success is its pre-training dataset, which consists of generic data, enabling it to capture both language and domain information. After pre-training, GPT can be fine-tuned for specific NLP tasks such as Question Answering, Automatic Translation, or Creative Writing, using task-specific datasets.

Over the years, GPT has undergone several improvements, resulting in multiple versions: GPT-1 (2018), GPT-2(2029), GPT-3(2020), and GPT-4(2023). Each

[1]huggingface.co/blog/bert-101

new version has achieved better performance and higher-quality results than its predecessor.

GPT can tackle a wide range of NLP tasks, excelling at zero-shot and few-shot learning, allowing it to address new problems with minimal or no training data. Its output remains coherent and contextually relevant, even for complex topics, especially with GPT-3 and GPT-4. However, GPT can occasionally generate hallucinations and text that seems plausible but contains false or inaccurate information. Additionally, it lacks a real comprehension of the world, and it requires significant computational resources to train and deploy.

More details about GPT models will be discussed in the next chapters, as these models are central to this thesis and have been used for Text Summarization.

2.2 Generative Artificial Intelligence

Generative Artificial Intelligence (GenAI) is undoubtedly one of the most transformative topics in technology today. «The field of artificial intelligence arose from the idea that machines might be able to think like humans do. It required an analysis of how our brains process information and use it to perform new tasks and adapt to novel situations.»[2] AI has found applications across every existing field, making it a foundational technology.

GenAI refers to AI systems that can generate various types of content - text, images, music, programming code, etc. - in response to prompts. Unlike traditional AI, which focuses on analyzing data to detect patterns, GenAI's primary objective is to learn patterns from vast datasets to produce creative and effective outputs. GenAI has revolutionized areas like text, image, and video generation, and it is also used in many tasks requiring creativity. Its backbone lies in Deep Learning Architectures, which predict and generate contextually coherent content.

2.2.1 Generative networks through the years

In the late 2000s, neural networks were primarily trained as discriminative models. However, in 2014, new deep neural networks capable of generating complex data, such as images, emerged. The two most significant models from this period were Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs).

VAEs have an encoder-decoder structure but differ from traditional autoencoders because they introduce a probabilistic layer, allowing for the generation of new,

[2]scienceexchange.caltech.edu/topics/artificial-intelligence-research/artificial-intelligence-definition

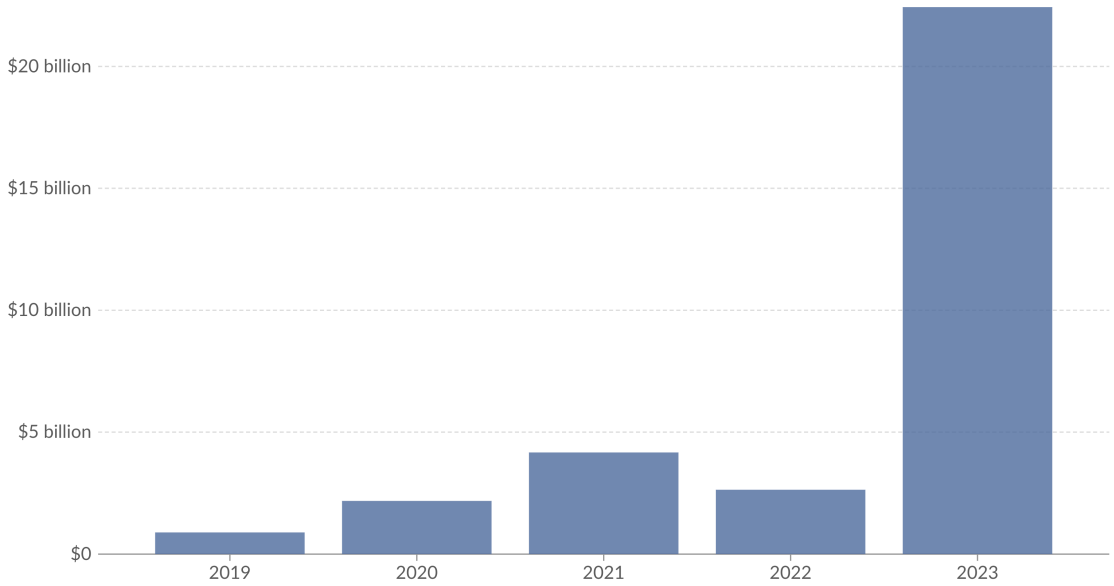


Figure 2.9: Global investment in Generative Artificial Intelligence in the years 2019-2023.[3]
Data Source: Quid via AI Index, U.S. Bureau of Labor Statistics (2024)

unseen data by sampling from a distribution in the latent space. These models became popular for image generation.

GANs also generate images but use adversarial training to improve the quality of the output. They consist of two neural networks, where one - the generator - creates pictures and the other - the discriminator - evaluates their realism. Since neural networks are very good at finding tricks to improve the loss function without underperforming in the task, the discriminator is used as a loss function, trained to recognize these tricks and force the generator to create increasingly realistic images. Autoencoders are made of an encoder block followed by a decoder block, while GANs reverse the order of the encoder and decoder blocks.

In 2017, the Transformer model brought groundbreaking advancements to the generative field, surpassing older LSTM models. After the publication of the paper "Attention is All You Need"[6], OpenAI introduced the first Generative Pre-trained Transformer (GPT) model in 2018. GPT-2, released shortly after, demonstrated that these models could generalize across a variety of tasks.

By 2020, the golden age of GenAI began, with the creation of models like DALL-E, Midjourney, and Stable Diffusion, transformer-based models that can generate good-quality images from prompts. In particular, GPT gained widespread attention

[3] ourworldindata.org/artificial-intelligence

in 2022 with the public release of ChatGPT, which enabled general-purpose GenAI tasks. Since 2019, the global investment in GenAI has been continuously growing, as shown in Figure 2.9.

2.2.2 Top performing models

Today, several generative models excel in different domains, with GPT, DALL-E, and Stable Diffusion leading the field.

- **GPT**: developed by OpenAI in 2018, GPT is a Transformer-based model specializing in text generation. With the release of GPT-3 in 2020, the model became a cornerstone in NLP, thanks to its 175 billion parameters and enabling applications such as creative writing, chatbots, and virtual assistants;
- **DALL-E**: also developed by OpenAI in 2021, DALL-E is a neural network that generates images from text descriptions. It combines language and computer vision, unlocking the potential of multimodal models for creative applications;
- **Stable Diffusion**: created by Stability AI in 2022, Stable Diffusion is a latent diffusion model that generates high-quality images. It is computationally efficient and can be deployed on less powerful hardware, making GenAI more accessible for both artistic and commercial purposes.

2.2.3 Advantages and Limitations

Transformer-based models like GPT-4 and BERT leverage self-attention mechanisms to achieve high-quality results in text generation and language understanding, revolutionizing NLP. Similarly, multimodal models like DALL-E and Stable Diffusion reduce the gap between text and vision by generating high-quality, realistic, and artistic images from natural language prompts.

One key advantage of GenAI models is their ability to perform zero-shot and few-shot learning, enabling them to fulfill tasks with few or no examples during training.

They are widely used in all creative fields, such as content creation, writing, music composition, video generation, and even code generation, and are innovating a lot of sectors, such as biotechnology, education, and marketing. These models can support creatives and designers in their work, and also allow users without any particular experience or competences in a certain field, to produce a good quality result without too much effort.

However, despite their power, GenAI models face various challenges. One major issue is the tendency to produce hallucinations, where the generated content, while

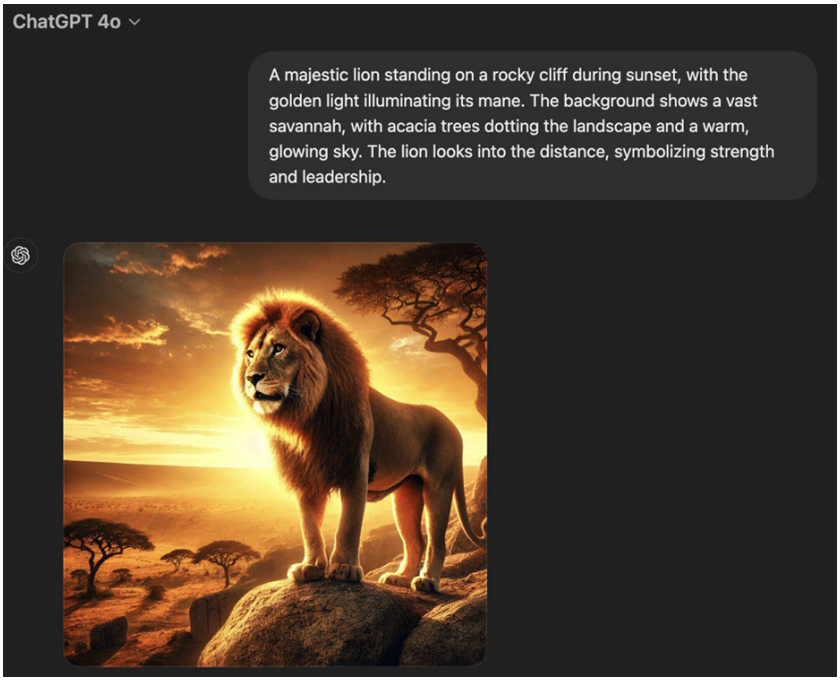


Figure 2.10: An example of DALL-E image generation, integrated into ChatGPT-4o.

plausible, contains inaccurate information not based on real data. This arises from the models' reliance on patterns learned from vast datasets without real-world comprehension. Additionally, bias in training data can lead to ethical and social biases in outputs. GenAI models also require extensive computational resources, consuming significant energy during training and inference.

Ethical concerns have also emerged, particularly regarding the creation of deepfakes, which can spread disinformation. Copyright issues may also arise, as generating content similar to existing works raises legal matters and intellectual property questions.

2.2.4 Applications

GenAI is finding increasing applications in various fields beyond the commonly known text, image, and video generation tasks.

According to the paper "The Potential of Generative Artificial Intelligence Across Disciplines: Perspectives and Future Directions" [7] (Ooi et al., 2023), some of the most impactful areas include:

- **Marketing**: GenAI tools have revolutionized digital marketing by enabling new advertisement formats, localized offers, and personalized content creation

Figure 2.11: Reply AI Film Festival, an international competition for AI-generated videos and films.[4]

for social media posts. Virtual assistants and recommendation systems powered by **GenAI** can generate advertisements, develop tailored digital marketing strategies, and create custom chatbot-based solutions or blog posts. These capabilities help companies enhance customer experiences and increase revenue.

Personalization is becoming increasingly important in this field, with GenAI tools analyzing search history, likes, and other user behavior to deliver highly targeted content. Many firms are already using their GenAI systems, using training datasets made of company projects on top of open-source data, charging further costs for customers who want to access these premium systems. However, the integration of public and private data introduces challenges related to data ownership, intellectual property, and privacy. Moreover, search bias in GenAI models can damage a company's brand reputation if not properly managed;

- **Health Care**: in the healthcare sector, GenAI can support both doctors and patients by improving diagnostic processes and patient care. For doctors, GenAI tools can analyze patient histories to help avoid prescriptions that may conflict with previous medical conditions. On the patient side, GenAI can provide quick, accurate answers to health-related queries without having to browse all the possible information on websites or documents and help translate medical documents into more understandable formats. However, the lack of transparency in AI-driven decision-making might lead to trust issues,

[4]reply.com/en/artificial-intelligence/reply-ai-film-festival

as patients may be unsure of how these tools generate their recommendations and may continue to prefer human advice. Biases and prejudices in training data can also result in incorrect medical advice or even illegal suggestions in some countries. The quality of the input data is more critical than in other tasks. If it is wrong because patients cannot provide a good prompt or there is not enough available data, this can lead to potentially harmful outcomes, which could be fatal to users;

- **Education**: GenAI has begun transforming the educational landscape, benefiting both students and educators. Students can utilize AI-powered tutors for personalized learning, access tailored educational materials, and receive assistance with essay writing and research. Educators can leverage GenAI for grading, creating assessment matrices, and generating lesson plans or support materials. This could help both novice teachers who still have to build their material, but also more experienced educators, who can improve what they already use. However, the usage of AI for educational purposes is raising various concerns. If misused or applied with malicious intent, AI tools can potentially obstacle to enhancing the learning experience. Consequently, while some institutions are actively encouraging students to engage with these new tools, others have opted to prohibit their use in schools and colleges.

  It has become increasingly challenging to determine if an essay is genuinely authored by a student or generated by an AI, raising significant concerns about plagiarism and academic integrity. Privacy and data security are also critical issues, as students' data could be accessed or shared without their consent, leading to potential breaches of confidentiality. Additionally, the psychological and communicative distance between learners and educators may increase, underscoring the continued importance of a human teacher in the learning process. Finally, many AI models used in educational contexts may not reflect the latest developments, potentially leading to outdated or inaccurate responses. This lack of current information can result in erroneous interpretations or explanations, especially if recent advancements in the subject matter are omitted;

- **Banking**: the banking sector is another prominent field for applying GenAI. Banks have a unique advantage over other sectors due to their vast datasets collected over the years, which serve as a valuable resource for AI and machine learning.

  A primary use case is the development of custom models to address customer inquiries on financial services and products. For example, Bloomberg developed BloombergGPT [5], a 50-billion parameter model trained on both proprietary and open-source financial data. This model demonstrates high

performance on financial benchmarks while retaining versatility as a general-purpose tool. GenAI-driven chatbots can offer personalized financial advice and recommendations to meet users' needs. Additionally, GenAI can be employed to tailor customer communication, such as personalized messages and advertisements, enhancing client engagement while optimizing costs. GenAI also has applications in fraud detection by recognizing patterns indicative of fraudulent behavior, allowing banks to identify suspicious transactions proactively.

However, several concerns accompany GenAI in banking, primarily surrounding data privacy, which must be safeguarded and strictly regulated. Public data sources may lack the necessary detail, and inaccuracies could have serious repercussions for clients. There is ambiguity regarding responsibility for errors, whether due to model limitations or user input, complicating liability. Banks are currently exploring best practices for deploying GenAI, including building customer trust in AI-driver services and understanding employee attitudes toward AI integration;

- **Human Resource**: Human Resource Management can also benefit significantly from GenAI systems, applied to various purposes such as enhancing the recruitment process, automating candidate information extraction from resumes, and assisting managers in identifying applicants well-suited to specific roles. Additionally, GenAI can help assess employee engagement by gathering data from social media and analyzing content through NLP and Social Media Analysis techniques. However, a lack of transparency in these systems could pose challenges, potentially diminishing employee trust in GenAI applications, colleagues, and even the organization itself. Furthermore, these models require substantial training data that accurately represents diverse employee groups; otherwise, they risk being ineffective or introducing bias that could impact fairness and inclusion.

2.3 Large Language Models

Large Language Models (LLMs) represent a sophisticated subset of Artificial Intelligence, excelling in complex tasks such as advanced language comprehension, prediction, and the generation of human-like text. LLMs are extensive deep learning models trained on large text corpora, often comprising vast datasets. Recently, LLMs have shown human-level proficiency in various NLP tasks, providing coherent, contextually accurate responses to user queries.

[5]bloomberg.com/company/press/bloomberggpt-50-billion-parameter-llm-tuned-finance

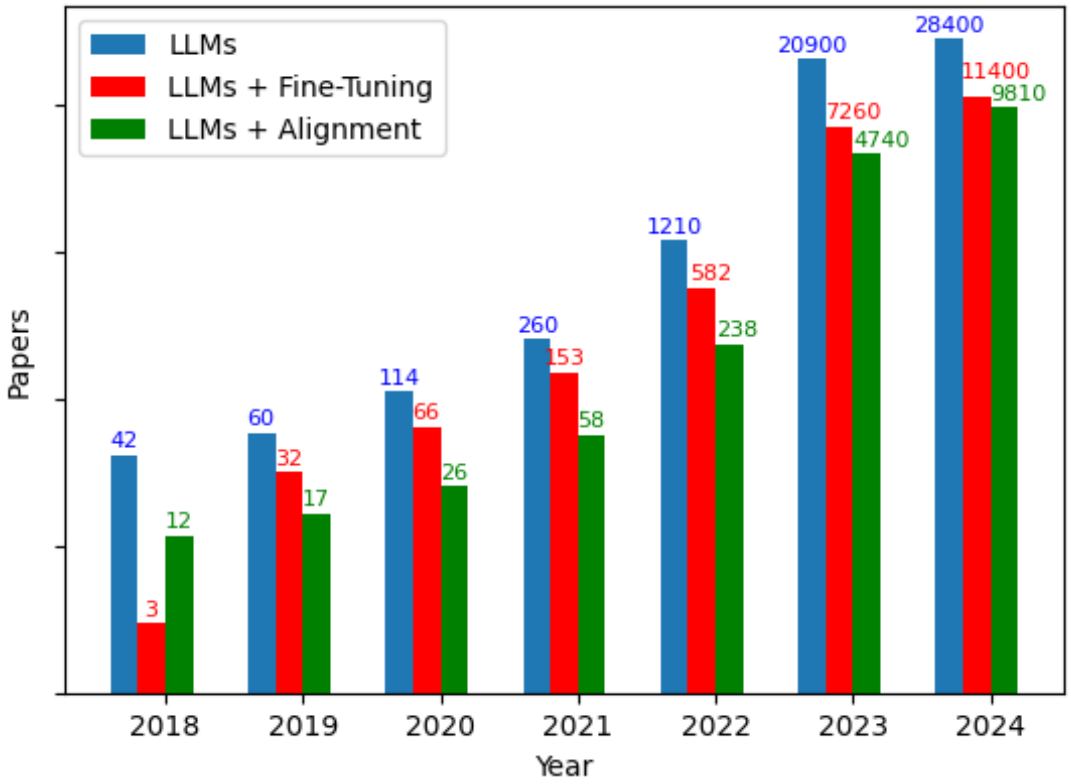


Figure 2.12: Number of papers published since 2018 containing the keywords "LLM", "LLM + Fine-Tuning", and "LLM + Alignment". Source: Navee ed Al. (2024) [8]

LLMs have rapidly become a central topic in contemporary scientific research. As illustrated in Figure 2.12, the number of publications including the keyword “LLM" has continued to rise consistently over the years, reflecting the growing interest and advancements in this field.

2.3.1 How LLMs Work

LLMs are constructed from multiple layers, including feedforward layers, embedding layers, and attention layers. The transformer architecture, particularly with self-attention mechanisms, represents a significant innovation that has enabled LLMs to achieve state-of-the-art performance in various language-related tasks.

Unlike early machine learning techniques that represented each word numerically in isolation, LLMs use multi-dimensional vectors, or word embeddings, to encode words. This approach places words with similar contextual meanings closer together in vector space. The transformer architecture processes through an encoder to

interpret word meaning, importance, and context and applies this understanding with a decoder to generate coherent, high-quality output. This enables the model to capture dependencies, weigh different parts of the input, and predict subsequent tokens effectively.

Transformer-based models are exceptionally large, containing vast numbers of nodes and layers. Each node connects to others and possesses weights and biases, also known as model parameters. These LLMs can enclose billions of parameters, and training involves vast amounts of diverse textual data. This training typically uses self-supervised learning, where the model iteratively adjusts parameters to accurately predict the next token in a given sequence.

While LLMs perform well on many tasks, they may struggle in zero-shot or few-shot settings without fine-tuning. To enhance generalization to unseen tasks, models are often fine-tuned with specific task instructions or user preferences. This fine-tuning helps tailor LLMs for specialized applications, optimizing them for particular use cases and enabling more natural interactions with users.

2.3.2 Architecture

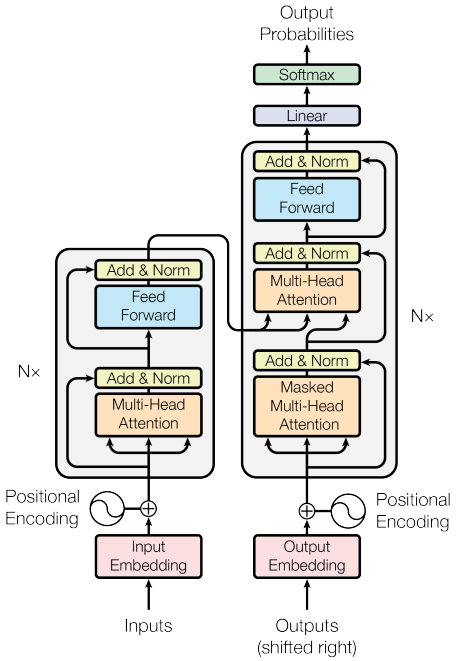


Figure 2.13: An example of Tranformer architecture.

LLMs can vary depending on factors such as target tasks, available computational resources, and specific objectives of the model. However, Transformer-based

architectures have become foundational, revolutionizing NLP tasks, with most LLMs following a general structure similar to that illustrated in Figure 2.13. The main components include:

- **Input Embeddings**: to capture the semantic and syntactic properties of the input, the text is first split into smaller units known as tokens. Each token is then mapped into a multi-dimensional vector, allowing the model to represent words in a form that preserves their contextual relationships;

- **Positional Encoding**: since Transformers do not inherently consider the order of tokens, positional encodings are added to the embeddings. This encoding provides a way for the model to retain the sequential structure of the text. ensuring that token order is factored into its understanding;

- **Encoder**: the encoder processes the input text to generate hidden states that maintain context and meaning throughout the sequence. Each encoder layer contains a Self-Attention mechanism followed by a Feed-forward Neural Network, creating a structure capable of capturing complex dependencies and relationships within the input text;

- **Self-Attention mechanism**: the Self-Attention mechanism is a pivotal component, allowing the model to weigh each token's relevance to others in the sequence. It calculates attention scores, enabling the model to consider dependencies and contextual relationships. This mechanism facilitates a nuanced understanding of each token's significance, taking into account both immediate and broader context;

- **Feed-Forward Neural Network**: after the Self-Attention layer, the Feed-Forward Neural Network is applied independently to each token. It consists of fully connected layers with non-linear activation functions, enabling the model to capture complex interactions, which adds depth to the representation of linguistic nuances;

- **Decoder Layers**: Decoders, when used alongside Encoders, support autoregressive generation, allowing the model to produce output sequences by attending to previously generated tokens. This feature is especially valuable in text generation tasks, where the sequence of generated tokens builds progressively;

- **Multi-Head Attention**: Multi-Head Attention is a technique where Self-Attention is applied in parallel with multiple sets of learned attention weights. This allows the model to analyze diverse semantic relationships within the input sequence simultaneously, attending to various aspects of the text;

- **Layer Normalization**: to further stabilize the training process, Layer Normalization is applied after each sub-component or layer. This enhances the model's generalization ability across diverse inputs and improves the convergence speed during training;
- **Output Layers**: the architecture's Output Layers vary based on the intended task. For language modeling, a common setup includes a linear projection followed by a SoftMax activation function, which converts the output into probabilities for each token in the vocabulary.

While this outline covers the core components, various models may introduce additional elements or modifications to this base structure to better address specific research goals or applications.

2.3.3 Top performing models

Several LLMs, with various versions, have been developed over the last years. Defining the state-of-the-art for these models depends on the specific tasks and use cases, as different models excel in different domains. However, three standout models in 2024 are GPT-4o, Claude 3, and Llama 3.

GPT-4o

Released by OpenAI in 2024, GPT-4o is a multimodal LLM capable of processing and generating text, audio, and images. Its name, 'o' for 'omni', highlights its advanced ability to integrate multiple data types. It demonstrates significant advancements over GPT-4, particularly in real-time speech recognition, audio translation, and visual processing capabilities. Further details and key features will be analyzed in the next chapter, as GPT-4o is the model used for the final product of this thesis.

Claude 3

Claude 3, developed by Anthropic, is recognized for its strong performance across a variety of complex tasks, including NLP, problem-solving, and code generation. Debuting in March 2024, it features improvements in reasoning, multilingual support, and complex task handling. A key differentiator of Claude 3 is its 200k context window, allowing it to handle large documents such as scientific papers with ease. This window can even be expanded to up to 1 million tokens for enterprise use, offering unprecedented processing capabilities. Thanks to this big context, Claude 3 improves factual accuracy, reduces hallucinations, and provides citations to verify its generated content.

To enhance its reliability, especially in ethically complex tasks, Claude 3 uses Constitutional AI training, which reduces the need for human feedback by adhering to predefined ethical guidelines. It is available in three versions - Haiku, Sonnet, and Opus - each optimized for different needs, from real-time interactions to large-scale data processing and research. In June 2024, Anthropic released Claude 3.5 Sonnet, further improving performance in specific areas. These features have positioned Claude 3 as a strong competitor to OpenAI's GPT-4o.

Llama 3

Llama 3, the latest iteration in Meta's autoregressive LLM, was released in 2024. It has positioned itself as a strong competitor to models like GPT-4o and Claude 3 due to its versatility and open accessibility. Llama3 supports multiple languages, making it particularly effective for translation tasks. It offers various configurations, ranging from smaller versions with 8 billion parameters to expansive models with up to 406 billion parameters. These variations allow for scalability across different applications, from lightweight to resource-intensive tasks.

Llama 3 also features an expanded context window, enabling it to process large user inputs and handle multi-step tasks with greater efficiency. A key advantage is the availability of many Llama 3 versions to developers for free, encouraging wider adoption and integration into diverse applications. Compact versions, such as Llama 3.2, are designed for use on mobile devices and edge computing, utilizing quantization techniques to ensure efficiency on low-power hardware without compromising performance. This flexibility makes Llama 3 highly effective for coding assistance, language generation, and multilingual conversation.

Each of these models - GPT-4o, Claude 3, and Llama 3 - has contributed significantly to the evolution of GenAI, offering tailored solutions for diverse tasks and paving the way for further advancements in the field. Figure 2.14 shows a comparison between them.

2.3.4 Applications

The versatility of LLMs enables their application across diverse industries, where they simplify processes and enhance productivity. Many business operations can be automated using these models, including content creation for marketing, technical documentation, customer service chatbots, and virtual assistants, sentiment analysis, code generation, debugging, and various other text-based operations such as translation, summarization, and text generation.

In industrial settings, LLMs support more specific processes, such as optimizing supply chain management through predictive analytics, inventory control, and aiding

Feature	Llama 3.1	GPT-4o	Claude 3.5
Developer	Meta	OpenAI	Anthropic
Model Type	Decoder-only transformer	Transformer-based	Transformer-based
Primary Use Cases	Synthetic data generation, model distillation, multilingual agents	Creative writing, technical documentation, problem-solving	Context-sensitive customer support, multi-step workflows, visual reasoning
Training Data	Over 15 trillion tokens	Diverse datasets	Proprietary datasets
Access and Integration	Meta.ai, Hugging Face, partner platforms	Various tools and APIs	Claude.ai, Anthropic API, Amazon Bedrock, Google Cloud's Vertex AI
Ecosystem Partnerships	AWS, NVIDIA, Databricks, Groq, Dell, Azure, Google Cloud, Snowflake	Various integrations and partnerships	Anthropic API, Amazon Bedrock, Google Cloud's Vertex AI
Deployment Flexibility	On-prem, cloud, local deployment	Cloud, APIs	Cloud-based services, APIs

Figure 2.14: Comparison between Llama 3.1, GPT-4o, and Claude 3.5, source: marktechpost.com[6]

technicians in troubleshooting complex issues. In industry-specific environments, LLM models can be integrated directly into the production applications after cloud-based training on company-specific data. This allows LLMs to produce responses and forecasts uniquely tailored to the operational requirements of the environment in which they are deployed.

In summary, LLMs demonstrate strong performance across a variety of NLP tasks, including Natural Language Understanding, Content Generation, Language Translation, Text Summarization, and Sentiment Analysis, making them invaluable assets across both commercial and industrial domains.

[6]marktechpost.com/2024/07/27/llama-3-1-vs-gpt-4o-vs-claude-3-5-a-comprehensive-comparison-of-leading-ai-models

2.3.5 Advantages and Challenges

LLMs offer significant benefits, but they also present notable challenges.

One of their primary advantages is efficiency in processing vast amounts of data. LLMs excel in tasks requiring complex natural language understanding, such as language translation and document summarization. They can be fine-tuned to adapt continuously to specific domains or datasets, making them suitable for specialized applications. Their adaptability also extends to zero-shot learning, where they perform well in new scenarios without additional training. This flexibility enables automation, significantly lowering costs in areas like customer support and data analytics. Additionally, by providing personalized, real-time assistance, LLMs improve customer experience and allow human resources to focus on more complex activities.

However, these capabilities come with challenges. Training and inference are slow, require substantial computational resources, and incur high operational costs. Training a large-scale model can cost millions of dollars and leave a significant carbon footprint, comparable to the lifetime emissions of several vehicles, posing environmental concerns. Implementing, fine-tuning, and maintaining LLMs also require highly skilled personnel. Given the vast amounts of textual data processed, questions arise around data scraping legality and the protection of sensitive information, especially in datasets with limited transparency regarding sources. Intrinsic biases in training data may be reflected in the model's outputs, perpetuating stereotypes or inaccuracies. Additionally, like other GenAI models, LLMs are prone to hallucinations, producing responses that may be inaccurate or incoherent. Finally, an over-reliance on these models can make businesses vulnerable, as any failure or inaccuracy in the model's output may disrupt operations or lead to invalid decision-making.

In summary, while LLMs offer transformative potential, businesses must weigh these benefits against the resource demands, ethical considerations, and operational dependencies inherent to their use.

2.4 OpenAI GPT

In recent years, GPT has emerged as one of the most transformative developments in the Artificial Intelligence field, capturing global attention and redesigning the landscape of machine learning applications. OpenAI is a pioneering American company specializing in artificial intelligence research and deployment that has rapidly gained prominence as the leading organization in this field. OpenAI's mission, «to ensure that artificial general intelligence benefits all of humanity»[7] reflects its commitment to the ethical development of advanced AI systems. The

development of GPT, OpenAI's flagship model, embodies this mission.

GPT's architecture leverages the Transformer model, which utilizes self-attention mechanisms to process language with unprecedented accuracy and fluency. This approach revolutionized NLP by enabling models to consider contextual information from entire text sequences, allowing for more coherent and contextually accurate responses.

In November 2022, OpenAI released ChatGPT, a conversational model based on GPT, which gained massive popularity, reaching one million users within its first week. ChatGPT marked a pivotal moment in AI, introducing a vast audience to the potential of GenAI tools and helping the widespread adoption of AI chatbots in everyday tasks. This model demonstrated how conversational AI could be integrated into both personal and professional applications, further accelerating AI's role in society.

2.4.1 History and evolutions

Before GPT, OpenAI was originally founded in 2015 as a research-focused nonprofit by visionaries like Sam Altman and Elon Musk, and aimed to promote safe and beneficial AI technologies. With the release of the research platform for reinforcement learning OpenAI Gym in 2016[9], the company introduced the concept of generative models, beginning its journey into AI innovation. By June 2018, OpenAI achieved outstanding results in language tasks, revealing GPT and signaling a new era in language models.

GPT-1

In 2018, OpenAI released the first version of the LLM model, GPT-1, which laid the foundation for large-scale language modeling. This model utilized a unidirectional approach, processing text from left to right, considering only the preceding tokens. GPT-1 included 12 layers, 768 hidden units, and 12 attention heads, totaling 117 million parameters. GPT-1's major contribution was its pioneering use of pre-training on a large generic text corpus, followed by fine-tuning for specific tasks, adapting to various use cases. Despite having only 117 million parameters, GPT-1 demonstrated that pre-training followed by task-specific fine-tuning could achieve state-of-the-art results across various NLP applications.

[7] openai.com

GPT-2

GPT-2, launched in 2019, significantly advanced OpenAI's language modeling capabilities, including 48 transformer blocks, 1600 hidden units, and up to 1.5 billion parameters. It showed that increasing the model size and training data could drastically improve the performance. GPT-2's ability to generate coherent and creative texts led to its impressive zero-shot and few-shot learning capacities, which allowed it to tackle new tasks for which it was not specifically trained during fine-tuning, but based on prompts alone.

Since its generated text reaches high-quality results, GPT-2's potential for misuse, particularly in generating fake content, raised ethical concerns, pushing OpenAI to delay the release of its most powerful versions to address security implications.

GPT-3

Released in 2020, GPT-3 represented the turning point for the LLM models. It is a big leap, with 175 billion parameters, bringing AI closer to human-like understanding.

This version is particularly known for its capability of generating incredibly natural and coherent texts and excelled in a wide range of language tasks, from translation to code generation. It has also shown increased abilities in zero-shot and few-shot settings, thanks to its immense parameter scale, covering numerous topics and domains. With GPT-3, prompt engineering emerged as a vital skill, as users could easily leverage complex prompts to instruct the model effectively.

Since GPT-3's behavior may resemble human understanding and reasoning, with similar performances, this also brought a widespread discussion on AI's ethical boundaries, leading to considerations for AI regulation.

In 2022, the enhanced GPT-3.5, which improved stability and performance, was released, overcoming some limitations of the previous versions. It is extremely important because it became the backbone of ChatGPT, the notorious chatbot that made GPT technology widely accessible.

GPT-4

GPT-4, introduced in 2023, further improved accuracy and adaptability. It remains the most advanced version of GPT, providing more coherent responses and being more accurate and versatile for a wide range of activities, though OpenAI has not disclosed its training parameters count. Key innovations include enhanced multimodal capabilities, allowing it to process text and image data and interpret also visual contents. OpenAI improved training of the model to reduce hallucinations,

resulting in more reliable responses, and it can be better customized for specific domains.

GPT-4 is now, together with some optimized sub-versions, foundational for ChatGPT, which still is the most used conversational agent.

GPT-4o and Optimized Models

GPT-4o is an optimized version of GPT-4 specifically designed to be lighter and less resource-intensive, without compromising performance. This model is particularly suited for tasks like multi-document summarization, where analyzing and synthesizing information from multiple sources is critical, making it a valuable addition to the project presented in this thesis.

OpenAI continues to develop specialized versions, with specific optimization techniques to increase the accessibility of these models across platforms, maintaining high accuracy and versatility with reduced computational demands.

2.4.2 Model Architecture

The GPT model architecture is fundamentally based on the Transformer model, which enables deep language comprehension and generation through self-attention mechanisms. This design allows GPT to capture intricate relationships between words in a text sequence, a crucial aspect for producing coherent and contextually accurate text.

While traditional Transformers consist of both an Encoder and a Decoder, GPT utilizes only the Decoder, optimized specifically for unidirectional text generation. In this unidirectional approach, GPT processes words sequentially from left to right, using only preceding tokens to predict the next word. This enhances the coherence of generated text and aligns perfectly with text generation tasks.

GPT's self-attention mechanism employs a special technique known as Causal Masked Self-Attention. With this approach, there is a causal mask in the attention matrix, so each token can not access words that appear subsequently in the input sequence, thus preserving causality in the text generation, ensuring that each predicted word is only based on the preceding context. To maintain the order of the words in the text, GPT also incorporates positional encoding, assigning a positional representation to each token. This encoding helps the model recognize the structure of sentences, enhancing coherence and readability.

One of the main characteristics of GPT's evolution is its increasing scale, marked by significant growth in the number of parameters. GPT-1 began with 117 million parameters, GPT-2 expanded to 1.5 billion, GPT-3 reached an impressive 175 billion, and GPT-4, though undisclosed, features even more parameters than its predecessors. This growth has enabled GPT to capture finer language nuances,

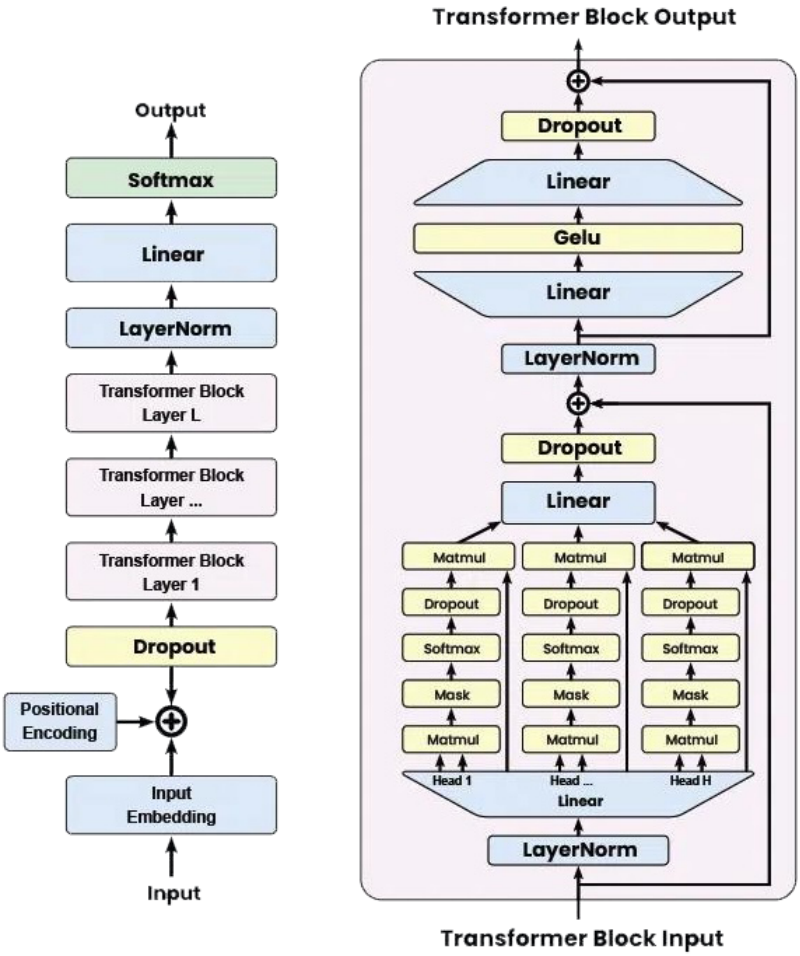


Figure 2.15: GPT Transformer Architecture, source: geekforgeeks.org[8]

producing highly accurate and contextually relevant text.

GPT's training involves two main phases: pre-training and fine-tuning. During pre-training, the model is exposed to large, unlabeled datasets, learning language patterns by predicting the next word based on the preceding context. To be adapted for specific applications, GPT is fine-tuned on targeted datasets, and recent versions incorporate Reinforcement Learning from Human Feedback (RLHF). This approach utilizes human feedback to refine the model's responses, improving the relevance and quality of its outputs.

GPT-4o and subsequent specialized models introduce new optimizations aimed at reducing computational requirements while preserving accuracy. Key techniques

[8]geeksforgeeks.org/introduction-to-generative-pre-trained-transformer-gpt

include Model Distillation, which compresses the model without significant loss of performance, Quantization, which reduces parameter precision to save memory, and Pruning, which eliminates redundant connections. These optimizations allow GPT-4o to function effectively on devices with limited resources, making advanced generative capabilities more accessible.

2.4.3 Distinctive Functionalities

GPT stands apart from other LLM models due to several unique functionalities, which contribute to its impressive performance in natural language generation and its suitability for a variety of complex applications.

A fundamental difference in GPT's architecture is its reliance on a unidirectional Causal Attention approach. GPT respects natural reading and writing orders by focusing only on preceding words, making it ideal for generating long, coherent text. This contrasts with bidirectional models like BERT, which incorporate both left and right contexts for each token, making GPT particularly well-suited for generating fluid narratives. Another unique architectural feature is its use of RLHF to refine responses. This technique aligns the model's outputs more closely with human expectations, reducing the likelihood of generating 'hallucinated' or inaccurate information. GPT architecture is also highly scalable, with each version showing off an increased parameter count. This expansion allows the model to handle complex contexts and produce articulate, nuanced responses, enhancing its ability to execute sophisticated tasks with greater precision compared to smaller models.

Additionally, thanks to extensive training on diverse textual datasets, GPT excels in zero-shot and few-shot learning, being able to address new tasks without requiring extensive examples. This adaptability, combined with fine-tuning capabilities, allows GPT to be effectively customized for a wide range of applications.

GPT's architecture also makes it exceptionally well-suited for multi-turn conversations. Its ability to handle long sequences allows it to remember details from earlier exchanges, ensuring continuity and coherence throughout dialogues. This feature enhances user interaction by creating a more natural conversational flow, a distinct advantage over other LLMs that are less adapted for ongoing conversations.

OpenAI has made GPT widely accessible via APIs, allowing easy integration into commercial applications and services. This accessibility has accelerated GPT's adoption across diverse industries, enabling companies to leverage GPT's capabilities in real-world scenarios.

Recent research highlights GPT's potential across specialized fields. For instance, Katz et al. (2024) [10] demonstrated that GPT-4, even in zero-shot scenarios, could pass the Uniform Bar Examination (UBE), a test for lawyers' knowledge and skills, surpassing human performance in five of seven subject areas. Similarly, Abdelghani et al. (2022) [11] showed that GPT-3 could effectively promote question-asking skills

in children, supporting educational settings with a creative and curiosity-driven approach.

2.5 Multi-Document Summarization

In an era where information overload is a constant challenge, Text Summarization plays a pivotal role in automatically distilling key insights from extensive documents. This task is common in everyday life, intending to condense large textual content into brief, meaningful summaries that retain essential information, context, and intent.

While traditional Text Summarization focuses on a single document, Multi-Document Summarization (MDS) expands this task by condensing information from multiple sources. MDS is particularly valuable when users must analyze large clusters of documents related to the same topic, helping them quickly identify core insights without redundancy. Automatically generated summaries from large document clusters save time and effort, especially when working with a huge amount of documents that require filtering through numerous sources for meaningful information.

MDS has emerged as a foundational NLP task, and over the years, a spectrum of approaches has been developed to tackle it, ranging from traditional methods to advanced AI-driven models that involve the latest advancements in deep learning and language modeling.

2.5.1 Traditional Approaches

Traditional approaches to MDS laid the foundation for modern techniques by focusing on generating sentences directly from the original texts. These methods prioritize simplicity and reliability, standing on well-established linguistic and statistical principles. Two primary strategies dominate traditional summarization: Extractive and Abstractive Summarization, each with distinct methodologies and applications, differing, above all, on the way sentences are produced.

Extractive Summarization

Extractive Summarization systems identify and extract sentences or snippets of text directly from the original documents to include them in the final summary. This method ensures that the summary retains the original phrasing and factual accuracy, minimizing semantic distortion. However, since it relies on existing text segments, the resulting summaries can sometimes appear disjointed or fragmented, especially when the extracted content lacks context.

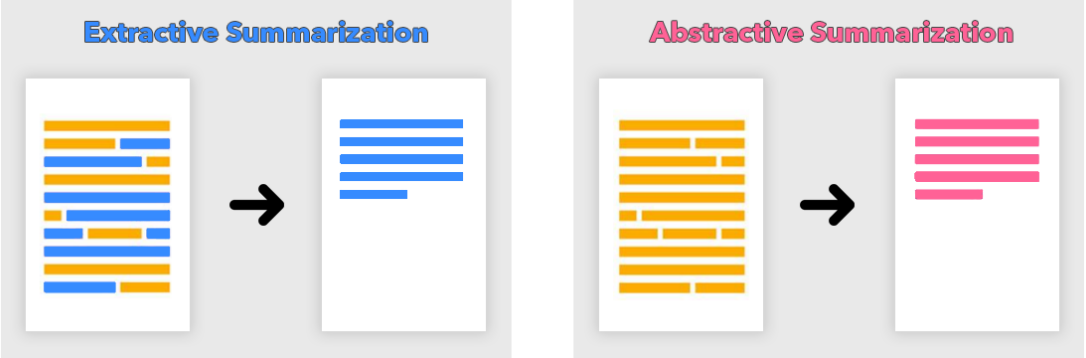


Figure 2.16: Extractive vs Abstractive Summarization representations.

This approach is computationally efficient and easier to implement compared to Abstractive Summarization, making it well-suited for applications like summarizing multiple news articles or documents focusing on a common topic or event. Common algorithms used for Extractive Summarization include TextRank, Latent Semantic Analysis (LSA), and Maximum Marginal Relevance (MMR).

Abstractive Summarization

Abstractive Summarization, on the other hand, generates new sentences that form the final summary, rather than copying sections verbatim from the source texts. Inspired by human summarization processes, this approach requires the system to deeply understand and reinterpret the content of the input texts, generating new expressions that concisely capture the core ideas and meaning. While this technique results in summaries that are more fluid, natural, and semantically rich, it also has to face computational and linguistic challenges. For example, ensuring that the generated summary is factually accurate and contextually relevant can be difficult.

Abstractive systems require advanced models capable of semantic understanding, which often translates into higher computational costs. Despite these challenges, Abstractive Summarization is increasingly favored for applications requiring high-quality, human-like summaries, such as legal document summarization or research paper synthesis.

2.5.2 Advanced Approaches

While traditional approaches like Extractive and Abstractive Summarization have been instrumental in advancing the field, they also come with limitations, such as fragmented summaries in extractive methods and high computational costs

in abstractive ones. These challenges have encouraged the development of more sophisticated techniques, leveraging modern advancements in deep learning and neural networks.

With the rise of Abstractive Summarization methods, which produce summaries with greater conciseness and readability, advanced techniques that use it as a basis have become the cornerstone of state-of-the-art MDS. However, recent advancements in AI and LLMs have introduced novel methodologies for MDS, leveraging neural architectures such as Sequence-to-Sequence and Transformer models. These innovations enable a better understanding of context and dependencies among sentences, resulting in accurate, fluid, and coherent summaries.

Pre-trained models, such as BERT and GPT, have significantly influenced recent advancements in MDS by providing a robust foundation for both Abstractive and Generative Summarization. As previously pointed out, these models offer pre-learned linguistic representations that can be fine-tuned for specific summarization tasks, enhancing performance across diverse datasets and domains.

BERTSUM

Introduced by Liu in 2019, BERT for Summarization (BERTSUM) [12] is an extension of the pre-trained Transformer model BERT, designed specifically for extractive summarization. The model adapts BERT to the summarization task by implementing a specialized sentence representation, including additional tokens at the beginning and end of each sentence, alternating segment embeddings to denote sentence positions, and new summarization-specific layers for handling longer sequences across sentences.

As shown in Figure 2.17, these modifications allow BERTSUM to excel at summarizing news-related documents, demonstrating its efficacy in extracting coherent and contextually relevant summaries from multiple sources.

PRIMERA

Proposed by Xiao et al. in 2022, PRIMERA [13] is a pre-trained generation model optimized for multi-document summarization tasks. The model employs an innovative pre-training strategy called Entity Pyramid Masking, which trains the system to identify and aggregate salient information across multiple documents. By integrating the Longformer-Encoder-Decoder (LED) model, PRIMERA effectively handles the long input sequences typical of multi-document tasks. The use of sparse attention mechanisms enables efficient processing of concatenated documents, making PRIMERA particularly well-suited for domains with complex and lengthy inputs. The model has demonstrated state-of-the-art performance in zero-shot, few-shot, and fully supervised settings, surpassing previous benchmarks across various domains.

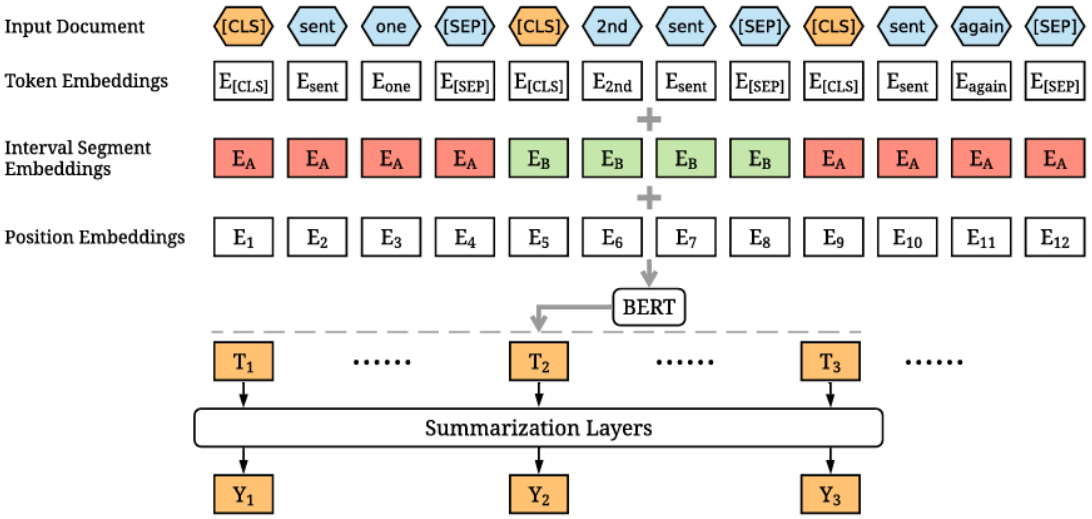


Figure 2.17: Overview of the BERTSUM architecture. Source: Fine-tune BERT for Extractive Summarization[12]

These advanced approaches represent a significant leap forward in the capabilities of Multi-Document Summarization. By leveraging innovative architectures and pre-training strategies, they have addressed many of the limitations of traditional methods, paving the way for further advancements in the field.

2.5.3 Datasets and Evaluation Metrics

Datasets have a central role in advancing MDS research, as they provide structured and diverse data to train and benchmark models. Some datasets have become widely adopted due to their unique characteristics, making them particularly suited for specific experiments and applications.

CNN/Daily Mail

The CNN/Daily Mail dataset, created in 2015, is one of the most widely used resources for text summarization tasks. Originally designed for reading comprehension and question answering, it has since been adapted for both single-document and multi-document summarization applications. This dataset comprises over 300000 articles from CNN and the Daily Mail. Each article is paired with a series of bullet points that can be interpreted as a resume, even though they were originally intended to represent the article content. The key points report the most essential information of each article. These articles focus on news, politics, world events, and other public themes, ranging from various textual content.

The dataset is suitable for training both extractive and abstraction methods: models like BERTSUM, T5, and PEGASUS have been tested on this dataset to evaluate their summarization capabilities. While it is versatile and large, it also presents limitations. These include redundancy in key sentences and its narrow focus on news content, which may limit its applicability to domains beyond current events and public affairs.

Multi-News

Introduced by Fabbri et al. in 2019, the Multi-News dataset [14] is a large-scale resource specifically designed for multi-document summarization in the news domain. It contains 56216 news articles curated from Newser.com, along with professionally written summaries by editors.

Figure 2.18 shows an example of three different source texts and their summary in the Multi-News dataset.

Unlike earlier datasets, Multi-News stands out for its diversity, offering clusters of documents presenting different perspectives on the same topics. This makes it an ideal benchmark for evaluating MDS models that aim to aggregate and reconcile conflicting information. Despite its strengths, its news-centric scope may limit its utility for domains requiring more heterogeneous inputs.

WikiSum

Proposed by Google AI in 2018, WikiSum was designed for abstractive summarization tasks. It synthesizes information from diverse sources to produce concise summaries analogous to Wikipedia articles. It is built from Wikipedia, each data point consists of a topic (a Wikipedia article title), external documents related to the topic, and a summary (the Wikipedia article text).

What makes WikiSum unique is its reliance on web scraping to gather source documents, which include news articles, blogs, Wikipedia pages, reports, and other authoritative texts. This diversity enables WikiSum to train models capable of handling varied linguistic styles and domains but also to be a benchmark for already existing models. However, challenges include variability in source quality, the computational cost of processing large datasets, and the potential obsolescence of scraped content over time.

Multi-Document Summarization of Medical Studies

Created by DeYoung et al. in 2021, Multi-Document Summarization of Medical Studies (MS^2) [15] is tailored for Multi-Document Summarization in the biomedic domain. With 470000 documents and 20000 resumes, it addresses the need for synthesizing information from medical studies to support healthcare professionals

Source 1
Meng Wanzhou, Huawei's chief financial officer and deputy chair, was arrested in Vancouver on 1 December. Details of the arrest have not been released...
Source 2
A Chinese foreign ministry spokesman said on Thursday that Beijing had separately called on the US and Canada to "clarify the reasons for the detention "immediately and "immediately release the detained person ". The spokesman...
Source 3
Canadian officials have arrested Meng Wanzhou, the chief financial officer and deputy chair of the board for the Chinese tech giant Huawei,...Meng was arrested in Vancouver on Saturday and is being sought for extradition by the United States. A bail hearing has been set for Friday...
Summary
...Canadian authorities say she was being sought for extradition to the US, where the company is being investigated for possible violation of sanctions against Iran. Canada's justice department said Meng was arrested in Vancouver on Dec. 1... China's embassy in Ottawa released a statement.. "The Chinese side has lodged stern representations with the US and Canadian side, and urged them to immediately correct the wrongdoing "and restore Meng's freedom, the statement said...

Figure 2.18: An example of a Multi-News Dataset Summary. Source: [14]

with their research, also being updated with the latest findings. The domain of documents is the real biomedical literature, creating a realistic bench. The dataset clusters documents around specific medical questions or topics, providing human-written summaries for each cluster, to use them as a comparison to evaluate the quality of the model's resumes. MS^2 is one of the biggest datasets in the biomedical field and it is particularly valuable for benchmarking MDS models in complex and specialized fields, although its reliance on domain-specific data may limit its generalizability to other areas.

Evaluating summaries generated by MDS models is inherently challenging. Effective evaluation metrics must balance precision, semantic coherence, and readability while ensuring that key information is retained and redundancy is minimized. Several metrics have been widely adopted, each with its strengths and limitations.

ROUGE

Recall-Oriented Understudy for Gisting Evaluation (ROUGE) is a family of evaluation metrics widely used in MDS tasks, particularly for extractive summarization. It measures the overlap between n-grams (sequences of n words) in the generated and reference summaries.

Common variants include:

- ROUGE-N: Evaluates n-grams overlap, where N is an integer that can represent unigrams, bigrams, or trigrams;
- ROUGE-L: Focuses on the longest common subsequence, assessing textual coherence.

While ROUGE is effective for extractive methods, its reliance on exact word matching limits its ability to assess semantic quality, making it less suitable for abstractive summarization, as it penalizes reformulations of the information.

BLEU

Bilingual Evaluation Understudy (BLEU), originally developed for machine translation, measures n-gram overlap between generated and reference summaries. It evaluates the quality of the summaries by counting the number of n-grams present in both of the summaries. It emphasizes precision but struggles with evaluating abstractive summaries due to its rigidity. Although less commonly used compared to other metrics, BLEU remains useful for tasks where precision is extremely important.

BERTScore

BERTScore evaluates the semantic similarity between generated and reference summaries, by leveraging pre-trained models like BERT. Both the generated and the reference summary are converted into dense vector representations, and their similarity is calculated. This metric is particularly well-suited for abstractive summarization, as it captures semantic alignment even when there is no exact word overlap. However, since BERTScore uses pre-trained models, it is computationally intensive, especially for long texts, and its performance heavily depends on the choice of the pre-trained model.

The continuous development of diverse datasets and robust evaluation metrics has been instrumental in advancing Multi-Document Summarization. These resources not only enable the training and fine-tuning of sophisticated models but also ensure their performance can be rigorously assessed across domains and applications.

2.5.4 Applications and Use Cases

MDS is a versatile tool with virtually limitless applications and use cases. Many fields can benefit from synthesizing diverse information originating from various sources. Its ability to reduce data dimensionality and represent relationships effectively makes it a powerful technique for exploring and understanding data. MDS facilitates decision-making processes, enhances access to information, and alleviates information overload.

Among the huge range of applications, news and medicine stand out as two of the most prominent fields where MDS is extensively utilized.

News and Journalism

In journalism, MDS plays a crucial role in summarizing news from different sources. This allows for the generation of comprehensive and impartial summaries of complex and evolving topics by integrating different perspectives. These resumes are particularly valuable in covering events or facts where different points of view must be evaluated. MDS helps readers quickly catch the essence of a topic without being overwhelmed by redundant or contrasting information.

The paper "Automated News Summarization Using Transformers", written by Gupta et al. in 2021 [16], explores the application of Transformer-based models to the task of news summarization. It focuses particularly on the comparison of pre-trained Transformer models on the BBC news dataset, showing they demonstrated strong capabilities in generating coherent and concise summaries. In particular, T5 achieved the best performance, followed by PEGASUS and BART.

Medicine and Biomedical research

In the medical domain, MDS is applied to condense clinical studies and research articles. This proves highly beneficial for healthcare professionals and researchers, who must stay updated on a vast amount of sources. By synthesizing relevant information, MDS reduces the time needed to extract insights, enabling faster and more informed decisions. Datasets Like MS^2 have been specifically designed to support this field, helping models handle the complexity and specificity of biomedical data.

In the article "Summarizing, Simplifying, and Synthesizing Medical Evidence Using GPT-3 (with Varying Success)", published in 2023 by Shaib et al. [17], the

authors try to understand if LLM models, in particular GPT-3, which works very well in the MDS task for general domain news articles in few-shot and zero-shot settings, has the same performance for specialized domains, such as biomedicine. To face this task they developed a multi-granular model, combining it with advanced models based on Transformers and optimization methods, exceeding the results obtained by traditional methods on standard benchmarks for long documents, such as PubMed and ArXiv.

While news and medicine are two major areas of focus, numerous other fields leverage MDS models to address their specific challenges. Some of these domains include:

- **Business Intelligence**: MDS models can be utilized to synthesize insights from market reports and financial data to support strategic decision-making;
- **Market Analysis**: consumer trends, product reviews, and competitor data can be summarized by MDS models to improve marketing strategies;
- **Legal Services**: extracting relevant information from lengthy legal documents with MDS models can assist in case preparation and analysis;
- **Psychology and Social Sciences**: MDS can be leveraged to aggregate qualitative research findings or survey responses into concise summaries for easier and quicker interpretation;
- **Customer Service**: customer support can be improved by summarizing chat logs, support tickets, and FAQs with MDS models to provide faster responses and solutions.

2.5.5 Challenges and Future Developments

MDS remains an auspicious NLP task with the potential to revolutionize how information is managed and synthesized across various sectors. Nevertheless, substantial challenges still need to be addressed to realize NLP potential fully.

One of the main challenges lies in creating summaries that can effectively capture diverse perspectives and highlight distinct aspects of the input documents. Current models often struggle to balance these viewpoints, leading to summaries that may inadvertently introduce biases or omit critical information. Addressing this limitation will require the development of more robust algorithms capable of understanding nuanced relationships and dependencies across multiple sources.

Another important challenge is the interpretability of MDS models. As models become increasingly complex, it is crucial to ensure that their decision-making

processes are transparent and interpretable, particularly in sensitive fields such as medicine, law, and finance. Enhancing interpretability will build trust in the use of MDS systems and make their outputs more reliable for end-users.

Evaluation metrics also present a key area for improvement. Metrics such as ROUGE and BLEU, while widely used, often fail to capture the semantic richness and contextual appropriateness of summaries, especially for abstraction methods. Future developments should focus on creating evaluation frameworks that better align with human judgment, incorporating both qualitative and quantitative aspects of summary quality, such as informativeness, coherence, and factual consistency.

The integration of multimodal data represents another frontier for MDS research. As the availability of data in formats beyond text (e.g., images, videos, and audio) grows, there is an increasing need for models that can seamlessly synthesize information from multiple modalities. This requires advancements in neural architectures capable of jointly processing and understanding heterogeneous data types.

Innovative techniques such as Retrieval-Augmented Generation (RAG) are also paving the way for more reliable and accurate summarization models. By combining retrieval mechanisms with generative capabilities, RAG-based approaches can enhance the factual accuracy and contextual relevance of summaries. This is particularly valuable in domains where up-to-date and precise information is critical.

Scalability and efficiency are additional challenges. As MDS models require extensive computational resources to process large clusters of documents, optimizing these models to reduce memory and energy consumption without compromising performance is a pressing need. Techniques such as model distillation, quantization, and sparse attention mechanisms are expected to play a pivotal role in this area.

2.6 Retrieval-Augmented Generation

Ensuring that language models remain accurate and contextually relevant is still today a significant challenge. RAG addresses this need by combining powerful text generation capabilities with dynamic information retrieval mechanisms. RAG models are more detailed than traditional LLMs and, by leveraging external knowledge sources, provide responses that are not only accurate and contextually coherent but also enriched with up-to-date and domain-specific information. This makes RAG a game-changing approach in the field of NLP.

When the system receives a request, it utilizes a retrieval module to search documents or relevant data from an external knowledge base, such as company datasets or online updated sources. The retrieved information is given to the generative model that utilizes them together with its pre-trained knowledge to elaborate a coherent and informed response.

2.6.1 Architecture

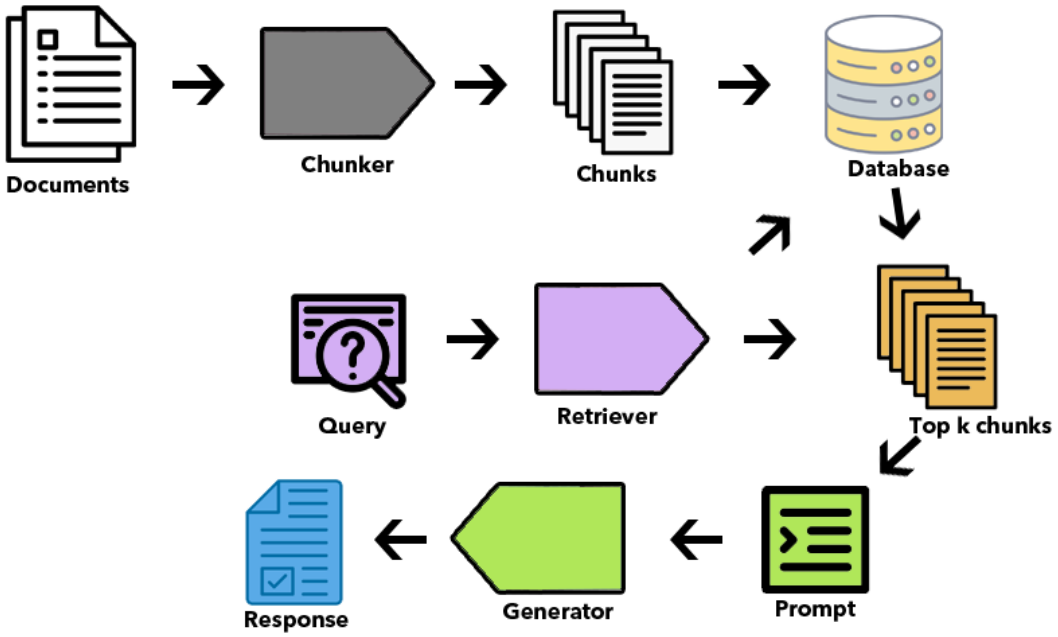


Figure 2.19: A representation of RAG workflow.

The architecture of a RAG model, as shown in Figure 3.5, combines two essential components: a retriever and a generator, which work in tandem to deliver precise and contextually enriched responses.

Retriever

The retriever is responsible for identifying and extracting relevant information from an external knowledge base or dataset. This step ensures the model responds with information enriched with fresh and reliable data. The process involves:

- **Semantic Encoding**: both the user query and the documents in the knowledge base are encoded into numerical embeddings, which capture the semantics and the context of the text;
- **Indexing and Recovery Techniques**:
  - **Keyword Searching**: the system searches for keywords in the document database or corpus to identify the most relevant content. One of the most popular keyword-searching algorithms is Best Matching 25 (BM25);

  - **Dense Vectors Embeddings**: numerical representations of words, sentences, or entire documents in a high-dimensional vectorial space, used to measure semantic similarity between query and documents during the retrieval process. Dense Passage Retrieval (DPR) is a model that generates dense embeddings for queries and text snippets;
  - **Hybrid Methods**: methods that combine approaches based on dense models and traditional approaches based on keywords, to obtain the best of both worlds.

- **Knowledge Sources**: RAG retrievers can access a variety of external data repositories, including company datasets, public knowledge bases like Wikipedia, or domain-specific repositories.

Generator

Once the relevant information has been recovered by the retriever, it is sent to the generator. The generator, typically an LLM trained on a huge textual dataset, produces natural language responses based on the retrieved information and the user's original query. Key features include:

- **Pre-trained Models**: Popular choices include GPT, T5, or FLAN, which are trained on huge textual datasets to generate fluent and contextually appropriate text;
- **Integration with Retrieval Results**: the generator processes the retrieved data to ensure the output is coherent, contextually aligned, and semantically accurate;
- **Output Quality**: the model guarantees that the generated text is fluent, informative, and tailored to the user's needs, balancing precision with readability.

Retrieval and Generation Process

The process starts with the user who provides a question or an input sentence. The retriever analyzes the query, retrieves the most relevant documents or data from the knowledge base, and provides this to the generator. The generator integrates the retrieved information with its pre-trained knowledge to produce a coherent and informative response. This approach allows language models to access a wide range of knowledge, reducing the possibility of incorrect answers.

This architecture enables RAG to deliver responses that are both accurate and up-to-date, making it a versatile solution for a wide range of applications, exploiting personalized datasets to support all systems that should maintain updated information or knowledge of a specific sector.

2.6.2 Applications

RAG models have found applications in various fields requiring accuracy, personalization, and continuously updated information. They improve the efficiency of AI-based support systems.

Some of the most significant use cases for RAGs are Customer Support, Research and Development, and Content Summarization and Generation.

Customer Support and Advanced Chatbots

RAG enhances Chatbots' and Customer Support systems' performances by providing accurate, personalized, and informed answers enhancing the user experience.

These systems can access company databases, FAQs, and internal documentation to resolve user queries efficiently. RAGs can also adapt responses to users' specific needs based on previous interactions, improving customer satisfaction. Also, more complex requests can be handled, such as technical troubleshooting or detailed service bookings.

For instance, a banking chatbot could use an RAG to provide detailed information about financial products by accessing real-time internal documentation. Similarly, the chatbot proposed in this thesis employs RAG to answer specific questions about insurance products.

Research and Development and Medicine

RAG-powered systems support scientific research, helping researchers find relevant information from heterogeneous sources and generate new hypotheses and ideas, stimulating innovation and creativity. RAGs can be employed to summarize and aggregate information from scientific articles, clinical studies, and knowledge databases, offering answers based on scientific proofs and recent updates.

In medicine, for example, a decisional support system for doctors can utilize a RAG to reply to specific clinical questions by accessing recent studies, and helping doctors make informed decisions, based on the evidence.

Content Summarization and Generation

RAGs facilitate summarizing and generating content from diverse sources, such as news articles, company reports, or any other textual content, saving time and resources.

These models can generate marketing materials tailored to specific audiences, but can also produce creative outputs like poems, or source code. For instance, an RAG-based platform could compile personalized daily news summaries, integrating multiple sources to provide a comprehensive overview of complex topics.

Beyond the applications mentioned above, RAG has further potential fields, such as:

- **Education**: developing intelligent tutors capable of retrieving and explaining information across diverse topics;
- **Finance**: providing real-time analyses of market trends and financial products;
- **Legal Services**: summarizing case law and legal documents for professionals;
- **National Security and Intelligence**: synthesizing data from multiple sources for informed decision-making.

Fan et al., in 2024 [18], published a survey on RAG techniques and their usage combined with LLMs. They mainly focused on the way these techniques could improve the generation of AI-based content, facing intrinsic LLM limitations, such as hallucinations and the absence of real-time updates.

Two outstanding examples are :

- A model proposed by Li et al. in the paper "Empowering Molecule Discovery for Molecule-Caption Translation with Large Language Models: A ChatGPT Perspective", written in 2023 [19], in which RAG is applied to ChatGPT for molecule discovery;
- A model presented in the paper "Large Language Models as Conversational Movie Recommenders: A User Study", written by Sun et al. in 2024 [20], where RAGs are utilized for personalized movie recommendations.

2.6.3 Advantages

RAG models offer a large number of advantages over traditional LLMs, especially when having updated, accurate, and specific information is extremely important.

Different from traditional language models, which can generate answers only with the knowledge acquired during the training phase with static data, RAGs can retrieve the latest information, ensuring relevance and accuracy without the need for retraining. They are perfect for answering questions on recent events or topics that frequently change, like news or regulations.

Generative models often output invented or inexact answers, especially if they lack specific knowledge. RAGs can reduce this risk, by utilizing external data as a reference. The integration of various external sources lowers the probability of generating wrong or hallucinated information, increasing the accuracy of the generated text, since the model has access to a larger context. In some specific contexts, such as customer chatbots, precision is critical to offer a high-quality service to the clients.

RAGs can be quickly adapted to new domains by updating the knowledge base, without the need to edit or re-train the generative model. This allows to re-employ RAGs in different fields, simply modifying the knowledge base. Re-adapting easily the model allows also to reduce costs and time to maintain the model updated, for this reason, RAG-powered models are extremely more efficient than traditional ones.

RAGs can extract custom and specific information from various sources, providing answers to very complex questions but also having the possibility of customizing them for the users' needs and preferences. This makes them very well-suited for contexts such as scientific research, academic essays, and legal assistance.

Furthermore, the modular structure of RAGs consents to utilize different tools for information retrieval, such as company databases, internal documents, and knowledge bases. This flexibility makes it easy to integrate them into company workloads, research applications, and assistance systems, making them a perfect versatile tool.

2.6.4 Challenges and Limitations

Despite their advantages, RAGs also have to face various challenges and limitations to guarantee optimal functioning.

The accuracy of the answers given by RAGs directly depends on the quality and pertinence of the external knowledge base. If data are imprecise, incomplete, or not pertinent, the quality of the generated text can be horrible. It is important to ensure that external data are updated and suitable to the specific scope, and also find a way to measure the reliability and relevance of the sources accurately, otherwise, the system could provide not coherent or misleading answers, especially in contexts where the information is conflicting.

The RAG-powered models could also inherit some bias if it is present in the training data or the information sources, and this could lead to offensive or discriminatory answers. It is hard to control it in external data, for this reason, it is necessary to develop techniques to identify and mitigate it in RAG models, for example, by diversifying training data and implementing control mechanisms.

Training, executing, and implementing RAG models require a huge amount of computational resources, thus limiting the accessibility for small projects or organizations. The information retriever and the generator modules should be perfectly synchronized to produce fluid and integrated answers, but synchronization could be very laborious, especially in cases where the retrieved information is fragmented or comes from different sources.

Sometimes the latency, due to the information retrieval phase that is done before generation, could be longer than pure generative models, and this could represent a problem for some specific applications that require real-time answers.

Keeping the external knowledge bases regularly updated has high maintenance costs and requires supervision, especially for dynamic fields such as medicine or finance. Another big challenge is to develop techniques to improve RAG efficiency, reducing computational costs without compromising performance.

Some RAGs can access private and sensitive data sources, which requires an accurate handling of privacy and security to avoid unauthorized data access or data leaks. It is fundamental to guarantee that these kinds of data are protected and that privacy regulations, such as GDPR, are respected.

2.6.5 Future Developments

RAG models are among the most important advancements in the AI field nowadays and their development will continue and will carry significant innovations in various fields. A lot of effort will be put into overcoming actual challenges, to improve access capability and information generation in a more accurate and personalized way.

Future RAG systems will focus on integrating sources with always-updated data, such as dynamic databases or knowledge graphs, to obtain quick and precise answers. The objective is to retrieve real-time information and improve the accuracy and freshness of responses, keeping the models updated without re-training them.

With the evolution of multimodal technology, future RAGs could also retrieve images, videos, and other media to reply to more complex questions. This will pave the way for more creative applications, and the model could combine different types of media for informative or visually enriched answers.

Another possible future development is to integrate RAGs with advanced dialogue models, to keep the context of the conversations in more interaction rounds and provide progressive answers. This will allow smoother conversations, keeping context through time, and raising the capacities of virtual assistance for customer support or medical care.

Latency and scalability are critical challenges for RAGs, future developments will focus on more rapid and flexible architectures, with optimized hardware and more efficient retrieval algorithms. This will be crucial for applications that require instant answers.

More sophisticated techniques to identify and mitigate bias in the RAG models should be developed, guaranteeing impartial and equal answers. Future models could include ethical filters to reduce prejudices and discrimination. Also privacy is an issue so future developments should focus on technologies such as data cryptography and differential privacy to guarantee secure and protected answers.

One final possible evolution could be addressed to improve the handling of the sources from which RAGs retrieve data. Future RAGs could automatize the selection and validation of the sources to guarantee accurate information gathering.

This will ensure answers will be based on verified data and authoritative sources.

2.7 Financial-related MDS works

Certain studies have explored Multi-Document Summarization in the financial domain. Two papers were particularly relevant because they focused on summarizing financial documents that integrate textual and tabular data.

The first work, "Long Text and Multi-Table Summarization: Dataset and Method"[21], published in 2023 by Liu et al., addresses the challenge of summarizing complex financial reports that contain extensive textual content alongside structured numerical data. The authors introduce FINDSum, the first large-scale dataset designed for this task, built from 21125 annual reports of 3794 companies. FINDSum is divided into two subsets, focusing on companies' operating results and liquidity.

The study explores three different summarization strategies: Generate-and-Combine (GC), Combine-and-Generate (CG), and Generate-Combine-and-Generate (GCG), exploiting the BART-large model. Additionally, new evaluation metrics are introduced to assess how effectively numerical information is incorporated into the summaries. This work represents a significant contribution to financial text summarization, offering interesting ideas on pre-processing, content selection, and evaluation tailored to financial documents that integrate both text and tables.

Similarly, the paper "Beyond Pure Text: Summarizing Financial Reports Based on Both Textual and Tabular Data" [22], published in 2023 by Wang et al., explores methods for generating resumes that integrate key information from both textual and tabular data. The authors propose a novel approach that applies data abstraction techniques to extract and synthesize relevant information from both formats. A new dataset, called USTT [5], was developed for this purpose. It consists

Unnamed: 0 int64	id string	table string	text string	summary string	table_title string
6,368	n_9158	{"2022年第一季度":{"国内生产总值当季值(亿元)":270177.8,"国内生产总值累计值…	事件：2022年3月31日，国家统计局公布2022年3月PMI数据，制造业…	核心观点：3月以来，国内多地出现聚集性疫情，企业生产经营活动受…	2022年3月PMI数据点评：疫情冲击经济动能，稳增长仍需发力
701	n_10665	{"2022年第二季度":{"国内生产总值当季值(亿元)":292463.8,"国内生产总值累计值…	资产表现及资金变化：国内商品涨跌幅前五：乙二醇4.14%、菜粕2.91%、…	美国5月贸易帐逆差855亿美元，预期849亿美元，前值逆差868.9亿…	宏观策略日报：总理主持召开经济形势座谈会，稳经济还需艰苦努力
6,665	n_10452	{"2022年5月":{"国家财政收入累计值(亿元)":86739.0,"国家财政收入累计增长…	事件：财政部发布数据显示，1-5月累计，全国一般公共预算收入86739亿…	全国一般公共预算支出99059亿元，比上年同期增长5.9%。 核心要点…	5月财政数据点评：内需稳步复苏，支出持续发力
1,821	n_1430	{"2018年第一季度":{"国内生产总值当季值(亿元)":202035.7,"国内生产总值累计值…	结论或者投资建议：2017年TFP明显改善，是经济超预期回升的主要动力。…	从来源上，TFP的增长来源于资源的优化配置、技术的提高等，与改革…	国内中观周度观察：经济复苏的真相：TFP改善
180	n_5447	{"2021年6月":{"居民消费价格指数(上年同月=100)":101.1,"食品烟酒类居民消费价…	要闻：国家粮食和物资储备局将于近期分批投放铜、铝、锌等国家储备。 投…	美联储FOMC6月点阵图显示，18位官员中有13人支持在2023年底前至少…	宏观大类日报：美联储释放鹰派信号 黄金短期警惕风险

Figure 2.20: Some rows from the USTT Dataset.

[4]huggingface.co/datasets/wza/USTT

of triplets in the form (table|text|human-generated summary). In this approach, tabular data are encoded both at the row and column levels, while textual data are processed at the sentence level using pre-trained models. To improve the selection of relevant content, the authors introduce a "salient detector gate", which identifies and prioritizes the most critical information to include in the final resume. The proposed architecture consists of four key components:

- An **encoder**, for tabular and textual data;
- A **content detection** module to identify relevant information;
- An **external knowledge base**, designed to mitigate selective bias issues;
- The **unbiased summarization generator**, responsible for generating the final summary.

The effectiveness of this approach was validated through both automatic and human evaluation criteria. The authors employed ROUGE-1, ROUGE-2, ROUGE-L, and BERTScore as automated metrics, while two groups of independent annotators with similar backgrounds assessed the results based on coverage and salience, using a scale from 1 to 5.

This study represents a significant advancement in the automatic summarization of financial reports. It demonstrates how MDS techniques can be adapted to handle both textual and numerical information, an essential aspect of financial applications.

Given the specific focus of this thesis, these two datasets were initially considered as potential baselines for financial summarization tasks. However, after a detailed evaluation, both resulted in being unsuitable for this project. FINDSum, despite integrating both numerical (tabular) and textual data, lacked a clear linkage between textual descriptions and tabular data, making it difficult to establish direct references between them. USTT provided a clearer structure for text-table relationships, but the dataset is entirely in Chinese, introducing an additional layer of complexity to the project. Translating the dataset into English would have increased the complexity of the project, leading to the decision to exclude it.

As a result, alternative approaches and datasets were explored to better meet this work's specific requirements.

2.8 Chain-of-Event prompting

Since the most promising financial-related works could not be used as a baseline for this project, the focus shifted back to the news domain to explore innovative techniques that could potentially be adapted to financial tasks.

In 2024, Bao et al. introduced a revolutionary approach in their paper "Chain-of-event prompting for multi-document summarization by large language models" [23]. The Chain-of-Event (CoE) technique was specifically designed to enhance Multi-Document Summarization using Large Language Models. While LLMs have demonstrated outstanding performance for NLP tasks, standard zero-shot, and few-shot prompting approaches often struggle to generate concise and relevant summaries for MDS. CoE addresses this limitation by improving prompt engineering strategies.

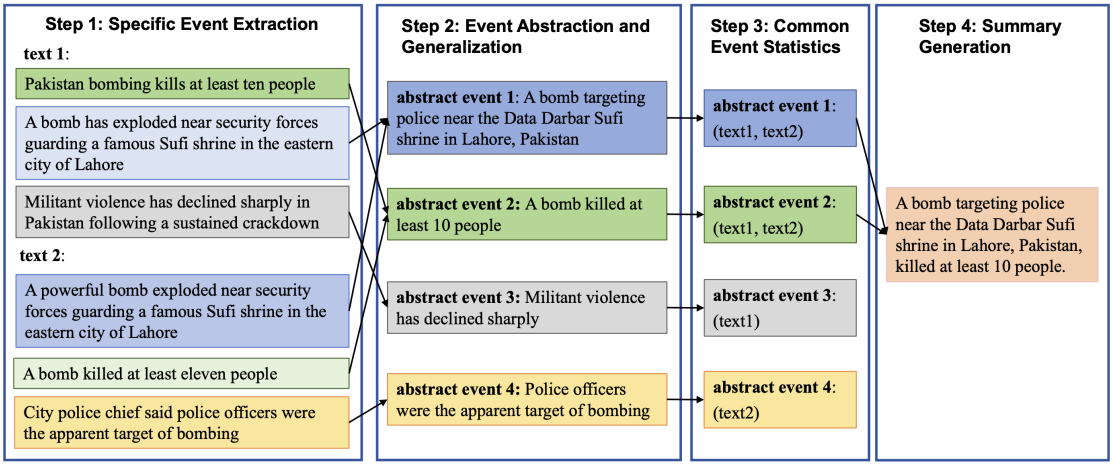


Figure 2.21: An example of the Chain-of-Events steps for the resume generation. Source: [23]

CoE follows a Chain-of-Thought (CoT) prompting approach, without requiring fine-tuning. The process consists of four sequential stages, designed to mimic human reasoning:

1. **Specific event extraction**: The model identifies and extracts fine-grained events from the input texts.

2. **Event abstraction and generalization**: Extracted events are grouped into more concise and generalized categories.

3. **Common event statistics**: The model selects the most frequently occurring events across the input documents.

4. **Summary generation**: The selected key events are structured into a coherent and concise summary, maintaining either chronological order or relevance-based ordering.

Before executing these steps, the LLM is provided with a task description and a few-shot examples of the reasoning process. These examples guide the model toward generating summaries that align with expectations.

This approach was tested on two datasets: Wikipedia Current Events Portal (WCEP) and Chinese Call Center Textual Corpus (CCTC).

WCEP is a dataset created in 2020 by Gholipour et al. [24], which contains short human-written summaries of news events collected from the Wikipedia Current Events Portal[6]. The dataset consists of over 10000 document clusters (of which almost 8000 were selected for experiments), with varying numbers of articles per cluster.

CCTC is a Chinese dataset containing 1,200 textual customer interactions corresponding to 100 customer complaints in call center conversations. All sensitive data has been removed to ensure privacy compliance.

The evaluation of CoE used both automatic and human metrics. The automatic evaluation included ROUGE-1, ROUGE-2, ROUGE-L, BERTScore, and G-Eval. For human evaluation, two analysts assigned scores based on conciseness, relevance, and fluency. The final evaluation score was computed as the average of the two reviewers' scores.

CoE demonstrated notable improvements over traditional prompting techniques, proving to be an effective way to enhance summary quality. Given its strong performance, CoE was selected as the baseline technique for this project, with prompt modifications tailored for financial document summarization.

[6]en.wikipedia.org/wiki/Portal:Current_events

Chapter 3

Use Case

This work presents a possible use case for Multi-Document Summarization techniques in the insurance sector.

The project was developed during an internship at the company Data Reply [1], where it was designed and implemented within a real-world business context.

Data Reply is a company within the Reply group that specializes in data analytics and artificial intelligence services across various industries. It supports executives in leveraging data to drive business outcomes by employing advanced technologies such as Big Data Engineering, Data Science, and AI/ML model implementation. Additionally, the company focuses on building scalable and secure data platforms.

The specific business unit involved in this project, "Big Data Engineering for Financial Services Industry and Security", specializes in data analytics and security for financial and insurance companies. Within this framework, the goal was to develop a chatbot that would assist insurance customers in summarizing and comparing different products, utilizing state-of-the-art Natural Language Processing and Artificial Intelligence techniques.

Multi-Document Summarization is a well-established subfield of Natural Language Processing that has seen significant advancements in recent years. As discussed in Chapter 2, its primary applications have traditionally focused on news and journalism, with many research efforts dedicated to generating high-quality news summaries. However, despite the vast amount of literature in this domain, its application in the economic sector, particularly in the insurance industry, the core focus of the Business Unit where this thesis was developed, remains relatively unexplored.

The project's first phase involved reviewing recent research on MDS applications in similar domains. These studies were used as a baseline to develop and compare

[1] reply.com/data-reply

new solutions.

3.1 Experiments with LangChain in financial domain

Since CoE proved effective in leveraging Generative AI for summarization, the next step was to apply this approach in a real-world finance use case.

To facilitate the implementation of LLM-based summarization, the project exploited LangChain[2], a framework specifically designed for developing applications that integrate LLMs. LangChain provides modular components that allow easy integration with external data sources, making it a powerful tool for building scalable NLP pipelines.

Some of LangChain's strengths include its ability to integrate external sources and generate modular pipelines, its support for conversational memory (enabling models to retain context across interactions), its ability to handle complex prompts, and its support for multiple languages and models.

As a baseline for the initial experiments, the LangChain Summarization Use Case documentation [25] provided valuable examples and guidelines for implementing summarization workflows. This resource was instrumental in setting up the foundation for the final model. The LangChain documentation outlines three primary MDS techniques, each with distinct strengths and weaknesses:

1. **Stuff**: This method takes a list of documents, inserts them directly in a prompt, and uses them as input for a single call to the chosen LLM. All the documents will be concatenated inside a single text block, that is passed to the model for resume generation. It works well for short documents, is also very fast, and can count on full context awareness since all documents are processed at the same time. On the other side, it struggles with long inputs, since the model token limit can be reached easily, and making this technique ineffective and not usable.

2. **Map-Reduce**: This approach splits the resume chain into two different steps: map and reduce. During the Map phase, an LLMChain will summarize each document individually. In the Reduce phase, the individual summaries are merged into a final summary. This approach handles large document collections efficiently if documents are very rich in text, without reaching the token limit, but the final summary could lose some connections between documents, resulting in a lack of coherence if compared to Stuff summaries.

[2]langchain.com

3. **Refine**: This document chain loops over the input documents creating an initial summary and then iteratively updating it, by incorporating new information from subsequent documents. The advantage is that the model maintains a progressive context, avoiding information loss problems, so it adapts to documents with distributed information or incremental updates. On the other side, it is slower with respect than other approaches and the summary quality strongly relies on the order in which the documents are elaborated.

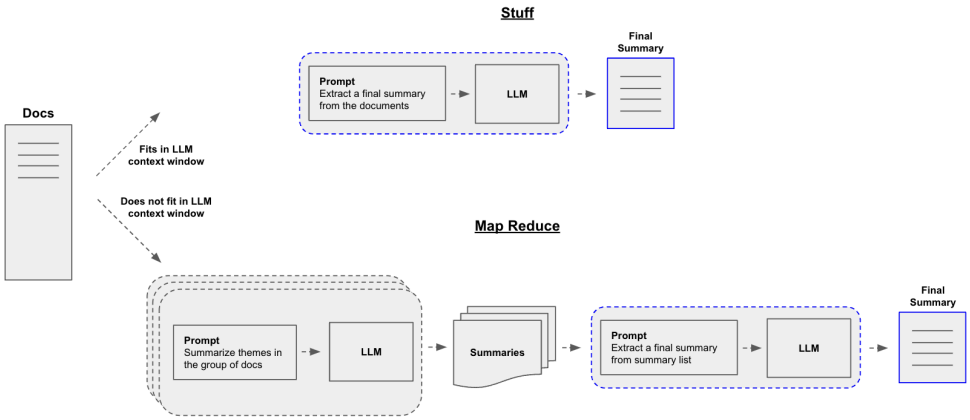


Figure 3.1: Comparison between LangChain Stuff and Map Reduce techniques. Source: [25]

After conducting preliminary tests on all three LangChain methods, for the specific case of this project, the Stuff and Map-Reduce techniques were selected as the most compatible with CoE prompting and were used as a baseline. Initial experiments were performed using the WCEP dataset, applying some variations:

- **Base Stuff Prompt**: A simplified version of the Stuff technique defined in the LangChain Documentation, slightly modified to include in the prompt all the documents (articles) and separate them using the text "_NEW ARTICLE_" for better clarity;
- **Base Map-Reduce Prompt**: Adapted from LangChain's Map-Reduce documentation, ensuring the map step extracts key themes, while the reduce step consolidates the summary;
- **CoE Stuff Prompt**: A modified CoE prompt, adapted to fit the Stuff technique, ensuring structured reasoning by clearly guiding the model through the four CoE stages. This prompt also provides some examples to make the

model produce a summary that is as expected. Also in this case, all the articles are included in the prompt, separated by the text "_NEW ARTICLE_".

- **CoE Map-Reduce Prompt**: A modified CoE prompt, adapted to perform a resume with the Map-Reduce technique. Again the prompt specifies the four CoE stages that should be performed to write a good resume of the provided articles but, in this case, the map phase performs only the event extraction step (the first CoE stage) and the reduce step performs all the remaining ones. Again, some examples of the various steps are provided to help the model perform well.

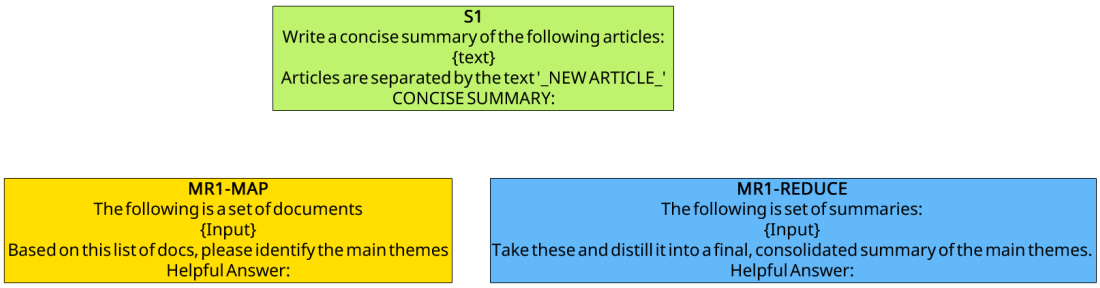


Figure 3.2: A comparison between the modified LangChain base prompts (Stuff version and Map-Reduce version) used for the experiments.

3.2 Evaluation Metrics

To assess the effectiveness of the proposed model, the evaluation methodology used in the Chain-of-Event paper has been considered as a starting point. Specifically, the automatic evaluation metrics employed in that work have also been adopted in this project to analyze the results obtained using the LangChain framework. For evaluating the outputs of different prompts, the following traditional supervised metrics were used: ROUGE-1, ROUGE-2, ROUGE-L, ROUGE-LSUM, BERTScore, BLEU, and G-EVAL. These metrics rely on comparing the generated summary with a human-written reference summary from the WCEP dataset, and for this reason, can be considered supervised. The reference summaries were manually created by two human annotators following predefined criteria.

However, this approach presents significant limitations. The evaluation score is highly dependent on textual similarity between generated and reference summaries rather than an actual assessment of summary quality. Furthermore, this evaluation method is dataset-specific, it can only be applied to a dataset that already includes human-generated reference summaries. These metrics fail to measure critical

semantic aspects, such as coherence and factual consistency which are crucial for high-quality summarization. Given these constraints, these supervised metrics were considered as secondary indicators and not as the primary method for selecting the best-performing prompt.

To overcome the limitations of traditional evaluation techniques, the G-EVAL framework was adopted as a complementary evaluation strategy. This framework, proposed in the paper "G-EVAL: NLG Evaluation using GPT-4 with Better Human Alignment"[26], aims to enhance text evaluation by exploiting LLMs as evaluators, reducing dependence on predefined reference summaries.

G-EVAL follows a Chain-of-Thought approach, breaking the evaluation into structured reasoning steps. Unlike ROUGE or BLEU, which rely solely on lexical overlap, G-EVAL allows LLMs to assess text quality based on intrinsic properties such as factual consistency, fluency, and coherence.

The paper tested G-EVAL on three datasets (SummEval, Topical-Chat, and QAGS) and demonstrated that it outperforms traditional human evaluation in terms of alignment with expert judgments. However, it also introduces potential biases, as LLM-based evaluators may favor AI-generated content over human-written text.

G-EVAL was initially employed in this project as a supervised evaluation metric, comparing WCEP reference summaries with the generated ones. However, to develop a more generalizable evaluation framework, an unsupervised adaptation of G-EVAL was explored. In the first phase, the evaluation focused on a single metric called "Correctness", measuring whether the generated summary accurately reflected the original documents' events. Subsequently, inspired by the evaluation process in the G-EVAL paper[3], a more comprehensive assessment framework was introduced, incorporating four key metrics:

- **Coherence**: Assesses how well the individual sentences in the summary flow together;
- **Consistency**: Evaluates whether the generated summary is factually aligned with the sources;
- **Fluency**: Examines grammar, spelling, punctuation, word choice, and overall readability;
- **Relevance**: Measures the extent to which the summary captures the most important content from the original documents.

Each metric was evaluated independently using dedicated prompts that guided the model through a structured reasoning process. The prompts follow a CoT-based

[5]github.com/nlpyang/geval

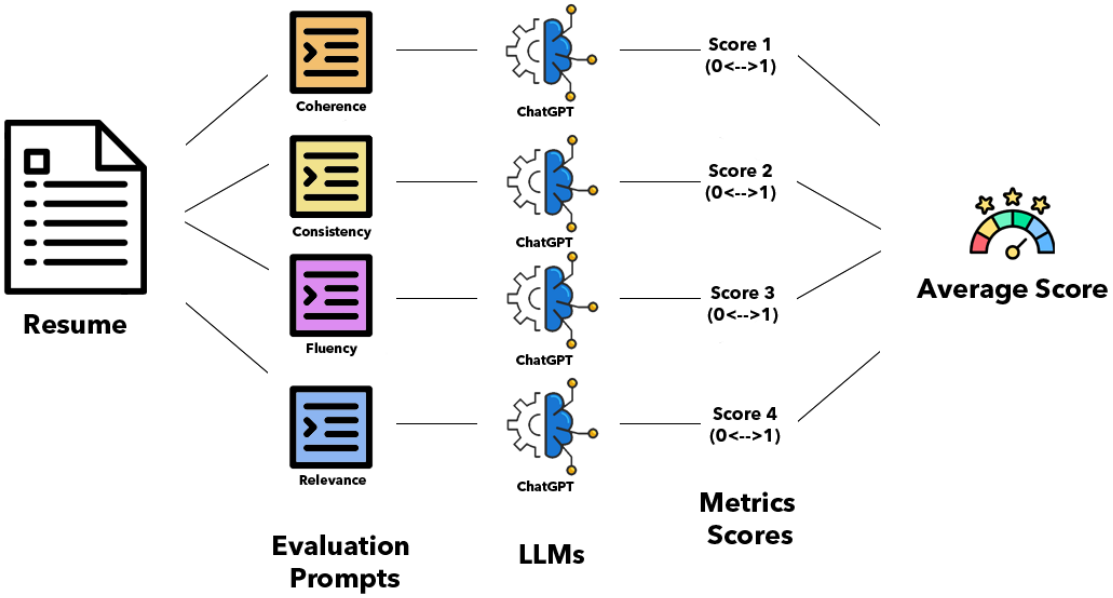


Figure 3.3: A schematization of the evaluation mechanism adopted for the resume evaluation.

approach, explicitly listing evaluation criteria and including examples to help the model assign a score between 0 and 1. The final evaluation score is computed as the average of these four metrics.

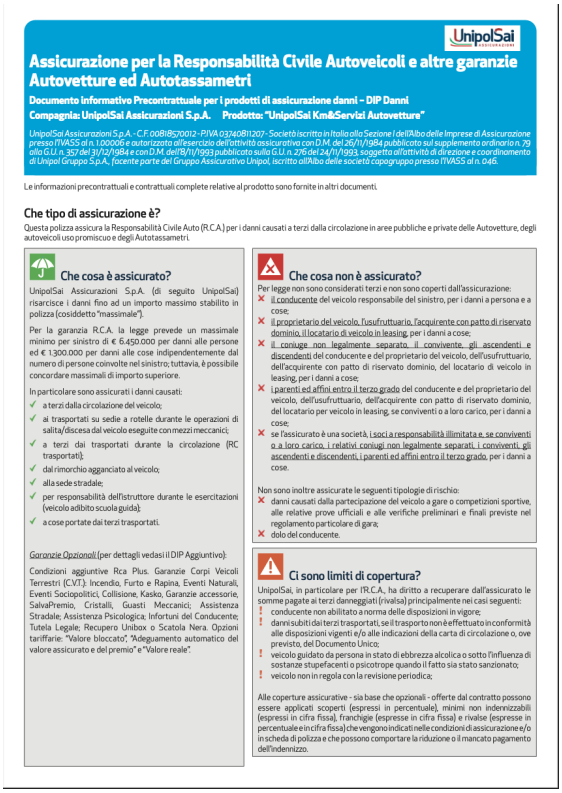
This new approach to evaluating the produced summaries proved more effective than the traditional evaluation techniques, ensuring a more context-aware and qualitative assessment. For this reason, it was the only one considered for selecting the most promising prompts and choosing the final version of the model.

3.3 Financial Dataset

As previously discussed, the initial experiments were conducted using the WCEP Dataset, which contains clusters of news articles. However, since the primary objective of this project is to perform Multi-Document Summarization in the financial and insurance domain, a new dataset was created using publicly available insurance-related documents from the Unipol company website [4].

Among the various insurance products offered by Unipol, 16 products were selected, and for each, the corresponding official documentation was collected.

[4]unipol.it



UnipolSai

Assicurazione per la Responsabilità Civile Autoveicoli e altre garanzie Autovetture ed Autotassametri

Documento informativo Precontrattuale per i prodotti di assicurazione danni – DIP Danni

Compagnia: UnipolSai Assicurazioni S.p.A. **Prodotto: "UnipolSai Km&Servizi Autovetture"**

UnipolSai Assicurazioni S.p.A. - C.F. 00818570012 - P.IVA 03740811207 - Società iscritta in Italia alla Sezione I dell'Albo delle Imprese di Assicurazione presso l'IVASS al n. 1.00006 e autorizzata all'esercizio dell'attività assicurativa con D.M. del 26/11/1984 pubblicato sul supplemento ordinario n. 79 alla G.U. n. 357 del 31/12/1984 e con D.M. del 9/11/1993 pubblicato sulla G.U. n. 276 del 24/11/1993, soggetta all'attività di direzione e coordinamento di Unipol Gruppo S.p.A., facente parte del Gruppo Assicurativo Unipol, iscritto all'Albo delle società capogruppo presso l'IVASS al n. 046.

Le informazioni precontrattuali e contrattuali complete relative al prodotto sono fornite in altri documenti.

Che tipo di assicurazione è?

Questa polizza assicura la Responsabilità Civile Auto (R.C.A.) per i danni causati a terzi dalla circolazione in aree pubbliche e private delle Autovetture, degli autoveicoli uso promiscuo e degli Autotassametri.

Che cosa è assicurato?

UnipolSai Assicurazioni S.p.A. (di seguito UnipolSai) risarcisce i danni fino ad un importo massimo stabilito in polizza (cosiddetto "massimale").

Per la garanzia R.C.A. la legge prevede un massimale minimo per sinistro di € 6.450.000 per danni alle persone ed € 1.300.000 per danni alle cose indipendentemente dal numero di persone coinvolte nel sinistro; tuttavia, è possibile concordare massimali di importo superiore.

In particolare sono assicurati i danni causati:

- ✓ a terzi dalla circolazione del veicolo;
- ✓ ai trasportati su sedie a rotelle durante le operazioni di salita/discesa dal veicolo eseguite con mezzi meccanici;
- ✓ a terzi dai trasportati durante la circolazione (RC trasportati);
- ✓ dal rimorchio agganciato al veicolo;
- ✓ alla sede stradale;
- ✓ per responsabilità dell'istruttore durante le esercitazioni (veicolo adibito scuola guida);
- ✓ a cose portate dai terzi trasportati.

Garanzie Opzionali: (per dettagli vedasi il DIP Aggiuntivo):

Condizioni aggiuntive Rca Plus. Garanzie Corpi Veicoli Terrestri (C.V.T.): Incendio, Furto e Rapina, Eventi Naturali, Eventi Sociopolitici, Collisione, Kasko, Garanzie accessorie, SalvaPremio, Cristalli, Guasti Meccanici; Assistenza Stradale; Assistenza Psicologica; Infortuni del Conducente; Tutela Legale; Recupero Unibox o Scatola Nera. Opzioni tariffarie: "Valore bloccato", "Adeguamento automatico del valore assicurato e del premio" e "Valore reale".

Che cosa non è assicurato?

Per legge non sono considerati terzi e non sono coperti dall'assicurazione:

- ✗ il conducente del veicolo responsabile del sinistro, per i danni a persona e a cose;
- ✗ il proprietario del veicolo, l'usufruttuario, l'acquirente con patto di riservato dominio, il locatario di veicolo in leasing, per i danni a cose;
- ✗ il coniuge non legalmente separato, il convivente, gli ascendenti e discendenti del conducente e del proprietario del veicolo, dell'usufruttuario, dell'acquirente con patto di riservato dominio, del locatario di veicolo in leasing, per i danni a cose;
- ✗ i parenti ed affini entro il terzo grado del conducente e del proprietario del veicolo, dell'usufruttuario, dell'acquirente con patto di riservato dominio, del locatario per veicolo in leasing, se conviventi o a loro carico, per i danni a cose;
- ✗ se l'assicurato è una società, i soci a responsabilità illimitata e, se conviventi o a loro carico, i relativi coniugi non legalmente separati, i conviventi, gli ascendenti e discendenti, i parenti ed affini entro il terzo grado, per i danni a cose.

Non sono inoltre assicurate le seguenti tipologie di rischio:

- ✗ danni causati dalla partecipazione del veicolo a gare o competizioni sportive, alle relative prove ufficiali e alle verifiche preliminari e finali previste nel regolamento particolare di gara;
- ✗ dolo del conducente.

Ci sono limiti di copertura?

UnipolSai, in particolare per l'R.C.A., ha diritto a recuperare dall'assicurato le somme pagate ai terzi danneggiati (rivalsa) principalmente nei casi seguenti:

- ! conducente non abilitato a norma delle disposizioni in vigore;
- ! danni subiti dai terzi trasportati, se il trasporto non è effettuato in conformità alle disposizioni vigenti e/o alle indicazioni della carta di circolazione o, ove previsto, del Documento Unico;
- ! veicolo guidato da persona in stato di ebbrezza alcolica o sotto l'influenza di sostanze stupefacenti o psicotrope quando il fatto sia stato sanzionato;
- ! veicolo non in regola con la revisione periodica;

Alle coperture assicurative - sia base che opzionali - offerte dal contratto possono essere applicati scoperti (espressi in percentuale), minimi non indennizzabili (espressi in cifra fissa), franchigie (espresse in cifra fissa) e rivalse (espresse in percentuale e in cifra fissa) che vengono indicati nelle condizioni di assicurazione e/o in scheda di polizza e che possono comportare la riduzione o il mancato pagamento dell'indennizzo.

Figure 3.4: An example of a page from a document of the dataset.

Despite the differences between the selected products, their documents share a common structure, which includes the following sections:

- **Pre-contractual Information Document**: A summary of the key characteristics of the insurance product;
- **Additional Pre-contractual Information Document**: A more detailed explanation of the product's features, expanding on the pre-contractual summary;
- **Glossary**: A list of technical terms used in the document, providing definitions for policyholders;
- **Contract Rules**: The set of rules governing the insurance contract, including terms, conditions, and exclusions;
- **Product Specific Sections**: Custom sections tailored for the insured object. These sections vary across products, providing details that are relevant for each of them.

Although only one document per product was collected, each document was segmented into multiple sections, effectively transforming it into a multi-document dataset. This segmentation allows for more granular analysis and enables different summarization approaches.

Table 3.1 provides an overview of the number of pages and text tokens contained in each document. The dataset can be exploited for two distinct applications,

Product	**Pages**	**Tokens**
Km&Servizi Autovetture	224	184603
KM&Servizi Autocarri	204	166801
KM&Servizi Camper	204	166801
Km&Servizi Ciclomotori e Motocicli	184	151006
Casa & Servizi	124	118606
Condominio Più	120	110899
Nuova Prima Global Veicoli diversi da autovetture, ciclomotori e motocicli	156	102616
Nuova Prima Global CVT e Assistenza stradale	156	99855
InViaggio Full	68	92339
Navigare	84	85605
Nuova Prima Global Autovetture	124	83273
InViaggio Ready	64	72583
InViaggio Frequent	56	63522
Nuova Prima Global Ciclomotori e Motocicli	92	58108
Nuova Prima Global CVM e Assistenza	92	53654
Nuova Prima Global Natanti e Motori Amovibili	56	32143
Average	125.5	102650.875

Table 3.1: Pages and Tokens contained in each product document

depending on how the document sections are grouped:

- **Vertical Mode**: Summaries are generated by treating the different sections of a single product document as separate input documents, aiming to synthesize key information across all sections of a single product;

- **Horizontal Mode**: Summaries are generated by comparing similar sections across different product documents, enabling a cross-product comparison. This

approach is particularly useful for assisting customers in selecting the most suitable insurance product based on their needs.

Products	Documents
¯ Km&Servizi Autovetture	12
KM&Servizi Autocarri	11
KM&Servizi Camper	11
Km&Servizi Ciclomotori e Motocicli	10
Casa & Servizi	8
Condominio Più	7
Nuova Prima Global Veicoli diversi da autovetture, ciclomotori e motocicli	7
Nuova Prima Global CVT e Assistenza stradale	7
InViaggio Full	6
Navigare	6
Nuova Prima Global Autovetture	6
InViaggio Ready	5
InViaggio Frequent	4
Nuova Prima Global Ciclomotori e Motocicli	4
Nuova Prima Global CVM e Assistenza	4
Nuova Prima Global Natanti e Motori Amovibili	3
Average	6.9375

Table 3.2: Number of extracted sections (documents) per each product

This dual approach allows the dataset to support multiple summarization tasks, from in-depth analysis of individual products to comparative summarization across different offerings.

3.4 Final Product

After successfully applying the Chain-of-Thought technique to insurance-related documents, the next step was to develop a tool that could demonstrate the summarization and comparison capabilities in a real-world business scenario. The goal was also to make this AI-powered support accessible to users through an intuitive interface.

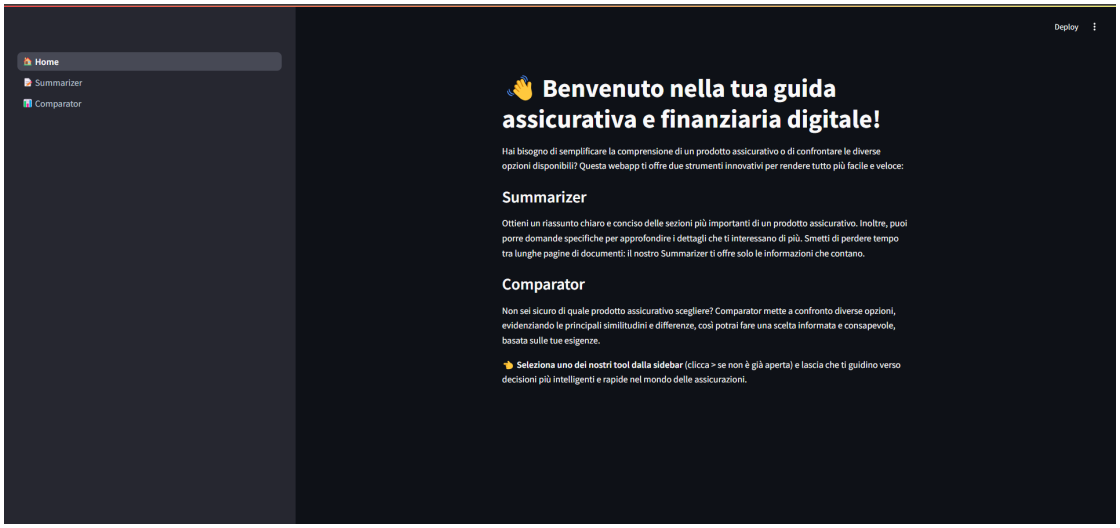


Figure 3.5: The demo homepage, created with Streamlit. The two sections (Summarizer and Comparator) can be chosen from the menu on the left.

To achieve this, Streamlit[5]was chosen as the framework for building an interactive web application. Streamlit is an open-source Python Framework designed for rapidly developing data-driven web applications, without requiring extensive knowledge of HTML, CSS, or JavaScript. It is particularly effective for AI, Machine Learning, and data visualization applications, as it seamlessly integrates with major data science libraries.

The final product of this thesis is a chatbot designed to assist customers in understanding and comparing insurance products the company offers. The chatbot includes two core functionalities:

- **Summarizer**: Generates a concise summary of all the documents related to a specific insurance product, allowing customers to quickly grasp its key characteristics;
- **Comparator**: Compares two different insurance products by summarizing their respective documents and highlighting their similarities and differences, helping users make informed decisions.

Internally, the chatbot utilizes LangChain to manage prompt engineering and document processing. The model employed for text generation is ChatGPT-4o Mini, a lighter and faster variant of ChatGPT-4o, which was released during the development of this thesis. Experimental results demonstrated that ChatGPT-4o

[5]streamlit.io

Mini significantly outperforms ChatGPT-3.5 in summarization tasks, particularly in document comparison and similarity/difference detection. The improvements in coherence and accuracy between the two versions are evident in the evaluation results presented in the next sections.

Beyond summarization, the chatbot also integrates a Retrieval-Augmented Generation (RAG) system to enhance its capabilities. This feature allows users to ask specific questions about an insurance product, retrieving relevant information from the documents. The RAG pipeline employs document vectorization to efficiently retrieve the most pertinent sections before generating a response using ChatGPT-4o Mini. This approach ensures that responses are not only concise but also grounded in the original content, reducing the risk of hallucinated information.

Chapter 4

Solution Design and Implementation

4.1 Design Strategy

This section presents the final outcome of this thesis: an AI-powered insurance chatbot, analyzed from an architectural perspective, focusing on its design.

Through graphical schematizations, the section will illustrate the technologies employed and the operational flow that enables the generation of summaries and customer interactions. The implementation details of the code will be covered in the following section.

4.1.1 System Overview and Architecture

Figure 4.1 provides a high-level overview of the architecture behind the financial chatbot developed in this thesis. The system is structured around a Streamlit-based chatbot interface, which serves as the main entry point for user interactions. Users can select different functionalities from the home page, including product summarization, product comparison, and the ability to ask specific questions about insurance policies.

At the core of the system lies ChatGPT-4o Mini, the chosen Large Language Model, which plays multiple roles in the chatbot's architecture:

- **Conversational Interface**: The model is used to generate responses for user interactions, ensuring a natural and engaging dialogue;
- **Summarization Engine**: Upon request, ChatGPT-4o Mini generates concise and structured summaries of insurance-related documents, exploiting advanced Multi-Document Summarization techniques;

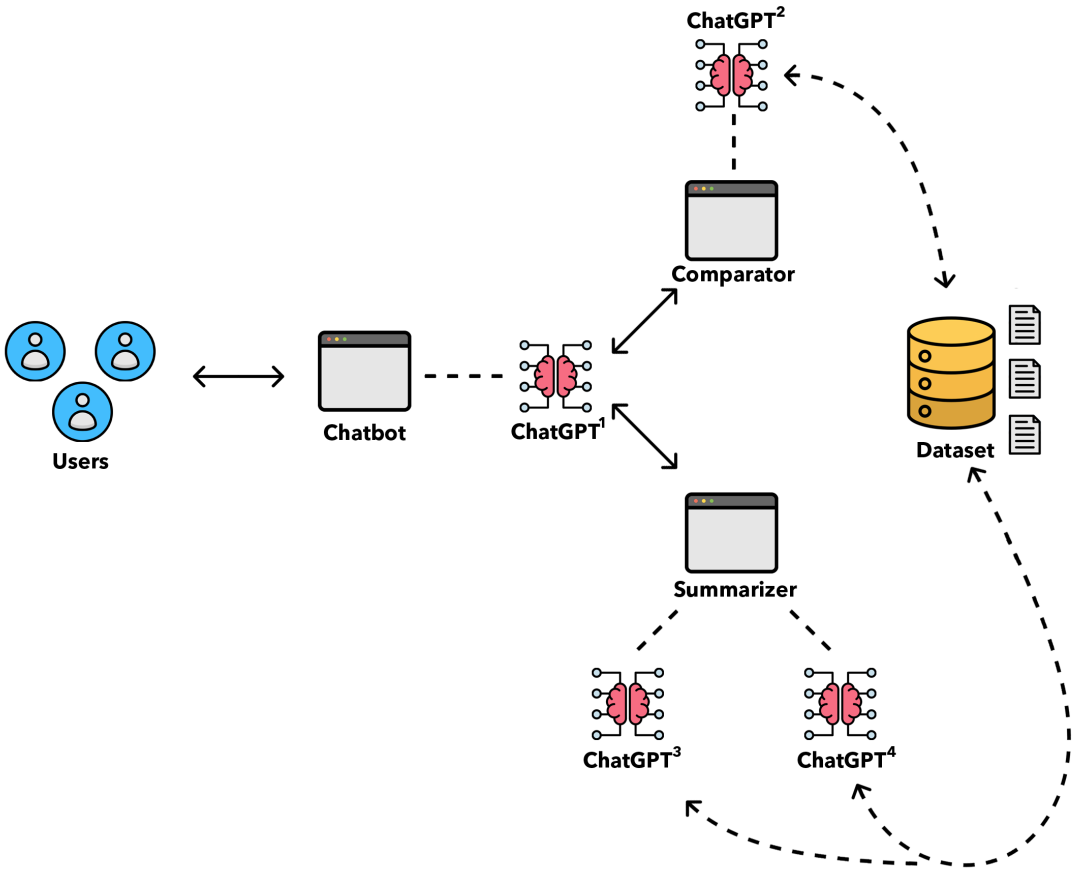


Figure 4.1: A schematization of the system architecture.

- **Retrieval-Augmented Generation**: The system incorporates RAG to improve the accuracy and relevance of responses. When users pose specific questions, the chatbot retrieves relevant documents from the dataset and provides answers based on both retrieved knowledge and the LLM's pre-trained capabilities.

A crucial part of the architecture is the document retrieval mechanism, which allows the chatbot to extract the most relevant information before generating responses to users. This ensures that the chatbot does not solely rely on its pre-trained knowledge but can also provide real-time, fact-based responses based on the latest available documents.

The design of this architecture offers several key advantages, including:

- **Scalability**: The modular approach allows for easy integration of additional functionalities or model upgrades;

- **Flexibility**: The system can be adapted to other domains beyond insurance and generalized by modifying the retrieved knowledge base;
- **Improved Accuracy**: Thanks to RAG, responses are more precise and grounded in actual data, reducing hallucinations and inaccuracies that might arise from pure LLM-generated text.

Overall, this architecture ensures an efficient, responsive, and domain-aware chatbot experience, leveraging state-of-the-art NLP and LLM-based techniques to assist users in quickly and easily obtaining insurance-related information.

4.1.2 Technological Stack

As shown in Figure 4.2, the chatbot system utilizes a variety of technologies, each playing a crucial role in different stages of interaction, data processing, and summarization. The system consists of a front-end, a back-end, a document management module, and an evaluation framework.

The chatbot's user interface is built with Streamlit, a Python-based framework for developing interactive web applications. Streamlit was chosen because of its ease of use, seamless integration with Python, and built-in support for real-time updates. Unlike traditional front-end frameworks such as React or Flask-based interfaces, Streamlit allows rapid prototyping and deployment without extensive web development experience.

OpenAI's ChatGPT-4o Mini and LangChain are the core of the chatbot's back-end. They work together to handle text generation, summarization, and question-answering tasks.

ChatGPT-4o Mini is responsible for:

- Generating abstractive summaries of insurance-related documents;
- Comparing two insurance products based on predefined prompts;
- Engaging in natural conversations with users.

LangChain serves as the framework for managing and optimizing LLM interactions:

- It structures prompt chains using the LLMChain module, ensuring that inputs are formatted properly for the ChatGPT-4o Mini;
- It integrates Retrieval-Augmented Generation via Facebook AI Similarity Search (FAISS), enabling the chatbot to retrieve relevant insurance documents before generating responses;

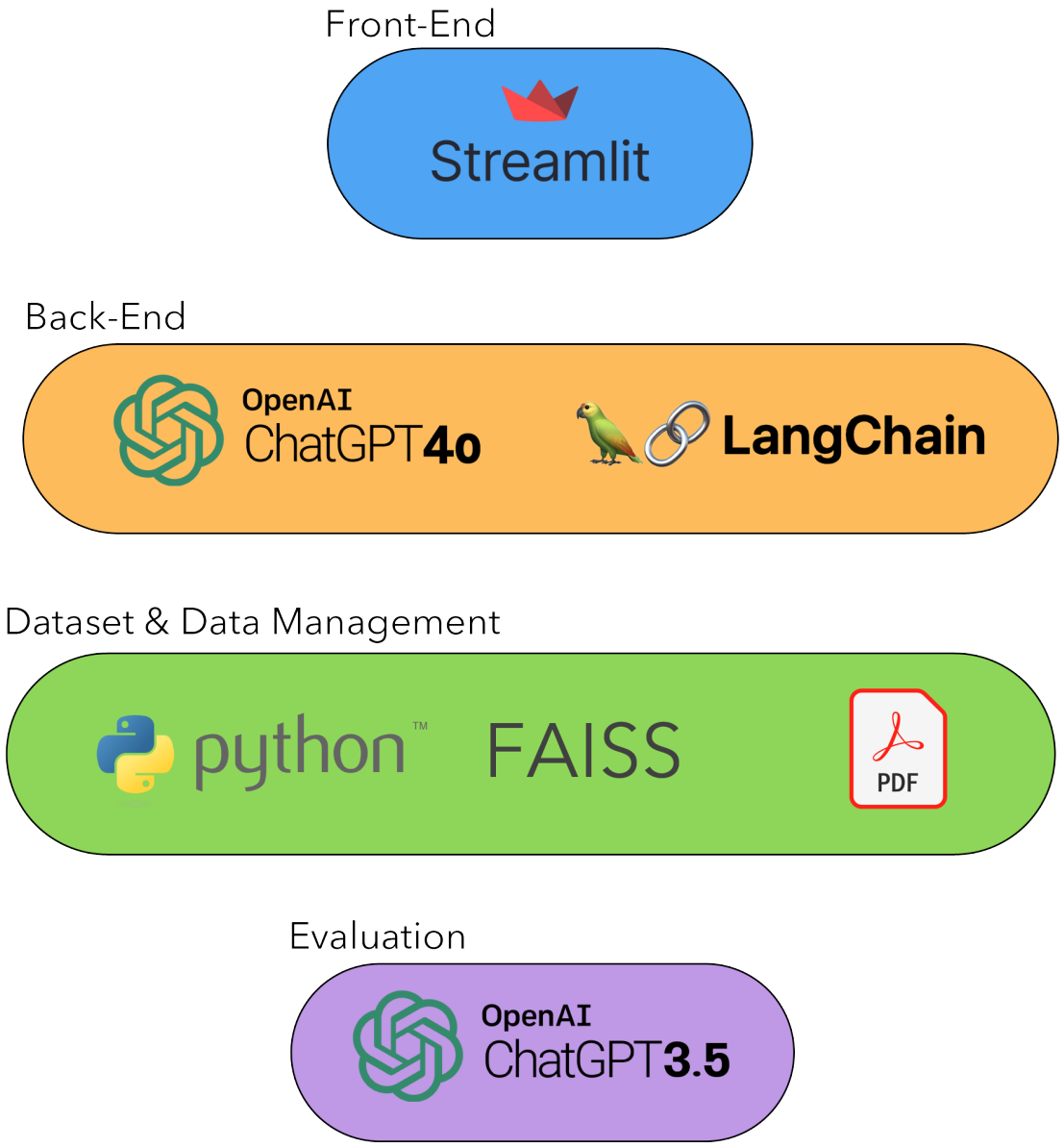


Figure 4.2: All the technologies involved in the project.

- It supports multi-turn memory, ensuring that the chatbot maintains conversational context across interactions.

The dataset used in this project consists of PDF documents related to insurance products, which need to be efficiently processed before being used by the chatbot. Using PyPDF, documents are read, cleaned, and split into sections. FAISS converts documented embeddings into a searchable index, enabling efficient retrieval of

relevant information.

To assess the coherence, consistency, fluency, and relevance of generated summaries, a LLM-based evaluation approach was adopted, inspired by the G-EVAL framework, exploiting again ChatGPT.

4.1.3 Operations Flow

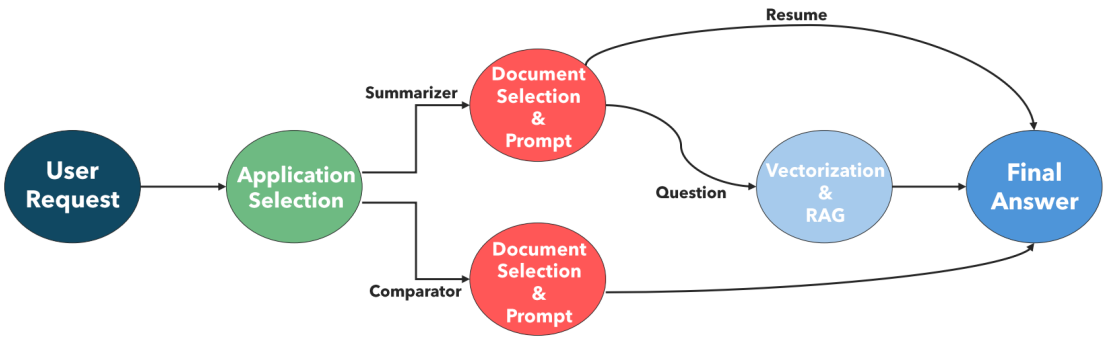


Figure 4.3: A schematization of the operations flow.

In the flowchart of Figure 4.3, it is possible to distinguish all the operations required to process the user's request and generate the chatbot's final response.

The process consists of four main steps:

- **User Request**: The user accesses the chatbot from the home page and selects one of the two available utilities, Summarizer or Comparator, through the left sidebar;
- **Application Selection**: After choosing the desired functionality, the user selects the product (or products) he is interested in summarizing or comparing;
- **Document Selection and Prompt**: In the back-end, the system retrieves the relevant documents from the dataset and constructs an appropriate prompt to send to the LLM. If the Comparator is selected, the model computes the resumes and compares products. Otherwise, if the Summarizer is selected, two different scenarios arise: if the user requests a summary, the model generates it directly; if the user asks a specific question about a product, the system additionally performs a Vectorization and RAG phase, retrieving the most relevant information to ensure an accurate response;
- **Final Answer**: Once the model generates the summary, it formats the response appropriately and displays it in the chatbot interface.

4.1.4 Summarization Pipeline

The core feature developed in this thesis is the ability to summarize multiple financial documents using AI tools. Figure 4.4 illustrates the pipeline from document selection to final summary generation:

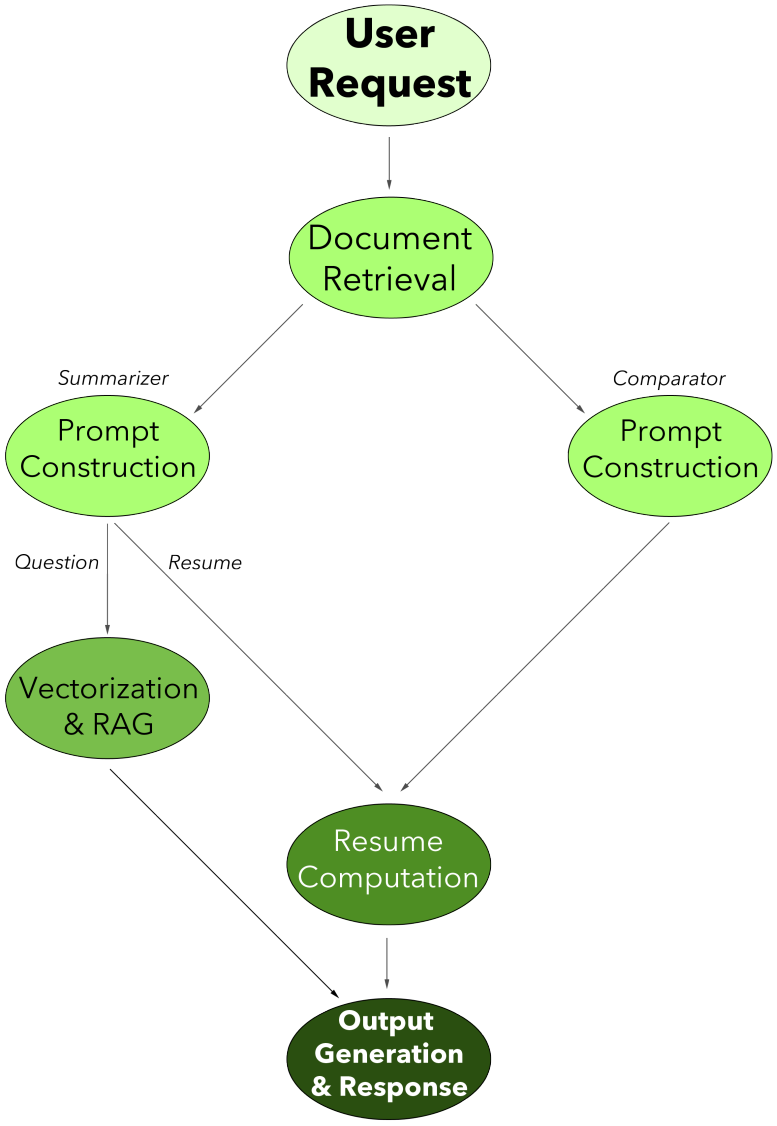


Figure 4.4: A schematization of the pipeline followed to generate the summary.

- **User Request**: The process begins when the user interacts with the chatbot to request the summary of some documents. Rather than selecting specific

documents, the user chooses between summarizing the details of a financial product or comparing two different products;

- **Document Retrieval**: The chatbot, through its back-end Python code, accesses the financial dataset and retrieves the relevant documents needed to generate the response;
- **Prompt Construction**: Based on the user's request, the process diverges into two paths: summarization or comparison. In both cases, a predefined Map-Reduce LangChain prompt is loaded from a JSON file containing all necessary prompts. The system then enriches this prompt with the retrieved documents to generate a meaningful summary;
- **Vectorization and RAG**: If the user chooses the Summarizer and asks a specific question about a product, an additional Vectorization and RAG step is triggered. The text is converted into numerical embeddings to facilitate the retrieval of relevant information. FAISS, a common library for similarity search, indexes and retrieves the k most relevant documents to ensure LLM has the necessary context before generating a response;
- **Summary Computation**: Once the prompt is finalized, LangChain processes it using a Map-Reduce approach. The model first generates individual summaries for document sections (*Map phase*), and then combines them into a final cohesive summary (*Reduce phase*);
- **Output Generation and Response**: After computing the summary (or performing RAG-based retrieval if needed), the chatbot formats and presents the final response to the user.

4.1.5 User Interaction and UI Design

The application's front-end is developed using Streamlit, providing a clean and interactive user experience. The interface consists of three main sections: Home, Summarizer, and Comparator.

Home Page

Upon launching the application, the user is directed to the Home page, which serves as an introductory guide (Figure 4.5). This page contains:

- A **welcome message** explaining the purpose of the application;
- A **quick guide** outlining the functionalities of the Summarizer and Comparator;

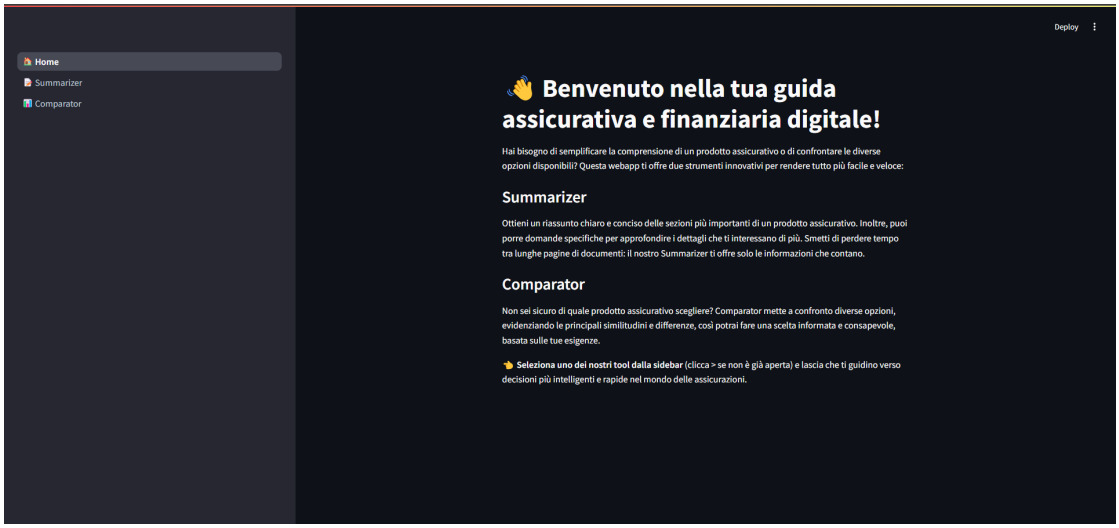


Figure 4.5: The chatbot Home Page.

- A **sidebar menu** on the left that allows users to navigate between the sections.

This landing page ensures that users quickly understand the chatbot's capabilities before proceeding to interact with it.

Summarizer Section

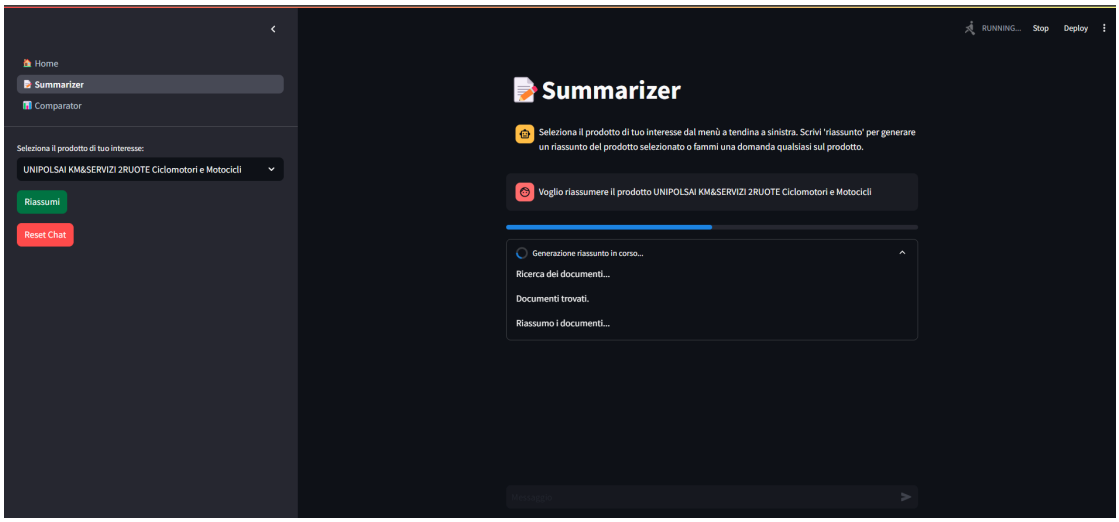


Figure 4.6: Summarizer section with a selected product.

The Summarizer allows users to generate concise summaries of financial products

(Figure 4.6).

User Interaction:

- A drop-down menu in the left sidebar enables users to select a financial product;
- Clicking the "Riassumi" (Summarize) green button starts the summarization process;
- Clicking the "Reset Chat" red button clears the conversation history.

Enhanced Summarization with RAG:

If the user wants more detailed information about a product, they can ask a specific question. In this case, the chatbot:

- Activates the RAG process;
- Retrieves the most relevant sections from the dataset;
- When possible, highlights the source pages where the information was found.

This process is illustrated in Figures 4.7 and 4.8, showing the chatbot retrieving relevant policy sections in response to a user query.

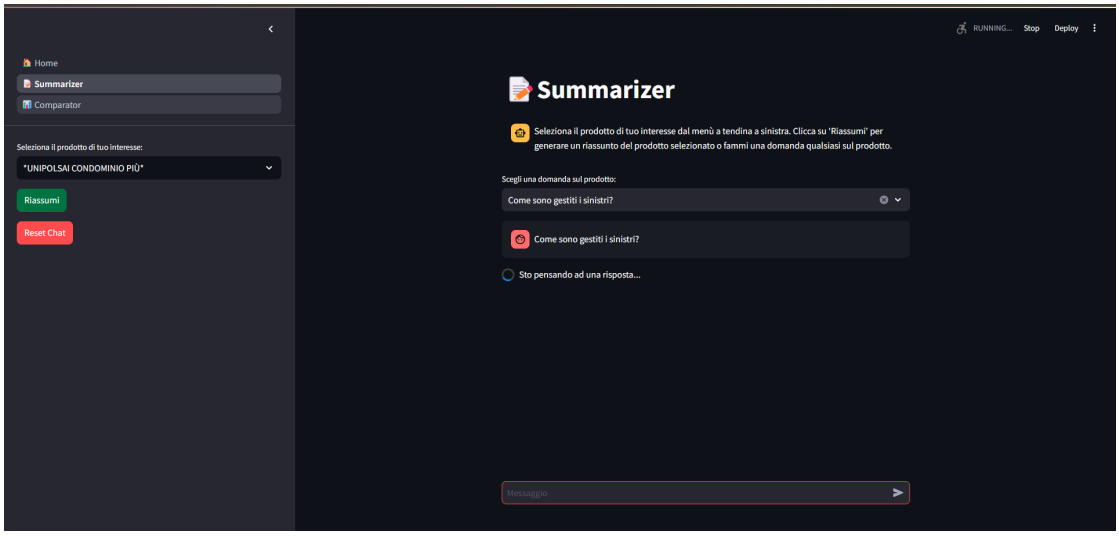


Figure 4.7: User asking a product-related question in the Summarizer section.

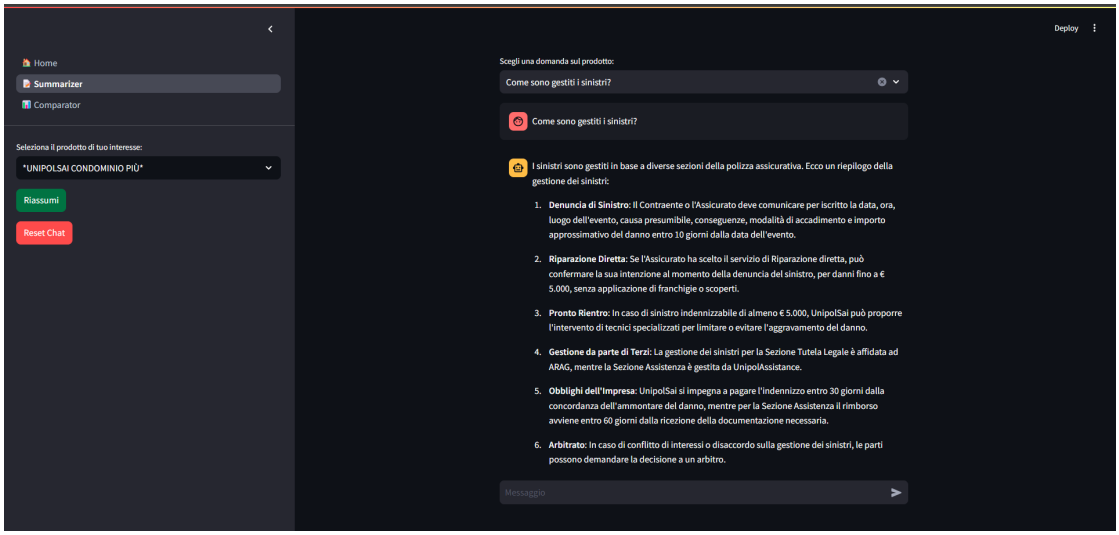


Figure 4.8: Chatbot providing an answer using RAG, referencing relevant policy sections.

Comparator Section

The Comparator allows users to compare two financial products side by side (Figure 4.9).

User Interaction:

- A drop-down menu enables the selection of two products for comparison;
- A radio button allows users to choose the type of document they want to compare.
- Clicking the "Confronta" (Compare) green button starts the comparison process;
- Clicking the "Reset Chat" red button clears the conversation history.

The chatbot processes the selected documents and generates a structured comparison, helping users make informed decisions.

Seamless Navigation and Conversation Persistence

One of the key features of the application is that users can switch between the Summarizer and Comparator sections without losing their conversation history. This allows for a continuous and fluid experience, enabling users to explore different functionalities without restarting interactions.

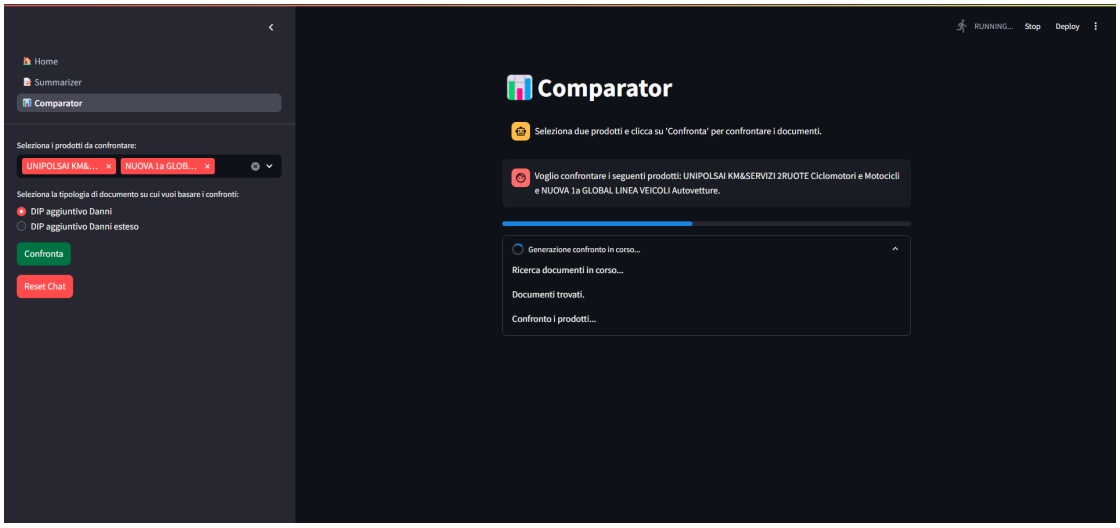


Figure 4.9: Comparator section where two products are selected for comparison.

4.2 Implementation Details

This section presents the implementation process for the final solution proposed in this thesis. It details the practical steps taken to develop each core component. The focus will be on data handling, the summarization system, and the Retrieval-Augmented Generation module, which will ensure that the chatbot can generate good-quality summaries and retrieve precise information from the dataset.

The section will first examine dataset preparation, preprocessing techniques, and how documents were managed to ensure optimal model performance. Next, it will cover the summarization pipeline, including the development of Map-Reduce prompts and optimization strategies. The integration of RAG will also be explored, describing how document knowledge retrieval improves answer accuracy.

Finally, the section will analyze the user interaction system, focusing on the chatbot interface and interaction flow, along with prompt engineering techniques designed to enhance response quality based on evaluation results.

4.2.1 Preprocessing and Data Management

The dataset used in this thesis for the chatbot solution consists of financial documents extracted from the Unipol website [1]. This dataset did not exist before this research, and it was created by collecting PDF documents related to various insurance products. To construct it, a Python-based web scraping script was

[1] unipol.it

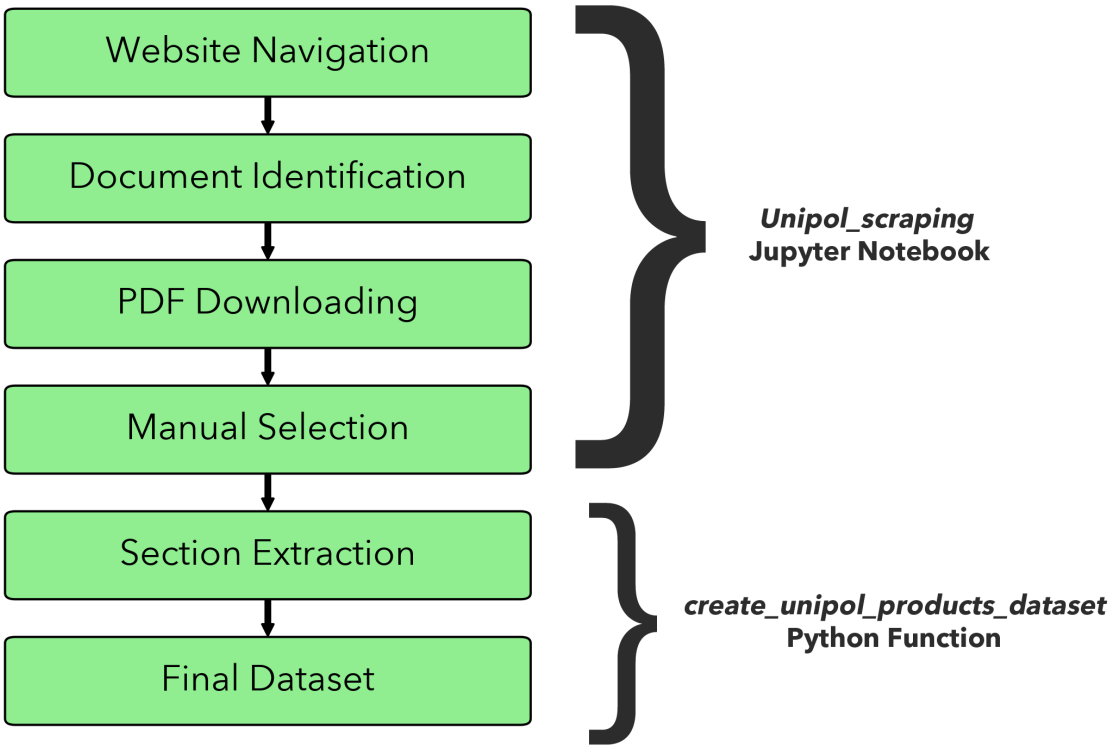


Figure 4.10: A schematization of the dataset processing flow. On the right, the details about the code structures that contain them.

developed to automate the retrieval of these documents. The script systematically scanned the website, identifying and downloading policy documents associated with selected insurance products.

The extraction process involved two main phases:

- **Navigation**: The script explored product-specific pages containing links to relevant policy documents;
- **Automated Document Retrieval**: All linked PDF files associated with each insurance product were identified and downloaded.

Following the web scraping phase, where 16 insurance products were selected, a manual preprocessing step was performed. This involved:

- **Documents Selection**: The retrieved documents were compared to determine the most relevant file for each product. Ultimately, one document per product was chosen, ensuring consistency across the dataset.
- **Content Structuring**: Although the selected documents shared a common structure, variation existed due to the specific nature of each insurance product.

To simulate a Multi-Document Summarization setting, the chosen documents were segmented into distinct sections, effectively generating multiple sub-documents per product. This approach enabled a more granular and structured evaluation of the summarization pipeline.

Figure 4.10 provides an overview of the dataset processing workflow, highlighting that the initial steps are performed within a Jupyter Notebook (*Unipol_scraping*), while the final steps are executed inside a dedicated Python function (*create_unipol_products_dataset*).

4.2.2 Summarization Pipeline Implementation

Listing 4.1: Summarization Pipeline Pseudo-Code

```
# Step 0    - Load OpenAI API key
openai_api_key <- load_openai_key(key_path)

# Step 1    - Identify the user's request
if user_input == "Summarization":
    summarization_type <- "Summarizer"
    selected_product <- get_selected_product()  # Retrieve
   the chosen product

else if user_input == "Comparison":
    summarization_type <- "Comparator"
    selected_products <- get_selected_products()  # Retrieve
    both selected products

else if user_input == "Question":
    summarization_type <- "RAG"
    user_question <- get_user_question()  # Get the user's
   query

# Step 2 - Select the appropriate documents based on user
   input
if summarization_type == "Summarizer":
    documents <- retrieve_documents(selected_product)  #
   Retrieve documents for the selected product

else if summarization_type == "Comparator":
    documents_1 <- retrieve_documents(selected_products[0])
    # Retrieve documents for first product
    documents_2 <- retrieve_documents(selected_products[1])
    # Retrieve documents for second product

```

```
else if summarization_type == "RAG":
    relevant_documents <- vector_search(user_question)  # Perform similarity search to find relevant docs

# Step 3 - Construct the appropriate prompt
prompt_template <- load_prompt(summarization_type) # Load the correct prompt template
if summarization_type == "Summarizer":
    prompt_text <- construct_summarization_prompt(documents)  # Generate summarization prompt

else if summarization_type == "Comparator":
    prompt_text <- construct_comparison_prompt(documents_1, documents_2)  # Generate comparison prompt

else if summarization_type == "RAG":
    prompt_text <- construct_rag_prompt(user_question, relevant_documents)  # Generate RAG-based prompt

# Step 4 - Generate the answer using the LLM
if summarization_type == "RAG"
    answer  <- generate_answer(prompt_text, model="ChatGPT-4o Mini")  # Send prompt to LLM and generate answer
else
    answer <- generate_summary(prompt_text, technique="map_reduce", model="ChatGPT-4o Mini")  # Send prompt to LLM and generate summary

# Step 5 - Post-process the output and return the response
formatted_answer <- postprocess(answer)  # Format the text to improve readability
display_in_chatbot(formatted_answer)  # Send the final response to the chatbot interface
```

The primary objective of this research was to demonstrate that, leveraging existing knowledge on news summarization, it is possible to adapt the methodology to the insurance domain while maintaining high-quality results. Unlike news summarization, the insurance domain presents additional challenges, primarily due to the critical importance of numerical values, such as coverage limits, deductibles, and reimbursement amounts. In contrast, numerical values in news summaries are often omitted without significantly affecting the information transmitted.

As previously discussed in Section 4.1, the technologies used to implement the summarization pipeline include Python, LangChain, and OpenAI APIs. These tools enable efficient processing, retrieval, and summarization of insurance-related

documents while ensuring coherence, accuracy, and usability in a chatbot interface.

Listing 4.1 provides pseudo-code summarizing the key steps required to generate a summary based on user input. The summarization pipeline follows a structured sequence of operations:

- **Step 0 - API Key Inizialization**: Before executing any operations, the OpenAI API key must be loaded to grant access to ChatGPT-4o Mini and LangChain APIs;
- **Step 1 - User Request Handling**: The summarization process begins with a user request, where the user selects one of the available applications:
  - *Summarization*: Generates a summary for a single insurance product;
  - *Comparison*: Compares the summaries of two distinct products;
  - *Question Answering*: Extracts information from documents in response to a specific user query.

  While question answering is treated as a separate option in this pipeline, it is actually integrated into the Summarizer application in the chatbot's final implementation. This means that a user can both request a summary and ask a specific question about a product;
- **Step 2 - Document Retrieval**: After selecting an application, the user specifies the insurance product(s) of interest. Based on the choice:
  - For Summarization: The system retrieves all relevant documents for the selected product;
  - For Comparison: The system retrieves documents for both selected products;
  - For Question Answering: A vector-based similarity search is performed on the documents to identify the most relevant sections for answering the user's query.
- **Step 3 - Prompt Construction**: Once the appropriate documents are retrieved, a LangChain prompt is generated:
  - The system selects the correct prompt template based on the application type;
  - The retrieved documents are incorporated into the prompt following the Map-Reduce approach:
    - The Map phase processes the individual documents, generating partial summaries;

    - The Reduce phase combines these partial summaries into a final, coherent output.
  - If the user posed a question instead of requesting a summary, a specialized RAG prompt is generated to retrieve an answer.

  The next subsection will discuss Prompt optimization and structuring in more detail;

- **Step 4 - Summary Generation via LLM**: The final prompt is submitted to LangChain, initiating the actual summarization process. The Chain-of-Event methodology is applied, ensuring that the output maintains logical coherence across multiple documents.

  If RAG-based QA was selected, the system provides the answer along with references to the document pages where the relevant information was found;

- **Step 5 - Post-processing and Chatbot Response**: The final summary or answer undergoes post-processing to enhance readability and structure. The formatted response is then delivered through the chatbot interface, ensuring a smooth and user-friendly experience.

4.2.3 Prompt Optimization and Fine-Tuning

The quality of summaries generated by Large Language Models is heavily influenced by the structure and clarity of the prompts. Consequently, prompt optimization represented a key phase in this research, particularly due to the complexity of the insurance domain, where it is essential to preserve financial and contractual details in a precise and coherent manner.

Unlike the experiments conducted on the WCEP news dataset, where brief and less structured outputs were acceptable, the insurance domain required more detailed and controlled outputs. The initial prompts were too generic and lacked precise instructions, which made them unsuitable for effectively summarizing insurance-related documents. This section outlines the iterative refinement process, providing examples of how the prompts were redesigned to enhance quality and coherence.

Initial Prompt Structure

The initial prompts - both in the Stuff and Map-Reduce configurations - were adapted from LangChain examples originally designed for general-purpose summarization. These were tested on insurance documents with minimal modifications, mainly to insert the texts to summarize.

The two initial versions were as follows:

Stuff Prompt

Write a concise summary of the following documents related to an insurance product:
{text}
Documents are separated by the text '_NEW_DOCUMENT_'
Concise Summary:

Map-Reduce Prompt

Map:
The following is a set of documents related to an insurance product:
Input
Based on this list of docs, please identify the main themes.
Helpful Answer:
Reduce:
The following is a set of summaries related to an insurance product:
Input
Take these and distill them into a final, consolidated summary of the main themes.
Helpful Answer:

These prompts were intentionally generic and provided only limited contextual information. No explicit guidance was given regarding tone, structure, or the inclusion of numerical values, which are critical in the insurance domain.

As a result, several issues were observed:

- Summaries lacked a logical order or clear structure;
- Repetitive or overly generic sentences were frequently produced;
- Important financial values (e.g., reimbursement limits, deductibles) were often omitted;
- The model frequently described the documents themselves rather than the insurance products.

Prompt Refinement Process

To address these limitations, a systematic refinement process was initiated. The goal was to provide more structured guidance to the model and ensure accurate, informative outputs.

The refined prompts were inspired by the Chain-of-Events approach introduced by Bao et al. (2024)[23], which structures the summarization into ordered reasoning steps. The new version aimed to:

- Enforce a logical and coherent structure in the summary;
- Eliminate references to the document itself;
- Emphasize the inclusion of monetary details and policy conditions;
- Consistently center the insurance product as the main subject of the summary.

Since preliminary experiments showed that the Map-Reduce technique outperformed the Stuff method, all the subsequent refinements focused exclusively on the Map-Reduce configuration.

Refined Map Prompt

Map Prompt

You will receive documents related to an insurance product:
{Input}
Read each document carefully and identify its type. Summarize the critical information, focusing on the insurance product itself rather than the document. Always use the product as the subject of the summary, describing its features, coverage, and terms, instead of mentioning "the document" or its contents. Integrate specific coverage amounts, reimbursement limits, deductibles, and other financial details where applicable. It is crucial to include all monetary amounts mentioned in the document, such as limits on reimbursement (e.g., €600 per incident) or deductibles (e.g., €150 deductible for water damage). Avoid using sentences such as "The document contains" or "This document outlines", instead create a summary that flows naturally where the product, not the document, is the thing the sentences mainly focus on. Each document's content should be merged into a continuous summary as if the information were part of a single coherent narrative.
Please note: there is a separate document that contains information on claims presentation and dispute resolution procedures. This document is part of the Additional pre-contractual document and should be summarized accordingly.
The document may be one of the following:
1. Pre-contractual document: ...
2. Additional pre-contractual document: ...
3. Glossary: ...
4. Contractual rules: ...
5. Claims liquidation rules: ...

6. Legal norms: ...
7. Product details documents: ...
Ensure that each summary uses the insurance product as the subject, and avoid mentioning "the document" or describing the document itself. Focus on presenting the product's features and terms. Make the summary flow smooth, integrating the financial details naturally, and including specific monetary figures when present in the document.

This version provided explicit structure, document classification, and instruction, maintaining focus on the product itself. It also stressed the importance of financial details, aiming to improve lexical quality and relevance. The descriptions of the possible documents in the example are not reported for space reasons.

Refined Reduce Prompt

Reduce Prompt

The following are summaries of various documents related to an insurance product:
{Input}
Your task is to combine these into a single, cohesive summary that focuses on the insurance product itself. The final summary should always use the insurance product as the subject of the summary, avoiding references to "the document" or its content.
Important: The final summary should be between 300 and 500 words. Focus on conciseness, ensuring that the key details are presented without overloading the text with unnecessary or repetitive information.
Ensure that all coverage amounts, reimbursement limits, deductibles, and other financial figures are clearly integrated into the final text. These monetary details are essential for understanding the scope of the insurance product and should not be omitted.
The final summary should include:
- A clear description of the product's features, benefits, and coverage.
- Specific coverage amounts, reimbursement limits, or deductibles for damages or claims (e.g., €600 per incident, €1,000 for electrical damage).
- Key contractual rules, including the duration of the policy, renewal terms, and any conditions for cancellation or termination.
- The claims process and legal norms, summarized concisely but without overloading the text with excessive details.
Avoid repeating phrases like "The document ..." and focus on describing the insurance product itself. The summary should flow naturally, providing a

comprehensive yet concise overview of the product, with a clear focus on its features and terms. Ensure that unnecessary repetition or excessive detail is avoided to maintain brevity.
Here is an example of a well-structured summary for a different insurance product:
—
...
—
Ensure that the final summary is written in Italian and remains within the word limit of 300 to 500 words.

This section also guides the model through the summarization process by explicitly defining the expected output, including an example of a well-structured summary (omitted here for brevity).

These improvements significantly enhanced output quality, producing more informative and well-structured summaries. However, two remaining challenges prompted further refinement:

- The initial tests used GPT-3.5, which showed limitations in complex summarization tasks;
- The prompts were written in English, while the input documents were in Italian, resulting in language mismatch and translation artifacts.

To address these, the final versions were translated and redesigned in Italian and tested using the improved GPT-4o Mini model.

Final Prompt Implementations

The final prompt suite consisted of three versions, MR1, MR2, and MR3, all based on the Map-Reduce strategy but with different styles and levels of structure. These prompts were developed in Italian and targeted at the Unipol insurance dataset. Each prompt shares the same logic:

- The **Map phase** is responsible for critical information extraction: it extracts and resumes the specific information of the product, producing a single-line resume for each document, also including financial and coverage details;
- The **Reduce phase** performs abstraction and generalization of the single-line resumes - to analyze them and extract common characteristics of the documents and generalize key details (again favoring numerical ones) -, the common element statistics - to select the details that are recurrent in almost all the documents -, and the final resume generation - to produce a unified summary -.

Prompt	Technique	Details
Unipol MR1	Map-Reduce	Conversational style with the required actions described in a more narrative way
Unipol MR2	Map-Reduce	Reasoning Steps presented as structured numerical points
Unipol MR3	Map-Reduce	Hybrid approach, with conversational style + step-by-step bullet points structure

Table 4.1: Unipol final prompts comparison summary

The full example of the MR2 prompt is provided in this chapter, while the others are summarized in Table 4.1.

Unipol MR2 Map Prompt

Ora sei un generatore di riassunti multi-documento. Il tuo <Input> è un insieme di testi assicurativi che riguardano un prodotto assicurativo specifico. Dovrai estrarre le informazioni principali e specifiche relative a ciascun documento, concentrandoti su dettagli finanziari e di copertura.
<Input>
Input
</Input>
<Estrazione delle informazioni critiche>
Per ciascun documento nel <Input>, estrai e riassumi le informazioni critiche relative al prodotto assicurativo. In particolare, concentrati sui seguenti aspetti chiave e produci una singola linea per ciascun documento, includendo le informazioni finanziarie e di copertura.
1. Descrizione del prodotto e tipi di copertura (es. decesso, invalidità permanente, danni materiali, ecc.).
2. Limiti di copertura e importi di rimborso specifici (es. €600 per incidente, €1.000 per danni elettrici).
3. Esclusioni (cosa non è coperto dalla polizza).
4. Franchigie e altri dettagli finanziari (es. €150 di franchigia per danni da acqua).
5. Opzioni di copertura aggiuntiva, inclusi eventuali costi aggiuntivi.
6. Durata del contratto, termini di rinnovo e pagamento.
7. Processo di gestione dei sinistri e condizioni per il rimborso.
Questo è un esempio di estrazione delle informazioni critiche:
<Esempio di estrazione delle informazioni critiche>

...
</Esempio di estrazione delle informazioni critiche>
Seguendo l'esempio sopra, estrai le informazioni critiche per ciascun documento assicurativo del <Input>, producendo una singola riga per ciascun documento.

Unipol MR2 Reduce Prompt

Il tuo <Input> è un insieme di riassunti specifici estratti da documenti assicurativi. Il tuo compito è analizzare questi riassunti, astrarre le informazioni principali e combinarle in una sintesi coesa.
<Input>
Input
</Input>
<Astrazione delle informazioni e generalizzazione>
1. Analizza ciascun riassunto estratto nella fase MAP e astrai le caratteristiche comuni del prodotto assicurativo.
2. Generalizza i dettagli chiave relativi a:
- Tipi di copertura e limiti di rimborso.
- Esclusioni principali.
- Franchigie e importi finanziari specifici.
- Durata del contratto e condizioni di rinnovo.
3. Raccogli i dettagli specifici e condensali in una singola versione astratta del prodotto, omettendo dettagli superflui o ridondanti.
Questo è un esempio di astrazione e generalizzazione:
<Esempio di astrazione e generalizzazione>
...
</Esempio di astrazione e generalizzazione>
<Statistica degli elementi comuni>
Dopo aver generalizzato le informazioni, trova gli elementi che ricorrono in più documenti e seleziona quelli che coprono la maggior parte dei testi. Questi elementi saranno centrali nel riassunto finale.
Questo è un esempio di statistica degli elementi comuni:
<Esempio di statistica degli elementi comuni>
...
</Esempio di statistica degli elementi comuni>
<Generazione del riassunto finale>
Infine, integra gli elementi selezionati e generalizzati per formare un riassunto finale coeso e ordinato cronologicamente o per importanza.
<Esempio di riassunto finale:>
...

</Esempio di riassunto finale>
Assicurati che il riassunto finale sia compreso tra 300 e 500 parole.

These prompts were integrated into the chatbot prototype, and the next chapter presents the experimental results that led to the selection of the optimal prompt.

Rather than undergoing a separate refinement process, the comparison prompts were derived by adapting the most effective summarization prompt to suit the dual-input nature of the comparator tool. The generated prompts for this section are two, which are different because of the possibility of resuming different sections of the documents.

Also in this case, an example of these prompts is presented in this section.

Unipol Comparator Map Prompt

Ora sei un generatore di riassunti comparativi per prodotti assicurativi. Il tuo <Input> consiste in una serie di documenti assicurativi. Dovrai estrarre le caratteristiche chiave di ciascun prodotto.
<Input>
Input
</Input>
<Estrazione delle informazioni chiave>
Per ogni testo nel <Input>, estrai i dettagli chiave relativi al prodotto assicurativo, concentrandoti su:
- Il tipo di assicurazione e il suo scopo (es. assicurazione sulla vita, assicurazione per moto)
- Tipi di copertura (es. copertura per decesso, copertura contro danni materiali per moto)
- Limiti di copertura (importi specifici di rimborso o percentuali)
- Esclusioni (ciò che non è esplicitamente coperto)
- Franchigie e scoperti (dettagli finanziari come franchigie o importi minimi non coperti)
- Coperture opzionali (es. protezione aggiuntiva contro infortuni, estensione della copertura per accessori della moto)
- Processo di reclamo e condizioni per il rimborso
</Estrazione delle informazioni chiave>
Questo è un esempio di estrazione delle informazioni chiave: <Esempio di estrazione delle informazioni chiave>
...
</Esempio di estrazione delle informazioni chiave>
Seguendo l'esempio sopra, produci l'estrazione delle informazioni chiave per

il tuo <Input>.

Unipol Comparator Reduce Prompt

Il tuo <Input> è un insieme di informazioni chiave estratte da diversi documenti assicurativi. Il tuo compito è confrontare questi dettagli e identificare gli elementi comuni e quelli differenti tra i prodotti assicurativi.
<Input>
Input
</Input>
<Confronto degli elementi comuni e different>
1.Elenca in una frase quali sono i prodotti che hai analizzato e descrivi brevemente cosa trattano.
2. Analizza i dettagli chiave di ciascun documento e confrontali in base ai seguenti aspetti:
- Tipi di copertura
- Limiti di copertura e importi di rimborso
- Esclusioni
- Franchigie e scoperti
- Coperture opzionali
- Processo di reclamo
3. Crea due elenchi: uno per gli elementi comuni e uno per gli elementi differenti. Nell'elenco degli elementi comuni, includi le caratteristiche presenti in tutti i documenti. Nell'elenco degli elementi differenti, evidenzia le discrepanze e le differenze tra i prodotti.
</Confronto degli elementi comuni e differenti>
Questo è un esempio di confront:
<Esempio di confronto degli elementi comuni e differenti>
...
</Esempio di confronto degli elementi comuni e differenti>
Produci il <Confronto degli elementi comuni e differenti>, elencando gli elementi comuni e differenti dei prodotti assicurativi.

4.2.4 Retrieval-Augmented Generation Integration

While the primary focus of this thesis is to demonstrate the feasibility of applying Multi-Document Summarization techniques to the insurance domain, the final solution is implemented as an interactive chatbot. To enrich its functionality and support a more dynamic user experience, the system was also extended to support question-answering capabilities, allowing users to ask specific questions about each

available product.

Inside the Summarizer section, once a product is selected, the user can submit a free-form question about it. However, general LLM-based responses are often not enough in these cases, due to some well-known limitations:

- **Outdated knowledge**: LLMs are trained on a fixed corpus and may not include the most recent or domain-specific information;
- **Generic answers**: Without direct access to external sources, the model may provide vague or high-level responses, which are not customized for the specific user requirements;
- **Lack of verifiability**: Generated outputs often cannot point to specific sources, reducing trust in response. They could also contain various inaccuracies since they have no updated and verified knowledge.

To overcome these challenges, the Retrieved-Augmented Generation methodology was integrated into the chatbot architecture. This approach supplements the language model with a retrieval mechanism, allowing it to ground its answers in documents. The resulting system ensures that answers are not only contextually appropriate but also traceable to the source material, increasing both the accuracy and trustworthiness of the responses.

Summarizing the concepts already presented in the Design Strategy section(4.1), the RAG process implemented in this project follows the pipeline:

User Input (Free Question) → Document Retrieval (Top-K) → Answer Generation using retrieved context

The user is first prompted to choose an insurance product. Once the user submits a specific question, the system conducts a similarity search on the product's associated documents. It then takes out the most relevant extracts and feeds them to the language model, along with the original question.

Again, the LLM employed in the RAG system is ChatGPT-4o Mini, integrated with LangChain APIs. The prompt used for this process, designed in Italian to align with the language of the documents, is presented below:

RAG Prompt

Di seguito ti fornisco più estratti di un documento assicurativo. Rispondi alla domanda basandoti esclusivamente sugli estratti forniti. Se la risposta non è presente chiaramente, segnala che le informazioni non sono sufficienti.
Estratti:
{context}
Domanda:
{question}

Risposta:

It is important to note that, unlike the summarization prompts, which underwent a detailed optimization and fine-tuning process, the RAG prompt was designed using a more straightforward approach. The emphasis was placed on clarity, actuality, and alignment with the structure of the source material. While future work could explore more advanced RAG tuning methods or the use of different retrievers, the current solution uses OpenAI Embeddings and FAISS for document vectorization, tools that proved to be effective, lightweight, and easy to integrate within the project scope.

The following pseudo-code illustrates the implementation of the RAG workflow in the chatbot system:

Listing 4.2: RAG-Based Question Answering Pseudo-Code

```
# Step 1 - Load the documents for the selected product
documents <- load_product_documents(product_name)

# Step 2 - Initialize the embedding model (OpenAI)
embedding_model <- load_openai_embeddings()

# Step 3 - Vectorize the documents using FAISS and the
    embedding model
vector_store <- vectorize_with_faiss(documents,
    embedding_model)

# Step 4 - Load RAG prompt template and build the prompt
rag_prompt_template <- load_prompt("RAG")
relevant_chunks <- vector_store.top_k(user_question, k=3)  #
     Retrieve top relevant chunks
prompt_text <- construct_rag_prompt(user_question, context=
    relevant_chunks)

# Step 5 - Generate the answer using the RAG LLM chain
answer <- generate_answer(prompt_text, model="ChatGPT-4o
    Mini")

# Step 6 - Post-process: check if the answer was found in
    the provided documents
if answer_found_in_documents(answer, vector_store,
    user_question):
     pages <- extract_pages(vector_store, user_question)  #
    Identify source pages
```

```
    final_answer <- concatenate(answer, " (Answer found in
   pages: ", pages, ")")
else:
    final_answer <- "Sorry, the information is not available
    in the provided documents."

# Step 7 - Display the result in the chatbot interface
display_in_chatbot(final_answer)
```

As shown in Listing 4.2, the system performs a sequence of steps that integrate retrieval and generation. Specifically:

- **Steps 1-3** involve loading and embedding the documents and building a searchable vector store using FAISS;
- **Step 4** retrieves the top 3 relevant extracts based on a similarity search and constructs a prompt that guides the model to generate an answer based solely on these extracts;
- **Steps 5-6** involve generating the answer and checking if it is grounded in the documents. If the relevant information is present, the output includes references to the source pages; otherwise, the user is notified that the information could not be found;
- **Step 7** renders the final response within the chatbot interface.

This integration allows the chatbot to move beyond a general-purpose summarization and support more personalized, query-driven interactions. RAG helps to enhance the flexibility of the developed system.

4.2.5 User Interface and Interaction System

Although the user interface is presented at the end of this chapter, it represents a fundamental component of the entire solution. It is responsible for receiving user inputs, managing the interaction flow, and displaying the system responses in a clear and user-friendly manner.

The interface was implemented using Streamlit, a Python framework that allows for the rapid development of interactive web applications. This choice made it possible to quickly integrate the functionalities developed in the backend and offer the user a smooth interaction experience. The web application provides access to the three main functionalities described throughout the thesis: *Summarization*, *Comparison*, and *Question Answering* (RAG). The architectural aspects of these features were already presented in previous sections; here, the focus is instead on their practical implementation within the graphical interface.

Pages Handling and Navigation

The application was designed using the multi-page functionality of Streamlit, which enables the division of the interface into three separate pages:

- **Home**: A landing page that introduces the user to the application and allows them to navigate to the desired tool;
- **Summarizer**: A page that enables the user to select a product and either generate a summary or ask a specific question about it;
- **Comparator**: A page that allows the comparison of two products by analyzing similarities and differences in their documentation.

Figure 4.11 illustrates the navigation structure of the chatbot interface, together with screenshots of the pages.

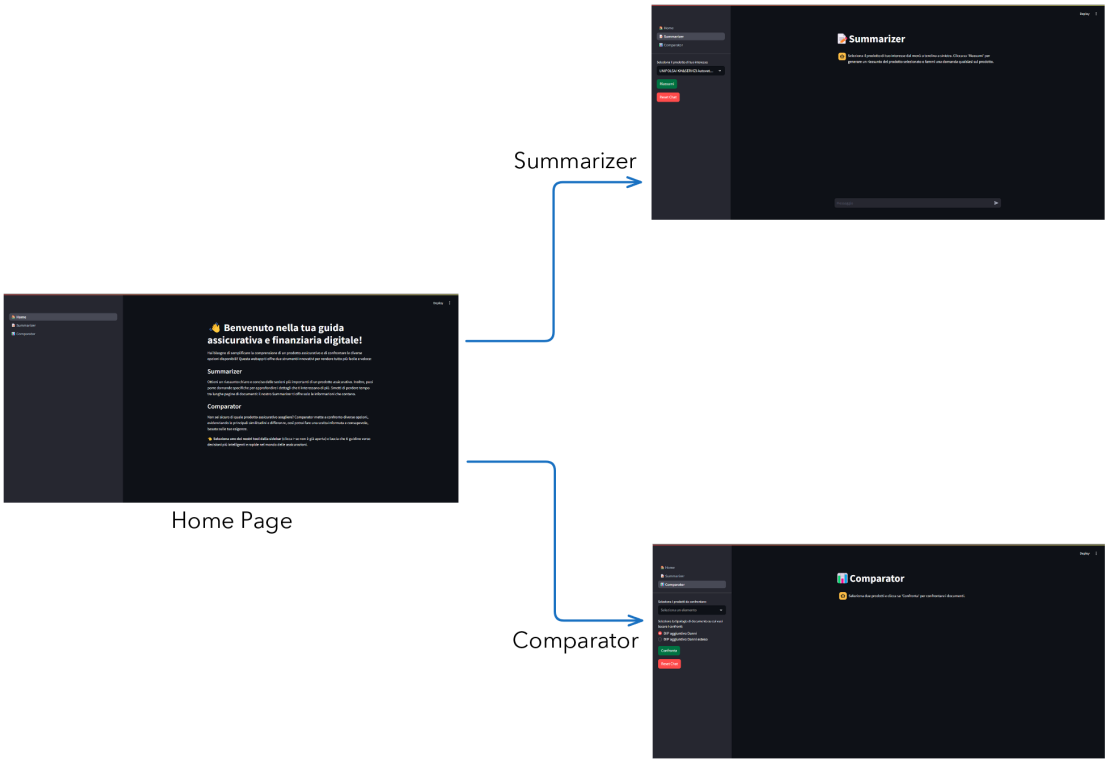


Figure 4.11: Navigation structure of the chatbot interface.

Within the Summarizer page, two interaction modes are managed:

- **Summarization request**: Upon clicking the "Riassumi" button, the selected product documents are summarized using the multi-document pipeline;

- **Question answering**: If the user submits a natural language question, the system utilizes the RAG mechanism to retrieve and answer based on the document contents.

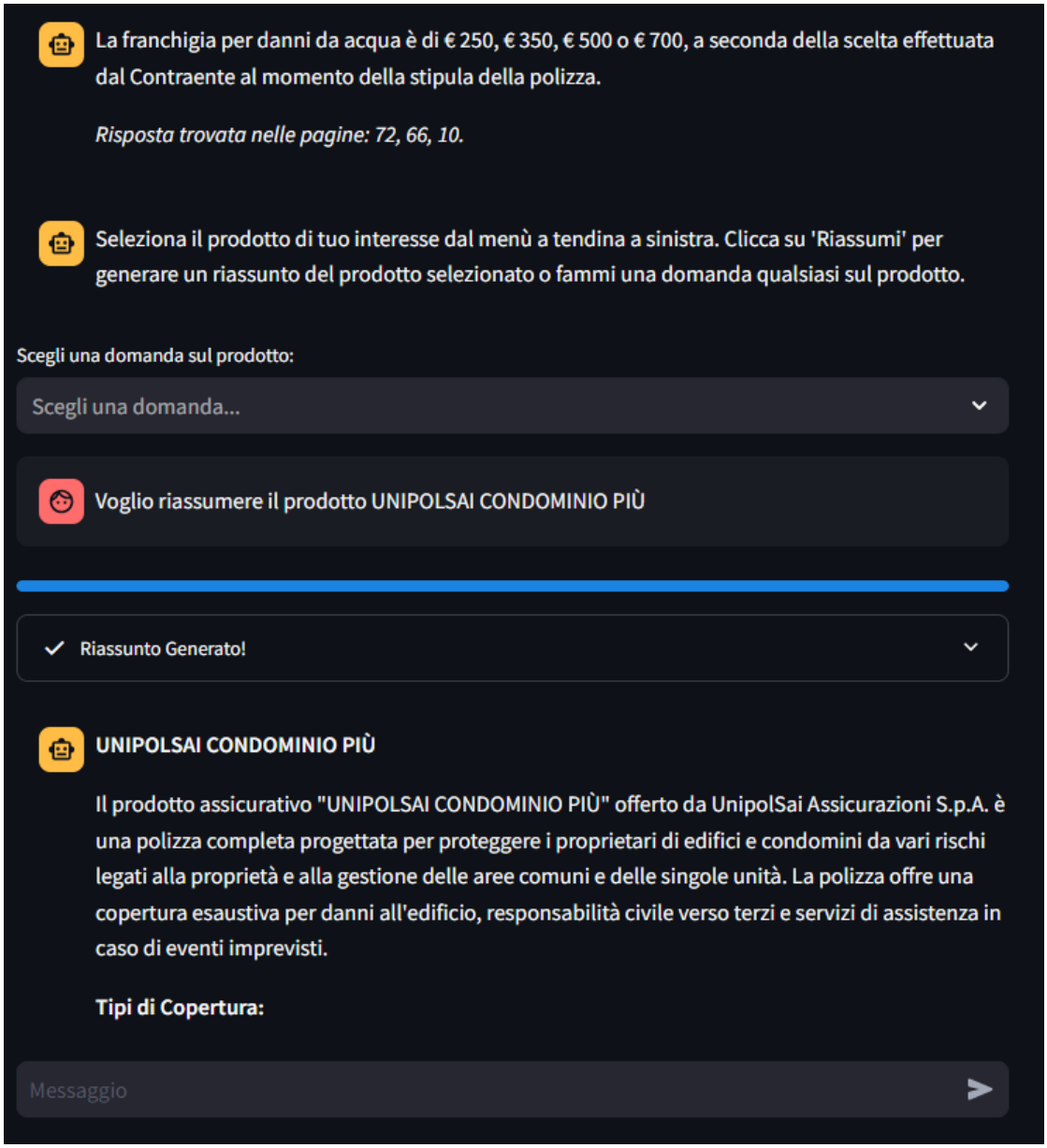


Figure 4.12: Example of chat interface during an active session.

User Input and Conversation Handling

User interaction is handled with care, ensuring clarity and ease of use. Product selection is performed through dropdown menus or multi-select forms, and user messages are entered using the chat input field. Custom and visible buttons are used to initiate the main actions of the application.

The chat is managed using the *streamlit.session_state* object, which stores a history of the conversation. Each message is rendered using the *streamlit.chat_message* function, visually distinguishing between the user and the virtual assistant. The session history is persistent across sections and can be reset via a dedicated button.

4.2.6 Backend and LLM Integration

The interface communicates with the backend components described earlier in the thesis, specifically:

- An LLMChain object is used to generate responses, with pre-loaded prompts, used with the OpenAI model (ChatGPT-4o Mini);
- Documents associated with the selected insurance products are dynamically loaded using the PyPDFLoader library;
- For question answering, documents are vectorized in real-time using OpenAI embeddings and the FAISS library, allowing fast and context-aware retrieval.

These operations are executed transparently, maintaining responsiveness and ensuring that the user always interacts with up-to-date information.

Output and Post-Processing

All the generated outputs are rendered as chat messages. In the case of RAG-based answers, an additional sentence is appended to the response to indicate the source pages from which the information was retrieved. If the answer cannot be found in the documents, a fallback message is shown to inform the user.

The chatbot interface is entirely text-based, but attention was paid to formatting and phrasing to ensure a fluent and accessible reading experience.

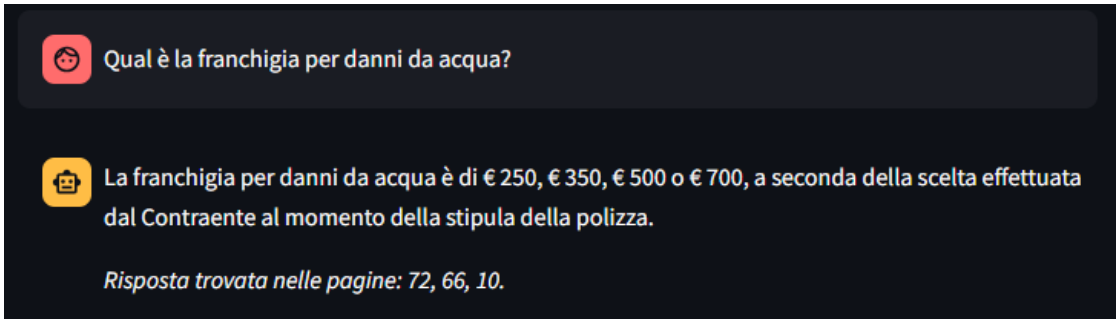


Figure 4.13: Example of RAG powered Q&A.

Chapter 5

Results and Statistics

During the development of this project, several experiments were conducted to assess and refine the summarization pipeline.

The initial step involved reproducing the examples from the LangChain blog [25], specifically applying the Stuff and Map-Reduce base prompts to the WCEP Dataset. This phase aimed to establish a baseline for comparison with the results obtained by modifying the prompts according to the selected Chain-of-Events methodology [23].

Subsequently, the experiments focused on optimizing the prompts to achieve the best possible performance on the WCEP Dataset. During this phase, the evaluation approach shifted from supervised to unsupervised metrics, as the former were deemed inappropriate for this thesis's final objectives and lacked sufficient generalization capabilities.

Once the best results were obtained on the WCEP Dataset, the next objective was to validate the approach on an insurance dataset, specifically created for this thesis. In this phase, three different prompt versions were tested, this time relying exclusively on the unsupervised evaluation metrics to assess performance.

5.1 Experimental Setup

As previously mentioned, the experiments can be divided into two main parts, one that involves the news summarization field and the other one that involves the insurance field.

Multiple setups have been tested, this subsection will explain in detail the datasets and prompts that were utilized.

Dataset	Clusters	Documents	Documents Per Cluster	Domain
WCEP-Full	8158	494151	60.57 (avg)	News
WCEP-1	65	1300	20	News
WCEP-2	1	10	10	News
WCEP-3	4	80	20	News
WCEP-4	20	500	4x10, 4x15, 4x25, 4x35, 4x40	News
Unipol	16	111	6.94 (avg)	Insurance

Table 5.1: Summary of the datasets variants used in the experiments.

5.1.1 Datasets

Two distinct datasets were used in the two main phases of the experiments: the WCEP Dataset and a financial Dataset obtained from the Unipol Website [1], hereafter referred to as the Unipol Dataset.

The WCEP Dataset used is a subset of the original collection, consisting of 494151 documents grouped into 8158 clusters, where each cluster represents a collection of documents sharing a common theme. However, this dataset was not used in its entirety, instead, each experiment was conducted on specific subsets, depending on the test configuration. Table 5.1 summarizes the different subsets used throughout the study.

In contrast, the Unipol Dataset consists of 16 documents, each corresponding to a different insurance product. However, these documents were further segmented into 111 smaller sections, effectively forming 16 clusters and a total of 111 documents. In this case, each cluster corresponds to a single insurance product.

5.1.2 Prompt Engineering

For the experiments, different prompting strategies were applied to both datasets, focusing primarily on two methods: Stuff and Map-Reduce.

Similarly to the dataset selection, the prompts used in the experiments can be categorized into two main groups: one set designed for WCEP Dataset experiments and another for Unipol Dataset experiments. Table 5.2 provides an overview of all prompts tested during the study.

A total of six different prompts were tested on the WCEP Dataset:

- **S1**: Inspired by the LangChain blog [25], this is a basic version of the Stuff methodology, modified only to include the documents within the prompt;

[1] unipol.it

Prompt	Technique	Notes	Dataset
S1	Stuff	LangChain	WCEP
S2	Stuff	CoE Paper	WCEP
MR1	Map-Reduce	LangChain	WCEP
MR1.1	Map-Reduce	CoE Paper + Split	WCEP
MR2	Map-Reduce	CoE Paper	WCEP
MR3	Map-Reduce	CoE Paper + Steps	WCEP
MR1	Map-Reduce	CoE Paper	Unipol
MR2	Map-Reduce	CoE Paper + Steps	Unipol
MR3	Map-Reduce	CoE Paper + Bullet Points	Unipol

Table 5.2: Summary of the prompts used in the experiments.

- **S2**: Based on the Chain-of-Event paper [23], this prompt also follows the Stuff methodology, attempting to replicate the four-step reasoning process;
- **MR1**: Inspired by the LangChain blog [25], this is a basic version of the Map-Reduce methodology, modified only to include the documents;
- **MR1.1**: A variation of MR1, where documents are split into chunks of up to 1000 tokens;
- **MR2**: Inspired by the Chain-of-Event paper [23], this prompt adopts the Map-Reduce methodology, integrating the four-step reasoning process: the first step takes place in the Map phase and the remaining three steps occur in the Reduce phase;
- **MR3**: A variation of MR2, where numerical steps replace the conversational format used to describe the reasoning process.

For the Unipol Dataset, a total of three distinct prompts were tested. In this case, only Map-Reduce prompts were considered, as they had demonstrated higher effectiveness in the WCEP experiments. The objective was to apply the best-performing methodology to the new dataset.

Since all Unipol Dataset documents are in Italian, the corresponding prompts were also written in Italian to ensure consistency and maintain the linguistic alignment between the input and the expected output.

- **MR1**: This prompt follows the four-step reasoning process inspired by the Chain-of-Event paper [23]. The first step is executed in the Map phase, while the remaining ones take place in the Reduce phase. The prompt is specifically designed to ensure a coherent and structured summarization of insurance documents, maintaining all critical financial details;

- **MR2**: A variation of MR1, but instead of using a conversational style, it presents the reasoning steps as structured numerical points. This approach ensures a more systematic and logically ordered extraction of information, which can be useful in maintaining consistency across multiple summarizations;

- **MR3**: A hybrid approach between MR1 and MR2, combining a conversational style with a step-by-step structure. Instead of using numerical steps, it organizes the reasoning process using bullet points, striking a balance between readability and structured information extraction.

5.1.3 Evaluation Metrics

The possible evaluation metrics applicable to Multi-Document Summarization have already been discussed in Chapter 3. This thesis employed both supervised and unsupervised metrics for the WCEP experiments. However, for the Unipol experiments, only unsupervised metrics were used, as supervised metrics were not only unfeasible (since the retrieved documents had no reference summaries) but also less informative for the specific characteristics of the dataset.

The evaluation metrics adopted in this study are categorized as follows:

- **Supervised Metrics** (used only for WCEP experiments):
  - *ROUGE-1*, *ROUGE-2*, *ROUGE-L*, *ROUGE-LSUM*: Measure lexical overlap between the generated summary and a reference summary;
  - *BERTScore*: Evaluates semantic similarity using contextual embeddings;
  - *BLEU*: Measures precision in n-gram matching, typically used for translation but applicable to summarization;
  - *G-EVAL*: A model-based evaluation metric that assesses summary quality.

- **Unsupervised Metrics** (used for both WCEP and Unipol experiments):
  - *G-EVAL Coherence*: Evaluates the logical structure of the summary;
  - *G-EVAL Consistency*: Measures factual consistency with the source documents;
  - *G-EVAL Fluency*: Assesses grammatical correctness and readability;
  - *G-EVAL Relevance*: Estimates how well the summary captures key information;
  - *G-EVAL Average*: Provides an overall score based on the previous four criteria.

5.1.4 Large Language Models

As previously stated, the core component of this project is the Large Language Model ChatGPT, which has been exploited for multiple tasks, including conversational interactions with the chatbot back-end, summary and comparison generation, information retrieval, and evaluation.

The experiments conducted in this study utilized different versions of the ChatGPT model, evolving over time to accommodate performance improvements, cost efficiency, and compatibility with dataset characteristics:

- **ChatGPT-3.5 turbo-16k**: Based on the ChatGPT-3.5 architecture, this model featured a context window of 16384 tokens, allowing it to process larger inputs compared to previous versions. It was the first model selected and was primarily used for WCEP experiments;

- **ChatGPT-3.5 turbo-0125**: After the deprecation of ChatGPT-3.5 turbo-16k, this version became the best available alternative for the experiments. While maintaining a context window of 16385 tokens, it introduced efficiency and performance improvements. However, due to its limitations in handling long documents, it was still mainly used for WCEP experiments rather than for the Unipol dataset;

- **ChatGPT-4o mini**: With the release of ChatGPT-4 (and ChatGPT-4o), which significantly outperformed ChatGPT-3.5, the decision was made to transition to the lighter and faster mini variant. This model offered several advantages:

  - Extended context window: 128000 tokens, making it suitable for processing the longer documents in the Unipol dataset;
  - Lower costs: \$0.15 per million input tokens and \$0.60 per million output tokens, less than half the cost of previous versions;
  - High efficiency: Faster response times and better overall performance.

  Given the potential real-world adoption of this system by an insurance company, cost-effectiveness was an important factor in selecting this model.

Additionally, ChatGPT-3.5 was retained as the LLM for evaluation metrics. Since it proved to be sufficiently reliable for evaluation purposes, keeping it ensured a fair comparison across different experiments.

5.2 Results

5.2.1 Initial Experiments and Baseline Analysis

The first step in the experimentation process was to demonstrate the feasibility of summarizing a news dataset (WCEP) using LangChain APIs. The S1 prompt was applied to the WCEP Dataset to establish a baseline, serving as a reference point for evaluating the improvements introduced by the Map-Reduce prompts.

Table 5.3 presents the results obtained using supervised metrics for S1, MR2, and MR3. The goal was to determine whether the Chain-of-Event methodology, implemented in MR2 and MR3, could produce more structured and effective summaries compared to the basic summarization approach used in S1. These experiments were conducted on the WCEP-1 and WCEP-2 dataset versions.

Surprisingly, S1 obtained higher scores than MR2 and MR3, contradicting expectations. A closer analysis revealed that this discrepancy arose due to the

Prompt	LLM	Dataset	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-LSum
S1	3.5 turbo-16k	WCEP-1	0.249	0.102	0.179	0.178
MR2	3.5 turbo-16k	WCEP-2	0.169	0.058	0.169	0.169
MR3	3.5 turbo-16k	WCEP-2	0.137	0.04	0.118	0.118

Prompt	LLM	Dataset	BERTScore	BLEU	G-Eval
S1	3.5 turbo-16k	WCEP-1	0.757	0.041	—
MR2	3.5 turbo-16k	WCEP-2	0.755	0.0	0.642
MR3	3.5 turbo-16k	WCEP-2	0.729	0.0	0.530

Table 5.3: Supervised evaluation results of WCEP experiments baseline.

strong dependence of supervised metrics on reference summaries. Metrics as BLEU assigned very low or even null scores to MR2 and MR3, despite their high quality when evaluated manually. This issue stemmed from two main factors:

- Reference summaries in WCEP were extremely short, often omitting critical details. Supervised metrics like ROUGE and BLEU prioritized exact word overlap, penalizing more detailed responses;
- Summaries which closely resembled the reference text were favored, even when they lacked coherence or completeness.

Given these limitations, it became evident that a different evaluation approach was needed, one that could assess summary quality independently of reference texts. This led to the adoption of G-Eval as an unsupervised metric, enabling a more comprehensive evaluation of coherence, consistency, fluency, and relevance.

5.2.2 Unsupervised Evaluation: Metrics and Experiments

After identifying new unsupervised evaluation metrics based on the approach described in [26], the next set of experiments focused on selecting a dataset subset to test six different prompt versions. These included:

- Two baseline prompts, implemented using both the Stuff and Map-Reduce methodologies;
- Four CoE-based prompts, each applying a structured reasoning process.

For comparison purposes, supervised metrics were also computed again. While these results might suggest that more detailed prompts perform worse when evaluated purely on supervised metrics, the unsupervised evaluation paints a different picture.

Prompt	LLM	Dataset	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-LSum
S1	3.5 turbo-16k	WCEP-3	0.256	0.073	0.155	0.155
S2	3.5 turbo-16k	WCEP-3	0.276	0.124	0.194	0.194
MR1	3.5 turbo-16k	WCEP-3	0.158	0.013	0.119	0.126
MR1.1	3.5 turbo-16k	WCEP-3	0.150	0.021	0.112	0.122
MR2	3.5 turbo-16k	WCEP-3	0.286	0.099	0.167	0.167
MR3	3.5 turbo-16k	WCEP-3	0.302	0.083	0.185	0.185

Prompt	LLM	Dataset	BERTScore	BLEU	G-Eval
S1	3.5 turbo-16k	WCEP-3	0.761	0.023	0.567
S2	3.5 turbo-16k	WCEP-3	0.763	0.057	0.569
MR1	3.5 turbo-16k	WCEP-3	0.698	0.000	0.765
MR1.1	3.5 turbo-16k	WCEP-3	0.692	0.000	0.924
MR2	3.5 turbo-16k	WCEP-3	0.772	0.055	0.523
MR3	3.5 turbo-16k	WCEP-3	0.773	0.047	0.555

Table 5.4: Supervised evaluation results of WCEP-3 combination experiments.

Prompt	LLM	Dataset	Coherence	Consistency	Fluency	Relevance	Average
S1	3.5 turbo-16k	WCEP-3	0.752	0.790	0.704	0.664	0.728
S2	3.5 turbo-16k	WCEP-3	0.787	0.786	0.742	0.736	0.763
MR1	3.5 turbo-16k	WCEP-3	0.766	0.810	0.729	0.698	0.751
MR1.1	3.5 turbo-16k	WCEP-3	0.758	0.782	0.671	0.685	0.724
MR2	3.5 turbo-16k	WCEP-3	0.902	0.855	0.790	0.812	0.840
MR3	3.5 turbo-16k	WCEP-3	0.902	0.865	0.778	0.795	0.836

Table 5.5: Unsupervised G-eval evaluation results of WCEP-3 combination experiments.

As shown in Table 5.4, supervised results for the more structured prompts were not consistently better than the baseline. However, when evaluated in the unsupervised setting (Table 5.5), the situation changed dramatically: both MR2 and MR3 significantly outperformed the other prompts, obtaining nearly identical scores.

One additional experiment involved testing a Map-Reduce baseline prompt where the input text was split into batches of a maximum of 1000 tokens. However, this segmentation did not yield any significant improvements. Due to its lack of advantages, it was not applied to other prompts in subsequent experiments.

5.2.3 Variable-Sized Clusters and Evaluation

After confirming that MR2 and MR3 performed exceptionally well on clusters with a fixed number of documents, the next step was to assess their robustness in a more variable setting. To achieve this, the WCEP-4 dataset, which contains clusters with a variable number of documents, was selected for further testing.

Prompt	LLM	Dataset	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-LSum
MR2	3.5 turbo-16k	WCEP-4	0.263	0.086	0.188	0.190
MR3	3.5 turbo-16k	WCEP-4	0.257	0.082	0.184	0.183

Prompt	LLM	Dataset	BERTScore	BLEU	G-Eval
MR2	3.5 turbo-16k	WCEP-4	0.775	0.040	0.597
MR3	3.5 turbo-16k	WCEP-4	0.768	0.033	0.535

Table 5.6: Supervised evaluation results of WCEP-4 combination experiments.

Prompt	LLM	Dataset	Coherence	Consistency	Fluency	Relevance	Average
MR2	3.5 turbo-16k	WCEP-4	0.720	0.760	0.685	0.674	0.710
MR3	3.5 turbo-16k	WCEP-4	0.721	0.768	0.644	0.631	0.691

Table 5.7: Unsupervised G-eval evaluation results of WCEP-4 combination experiments.

The results in Tables 5.6 and 5.7 reinforce previous observations:

- Supervised metrics continued to show inconsistent results, further confirming their limitations in this task;
- Unsupervised evaluation, however, demonstrated that both MR2 and MR3 maintained high performance levels, even when the number of documents per cluster varied.

These findings highlight the scalability of the Map-Reduce approach with CoE, proving that it remains effective even when the document distribution within clusters is not uniform. Having validated the approach to news summarization, the next logical step was to test its applicability in a different domain. The following experiments aimed to replicate these results in the insurance sector, where document structure and content differ significantly from news articles.

5.2.4 Domain-Specific Testing: The Unipol Dataset

The final phase of the experiments focused on insurance-related documents, using the Unipol Dataset. The objective was to assess the quality of summaries produced by prompts derived from MR2 and MR3 but specifically optimized for this task.

Tables 5.8, 5.9, and 5.10 present the evaluation results for summaries generated using MR1, MR2, and MR3, respectively. Each row corresponds to a different insurance product, which can be considered a document cluster within the dataset.

Figure 5.2 provides a comparative analysis of the results, showing the average G-Eval scores across all evaluation metrics. The data clearly indicate that MR1 and MR2 outperform MR3, achieving average scores of 0.819 and 0.829, respectively, compared to 0.764 for MR3.

Additionally, Figure 5.1 presents a detailed breakdown of each metric, reinforcing that MR1 and MR2 consistently achieve higher scores in coherence, consistency, fluency, and relevance.

Given these findings, the decision was made to integrate MR2 into the final implementation of the Insurance chatbot, as it demonstrated the best balance between coherence, consistency, and relevance for insurance-related document summarization.

Product	Coherence	Consistency	Fluency	Relevance	Average
Km&Servizi Autovetture	0.810	0.814	0.812	0.770	0.802
KM&Servizi Autocarri	0.848	0.861	0.854	0.843	0.852
KM&Servizi Camper	0.816	0.826	0.861	0.848	0.838
Km&Servizi Ciclomotori e Motocicli	0.810	0.858	0.792	0.763	0.806
Casa & Servizi	0.832	0.845	0.805	0.752	0.809
Condominio Più	0.830	0.825	0.828	0.802	0.821
Nuova Prima Global Veicoli diversi da autovetture, ciclomotori e motocicli	0.809	0.850	0.840	0.791	0.823
Nuova Prima Global CVT e Assistenza stradale	0.787	0.741	0.706	0.743	0.744
InViaggio Full	0.854	0.826	0.832	0.802	0.823
Navigare	0.798	0.811	0.749	0.833	0.798
Nuova Prima Global Autovetture	0.847	0.864	0.853	0.827	0.847
InViaggio Ready	0.855	0.848	0.796	0.818	0.829
InViaggio Frequent	0.878	0.863	0.893	0.880	0.878
Nuova Prima Global Ciclomotori e Motocicli	0.801	0.819	0.788	0.800	0.802
Nuova Prima Global CVM e Assistenza	0.822	0.863	0.783	0.820	0.822
Nuova Prima Global Natanti e Motori Amovibili	0.839	0.852	0.791	0.750	0.808
Average	**0.827**	**0.835**	**0.811**	**0.803**	**0.819**

Table 5.8: G-Eval evaluation scores for Unipol Dataset using MR1 prompt.

Product	Coherence	Consistency	Fluency	Relevance	Average
Km&Servizi Autovetture	0.859	0.849	0.854	0.843	0.851
KM&Servizi Autocarri	0.851	0.857	0.821	0.845	0.844
KM&Servizi Camper	0.840	0.829	0.827	0.873	0.843
Km&Servizi Ciclomotori e Motocicli	0.846	0.856	0.807	0.783	0.823
Casa & Servizi	0.827	0.852	0.819	0.830	0.832
Condominio Più	0.828	0.795	0.780	0.841	0.811
Nuova Prima Global Veicoli diversi da autovetture, ciclomotori e motocicli	0.855	0.843	0.794	0.863	0.839
Nuova Prima Global CVT e Assistenza stradale	0.838	0.828	0.819	0.763	0.799
InViaggio Full	0.800	0.796	0.791	0.808	0.798
Navigare	0.869	0.865	0.823	0.835	0.848
Nuova Prima Global Autovetture	0.849	0.845	0.795	0.802	0.823
InViaggio Ready	0.821	0.791	0.803	0.801	0.804
InViaggio Frequent	0.859	0.904	0.857	0.861	0.870
Nuova Prima Global Ciclomotori e Motocicli	0.795	0.806	0.775	0.721	0.774
Nuova Prima Global CVM e Assistenza	0.891	0.861	0.857	0.863	0.868
Nuova Prima Global Natanti e Motori Amovibili	0.877	0.839	0.825	0.781	0.831
Average	**0.844**	**0.838**	**0.816**	**0.82**	**0.829**

Table 5.9: G-Eval evaluation scores for Unipol Dataset using MR2 prompt.

Product	Coherence	Consistency	Fluency	Relevance	Average
Km&Servizi Autovetture	0.793	0.818	0.758	0.755	0.781
KM&Servizi Autocarri	0.701	0.719	0.735	0.708	0.716
KM&Servizi Camper	0.794	0.793	0.666	0.766	0.754
Km&Servizi Ciclomotori e Motocicli	0.798	0.777	0.620	0.767	0.741
Casa & Servizi	0.786	0.761	0.679	0.645	0.718
Condominio Più	0.816	0.800	0.777	0.844	0.809
Nuova Prima Global Veicoli diversi da autovetture, ciclomotori e motocicli	0.766	0.772	0.773	0.755	0.766
Nuova Prima Global CVT e Assistenza stradale	0.773	0.726	0.778	0.788	0.766
InViaggio Full	0.816	0.800	0.819	0.814	0.783
Navigare	0.816	0.789	0.730	0.793	0.769
Nuova Prima Global Autovetture	0.789	0.782	0.689	0.815	0.780
InViaggio Ready	0.795	0.779	0.737	0.775	0.771
InViaggio Frequent	0.781	0.766	0.774	0.775	0.774
Nuova Prima Global Ciclomotori e Motocicli	0.789	0.745	0.630	0.411	0.644
Nuova Prima Global CVM e Assistenza	0.836	0.839	0.791	0.845	0.828
Nuova Prima Global Natanti e Motori Amovibili	0.786	0.782	0.776	0.809	0.788
Average	**0.79**	**0.778**	**0.733**	**0.754**	**0.764**

Table 5.10: G-Eval evaluation scores for Unipol Dataset using MR3 prompt.

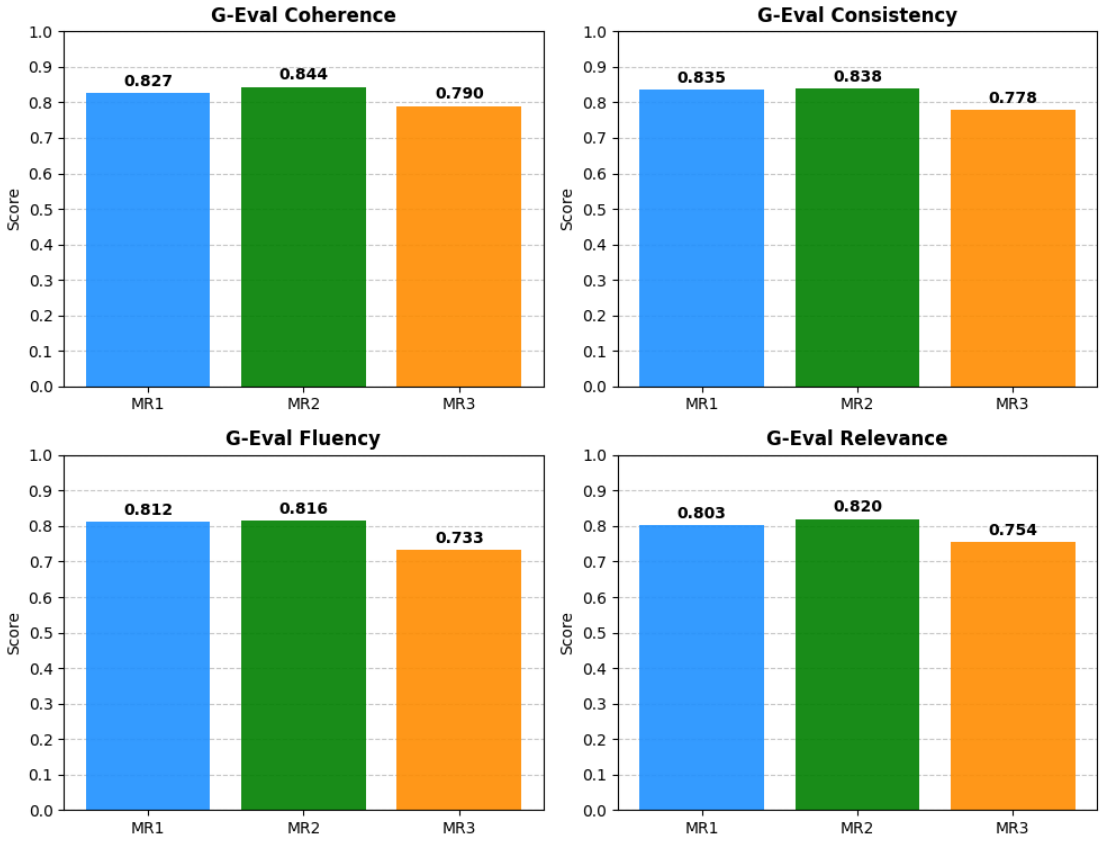


Figure 5.1: Comparation of the average of the evaluation metrics results for Unipol prompts.

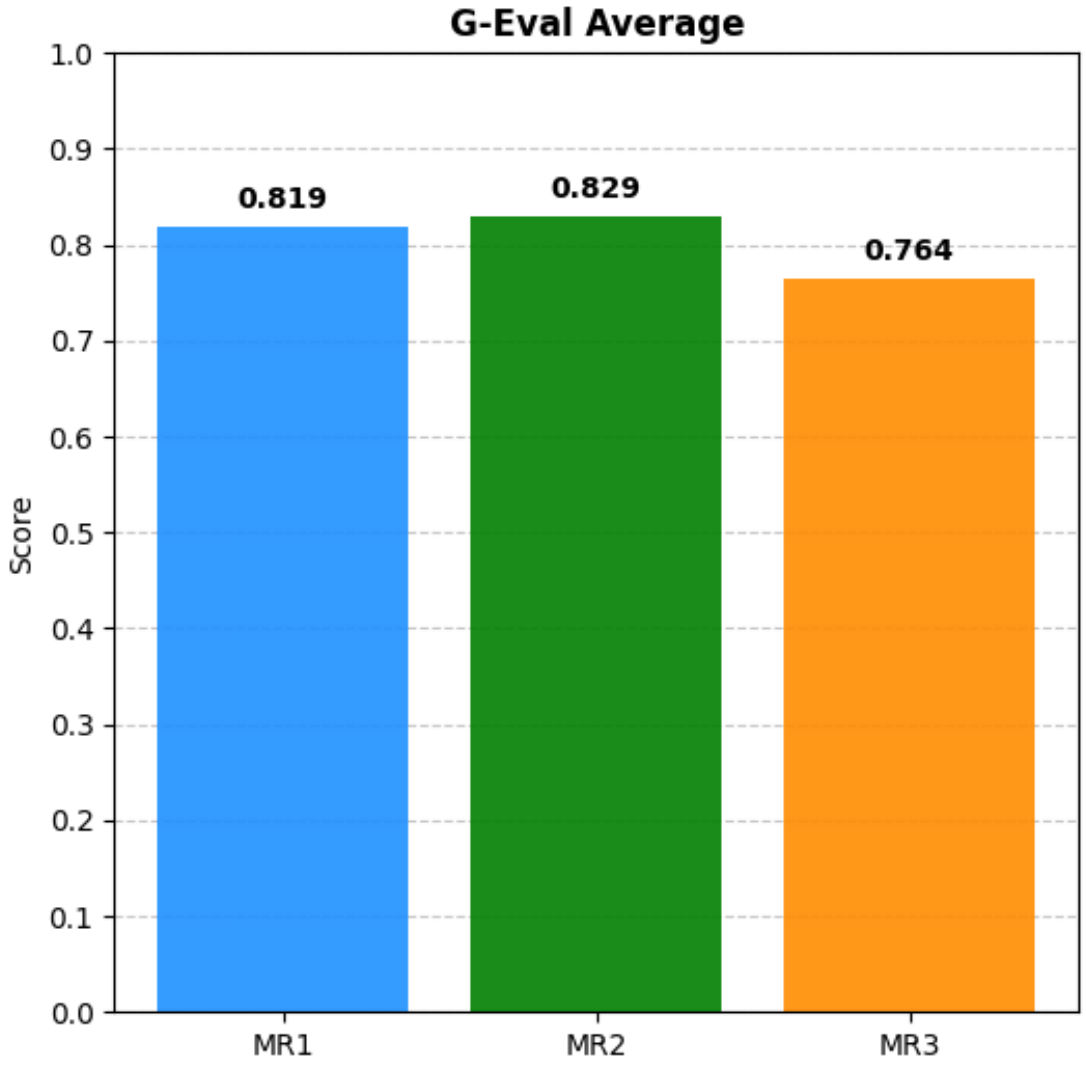


Figure 5.2: Comparation of the G-Eval Average results for Unipol prompts.

Chapter 6

Conclusion and Future Works

The principal scope of this thesis was to affirm that a Multi-Document Summarization Pipeline, powered by a Large Language Model, can provide a good quality summary also in the insurance field, as it already can do for news summarization.

A new dataset, made of information documents related to Unipol products, was created, and the Chain-of-Event prompting technique was applied to summarize it and to prove that the goal could be effectively reached. After testing its efficacy, a Summarization Pipeline was created to be employed in the insurance domain. This pipeline was made using LangChain APIs and ChatGPT 4o-mini LLM, and is the core of a Streamlit-based chatbot with multiple functionalities. The chatbot can summarize multiple documents related to the same product to provide its overview, but can also summarize documents related to two different products to highlight their similarities and differences. The chatbot was also enriched with a RAG to reply to customers' questions, taking information directly from the documents. The summary results were validated by introducing a new evaluation methodology, using ChatGPT 3.5 and the G-Eval framework, to evaluate four distinct metrics: Coherence, Consistency, Fluency, and Relevance. The results of the summaries were positive, showing that the chosen techniques proved very effective.

This work is very relevant, since it proposes a concrete and innovative application of GenAI in a structured domain that was quite unexplored until now. It is a technique that can help customers to obtain information about insurance products without a big knowledge and study of the topic, but also companies to provide their customers with more advanced tools to choose the correct product or to obtain quick answers to their questions about insurance products.

The provided solution still has some limitations, since it utilizes only one dataset that is made of only 16 products and 111 documents. Documents and prompts are all in Italian, lacking a direct comparison with multi-language versions. Finally, the evaluation is only automatic and is based only on G-Eval framework results, which for various reasons could also be biased.

Despite all its limitations, the solution provides many openings. The same approach could also be applied to different sectors, such as finance and healthcare, just adapting prompts to the specific topic. The model that was selected is ChatGPT 4o-mini, but different models, such as Claude, Gemini, or other versions of the same GPT, could be applied, simply replacing some parameters in the API calls. Another aspect that could evolve is the quality evaluation of the resumes. Currently, the model utilizes the G-Eval framework and four metrics to evaluate summaries, but it could be interesting to add human evaluations to the dataset. Finally, the conversational interface could also be improved, including more advanced interactions, for example, using the voice.

In a world where information access and comprehension are more and more complex, GenAI-based tools could make knowledge more inclusive, clear, and accessible for everyone. The hope is that this work could be followed by similar papers that provide a similar solution to deal with common problems people face every day.

Bibliography

[1] John McCarthy, Marvin L. Minsky, Nathaniel Rochester, and Claude E. Shannon. «A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955». In: *AI Magazine* 27.4 (Dec. 2006), p. 12. DOI: 10.1609/aimag.v27i4.1904. URL: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/1904 (cit. on p. 1).

[2] A. M. Turing. «I.—Computing machinery and intelligence». In: *Mind* LIX.236 (Oct. 1950), pp. 433–460. ISSN: 0026-4423. DOI: 10.1093/mind/LIX.236.433. eprint: https://academic.oup.com/mind/article-pdf/LIX/236/433/30123314/lix-236-433.pdf. URL: https://doi.org/10.1093/mind/LIX.236.433 (cit. on pp. 1, 9).

[3] Elizabeth D. Liddy. «Natural language processing.» In: (2001). URL: https://www.bibsonomy.org/bibtex/24864613c44e0472d987b3933e14ffc54/zromero (cit. on p. 4).

[4] Jan Sawicki, Maria Ganzha, and Marcin Paprzycki. «The State of the Art of Natural Language Processing—A Systematic Automated Review of NLP Literature Using NLP Techniques». In: *Data Intelligence* 5.3 (Aug. 2023), pp. 707–749. ISSN: 2641-435X. DOI: 10.1162/dint_a_00213. eprint: https://direct.mit.edu/dint/article-pdf/5/3/707/2158286/dint_a_00213.pdf. URL: https://doi.org/10.1162/dint%5C_a%5C_00213 (cit. on p. 16).

[5] Diksha Khurana, Aditya Koli, Kiran Khatter, and Sukhdev Singh. «Natural language processing: state of the art, current trends and challenges». In: *Multimedia Tools and Applications* 82.3 (Jan. 2023), pp. 3713–3744. ISSN: 1573-7721. DOI: 10.1007/s11042-022-13428-4. URL: https://doi.org/10.1007/s11042-022-13428-4 (cit. on p. 16).

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. «Attention is All you Need». In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017. URL: https://

proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf (cit. on pp. 18, 22).

[7] Keng-Boon Ooi et al. «The Potential of Generative Artificial Intelligence Across Disciplines: Perspectives and Future Directions». In: *Journal of Computer Information Systems* 0.0 (2023), pp. 1–32. DOI: 10.1080/08874417.2023.2261010. eprint: https://doi.org/10.1080/08874417.2023.2261010. URL: https://doi.org/10.1080/08874417.2023.2261010 (cit. on p. 24).

[8] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. *A Comprehensive Overview of Large Language Models*. 2024. arXiv: 2307.06435 [cs.CL]. URL: https://arxiv.org/abs/2307.06435 (cit. on p. 28).

[9] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. *OpenAI Gym*. 2016. arXiv: 1606.01540 [cs.LG]. URL: https://arxiv.org/abs/1606.01540 (cit. on p. 35).

[10] Daniel Martin Katz, Michael James Bommarito, Shang Gao, and Pablo Arredondo. «GPT-4 passes the bar exam». en. In: *Philos. Trans. A Math. Phys. Eng. Sci.* 382.2270 (Apr. 2024), p. 20230254 (cit. on p. 39).

[11] Rania Abdelghani, Yen-Hsiang Wang, Xingdi Yuan, Tong Wang, Pauline Lucas, Hélène Sauzéon, and Pierre-Yves Oudeyer. «GPT-3-Driven Pedagogical Agents to Train Children's Curious Question-Asking Skills». In: *International Journal of Artificial Intelligence in Education* 34.2 (June 2023), pp. 483–518. ISSN: 1560-4306. DOI: 10.1007/s40593-023-00340-7. URL: http://dx.doi.org/10.1007/s40593-023-00340-7 (cit. on p. 39).

[12] Yang Liu. *Fine-tune BERT for Extractive Summarization*. 2019. arXiv: 1903.10318 [cs.CL]. URL: https://arxiv.org/abs/1903.10318 (cit. on pp. 42, 43).

[13] Wen Xiao, Iz Beltagy, Giuseppe Carenini, and Arman Cohan. *PRIMERA: Pyramid-based Masked Sentence Pre-training for Multi-document Summarization*. 2022. arXiv: 2110.08499 [cs.CL]. URL: https://arxiv.org/abs/2110.08499 (cit. on p. 42).

[14] Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. «Multi-News: A Large-Scale Multi-Document Summarization Dataset and Abstractive Hierarchical Model». In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Ed. by Anna Korhonen, David Traum, and Lluís Màrquez. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 1074–1084. DOI: 10.18653/v1/P19-1102. URL: https://aclanthology.org/P19-1102 (cit. on pp. 44, 45).

[15] Jay DeYoung, Iz Beltagy, Madeleine van Zuylen, Bailey Kuehl, and Lucy Lu Wang. *MS2: Multi-Document Summarization of Medical Studies*. 2021. arXiv: 2104.06486 [cs.CL]. URL: https://arxiv.org/abs/2104.06486 (cit. on p. 44).

[16] Anushka Gupta, Diksha Chugh, Anjum, and Rahul Katarya. *Automated News Summarization Using Transformers*. 2021. arXiv: 2108.01064 [cs.CL]. URL: https://arxiv.org/abs/2108.01064 (cit. on p. 47).

[17] Chantal Shaib, Millicent L. Li, Sebastian Joseph, Iain J. Marshall, Junyi Jessy Li, and Byron C. Wallace. *Summarizing, Simplifying, and Synthesizing Medical Evidence Using GPT-3 (with Varying Success)*. 2023. arXiv: 2305.06299 [cs.CL]. URL: https://arxiv.org/abs/2305.06299 (cit. on p. 47).

[18] Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. *A Survey on RAG Meeting LLMs: Towards Retrieval-Augmented Large Language Models*. 2024. arXiv: 2405.06211 [cs.CL]. URL: https://arxiv.org/abs/2405.06211 (cit. on p. 53).

[19] Jiatong Li, Yunqing Liu, Wenqi Fan, Xiao-Yong Wei, Hui Liu, Jiliang Tang, and Qing Li. «Empowering Molecule Discovery for Molecule-Caption Translation With Large Language Models: A ChatGPT Perspective». In: *IEEE Transactions on Knowledge and Data Engineering* 36.11 (Nov. 2024), pp. 6071–6083. ISSN: 2326-3865. DOI: 10.1109/tkde.2024.3393356. URL: http://dx.doi.org/10.1109/TKDE.2024.3393356 (cit. on p. 53).

[20] Ruixuan Sun, Xinyi Li, Avinash Akella, and Joseph A. Konstan. *Large Language Models as Conversational Movie Recommenders: A User Study*. 2024. arXiv: 2404.19093 [cs.IR]. URL: https://arxiv.org/abs/2404.19093 (cit. on p. 53).

[21] Shuaiqi Liu, Jiannong Cao, Ruosong Yang, and Zhiyuan Wen. *Long Text and Multi-Table Summarization: Dataset and Method*. 2023. arXiv: 2302.03815 [cs.CL]. URL: https://arxiv.org/abs/2302.03815 (cit. on p. 56).

[22] Ziao Wang, Zelin Jiang, Xiaofeng Zhang, Jaehyeon Soon, Jialu Zhang, Wang Xiaoyao, and Hongwei Du. «Beyond Pure Text: Summarizing Financial Reports Based on Both Textual and Tabular Data». In: *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*. Ed. by Edith Elkind. Main Track. International Joint Conferences on Artificial Intelligence Organization, Aug. 2023, pp. 5233–5241. DOI: 10.24963/ijcai.2023/581. URL: https://doi.org/10.24963/ijcai.2023/581 (cit. on p. 56).

[23] Songlin Bao, Tiantian Li, and Bin Cao. «Chain-of-event prompting for multi-document summarization by large language models». In: *International Journal of Web Information Systems* 20.3 (Jan. 2024), pp. 229–247. ISSN: 1744-0084. DOI: 10.1108/IJWIS-12-2023-0249. URL: https://doi.org/10.1108/IJWIS-12-2023-0249 (cit. on pp. 58, 88, 102, 104).

[24] Demian Gholipour Ghalandari, Chris Hokamp, Nghia The Pham, John Glover, and Georgiana Ifrim. «A Large-Scale Multi-Document Summarization Dataset from the Wikipedia Current Events Portal». In: *CoRR* abs/2005.10070 (2020). arXiv: 2005.10070. URL: https://arxiv.org/abs/2005.10070 (cit. on p. 59).

[25] *LangChain Documentation: Summarization Use Case*. URL: https://python.langchain.com/v0.1/docs/use_cases/summarization/ (cit. on pp. 61, 62, 102–104).

[26] Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. *G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment*. 2023. arXiv: 2303.16634 [cs.CL]. URL: https://arxiv.org/abs/2303.16634 (cit. on pp. 64, 108).

Ringraziamenti

Ed eccomi, finalmente arrivato alla fine di questo lungo percorso, dedico questo spazio alle persone che durante tutti questi anni mi hanno fornito il loro supporto ed, in qualche modo, dato la forza di raggiungere anche questo traguardo.

Inizio ringraziando il mio relatore, il Professore Paolo Garza, per tutto il supporto e l'infinita disponibilità dimostratami durante la stesura di questo elaborato di tesi. È stato un vero piacere poter collaborare con un professore, ma soprattutto una persona, così in gamba e disponibile.

Un altro grosso ringraziamento va ad Alessandro, che mi ha seguito durante lo sviluppo di questo progetto e fornito consigli preziosi. Una menzione va inoltre a tutta Data Reply, in particolare nelle persone di Francesco e Stefano, che mi ha permesso di svolgere questo lavoro di tesi e contemporaneamente di crescere professionalmente.

Un grazie di cuore va alle persone che realmente ci sono state dal primo momento del mio lungo percorso universitario, Mamma e Papà, perché dal primo all'ultimo giorno mi hanno supportato, economicamente ma non solo, spronandomi a non mollare ed a raggiungere i miei obiettivi. Spero che nella vita sarete sempre al mio fianco per festeggiare qualsiasi traguardo arriverà.

Un mondo di gratitudine e di amore vanno alla *mia* Dajana, che giorno dopo giorno mi ha visto crescere, cambiare, gioire, disperarmi e poi gioire di nuovo e che, nonostante tutti gli imprevisti capitati sul percorso, è ancora qui a darmi forza. Sei la mia ancora ed io spero di esserlo sempre per te, i miei traguardi sono anche i tuoi ed i tuoi saranno anche i miei. Spero di trovarti sempre al mio fianco per orientarmi di fronte a tutte le difficoltà che supereremo insieme.

Un altro pensiero va a tutte le persone che hanno fatto parte del mio percorso universitario, anche se solo per poco, ognuno di voi ha portato in qualche modo ad una mia crescita personale e vi sono grato per questo. In particolare non posso

non menzionare Dario, Fabio e Peppe, che più di tutti mi hanno accompagnato in questo lungo viaggio e sono convinto saranno presenti anche una volta concluso.

Infine, l'ultimo ringraziamento lo faccio a me stesso. Come già avevo scritto nella mia precedente tesi: tutto, anche le cose che ci sembrano insormontabili, difficili o irrisolvibili, possono essere superate, basta volerlo ed impegnarsi! La fine di questo percorso è l'inizio di una nuova vita.