Republic of Tunisia
الجمهورية التونسية
Ministry of Higher Education and Scientific Research
وزارة التعليم العالي و البحث العلمي

---

University of Tunis El Manar
جامعة تونس المنار
National Engineering School of Tunis
المدرسة الوطنية للمهندسين تونس

**Engineering Internship Report**

**Information and Communication Technologies Department**

---

# A Comparative Study of Transformer-Based Language Models On A Project Specifications Understanding

---

*Elaborated by :*
Amen Allah BERREJEB
*Supervised by:*
Mr. Mohamed Aziz ATITALLAH

University year: 2023/2024

## Abstract

This internship report presents a comparative study of Transformers-based language models, with a primary focus on their efficacy in understanding project specifications. The analysis delves into various models, highlighting their strengths and weaknesses in the context of extracting meaningful insights from project requirements. Additionally, the report provides an overview of key research papers in the realm of Large Language Models (LLMs). This comparative study aims to furnish readers with a clearer picture of the capabilities and limitations of current Transformer architectures in project specification comprehension.


**Keywords:** Transformers, Language Models, Project Specifications, Comparative Study, Large Language Models, Comprehension, Research Overview.



## Résumé

Ce rapport de stage présente une étude comparative des modèles linguistiques basés sur les Transformers, en mettant l'accent sur leur efficacité à comprendre les spécifications du projet. L'analyse approfondit différents modèles, mettant en lumière leurs forces et faiblesses dans le contexte de l'extraction d'informations pertinentes des exigences du projet. De plus, le rapport donne un aperçu des principaux articles de recherche dans le domaine des Grands Modèles Linguistiques (GMLs). Cette étude comparative vise à fournir aux lecteurs une image plus claire des capacités et des limitations des architectures Transformer actuelles dans la compréhension des spécifications de projet.


**Mots Clés :** Transformers, Modèles Linguistiques, Spécifications de Projet, Étude Comparative, Grands Modèles Linguistiques, Compréhension, Aperçu de la Recherche.

# Acknowledgements

I would like to thank our supervisor, Mr. Mohamed Aziz Atitallah, who gave me the opportunity to work on this research project. This study deepened our understanding of the intricacies and potentials of modern natural language processing techniques. Finally, I warmly thank the members of the jury who agreed to evaluate this work.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

**AI** Artificial Intelligence

**NLP** Natural Language Processing

**NLU** Natural Language Understanding

**MnC** The company name

**LLM** Large Language Model

**GPT** Generative Pretrained Transformer

**BERT** Bidirectional Encoder Representations from Transformers

**BLEU** Bilingual Evaluation Understudy

**ROUGE** Recall-Oriented Understudy for Gisting Evaluation

**NSP** Next Sentence Prediction

**MSM** Masked Language Model

**ALBERT** A Lite Bidirectional Encoder Representations from Transformers

**T5** Text-To-Text Transfer Transformer

**RPA** Robotic Process Automation

**NLTK** Natural language Toolkit

**IPO** Intelligent Process Outsourcing

# General Introduction

## Context

Over the past few years, artificial intelligence (AI) has significantly transformed a myriad of sectors, including the healthcare domain. Natural Language Processing (NLP), a subset of AI, focuses on enabling computers to understand and interpret human language. The importance of NLP lies in its potential to bridge the gap between human communication and digital understanding. As AI continues to be a subject of interest and growth in the research community, the focus on large transformer-based language models, a branch of NLP, has grown exponentially.

Given the increasing emphasis on understanding project specifications in the IT sector, particularly in departments such as web management & development, there's a clear interest in utilizing AI, specifically language models, to streamline and enhance this process.

At the heart of this revolution is the application of transformer architectures, which have proved to be pivotal in developing state-of-the-art models in the NLP domain. Such advancements have been leveraged by companies to derive actionable insights, streamline processes, and improve decision-making.

## Problematic

In a rapidly digitizing world, how can transformer-based language models be effectively harnessed to enhance the understanding of project specifications, especially in complex sectors such as companies management & development?

## Objectives

The core objectives of this internship encompass:

- Understanding the intricate specifications of the M&C project.
- Conducting a thorough review of the state-of-the-art research in large language models to derive pertinent insights and knowledge.

- Development of a proof of concept titled ”ADADGPT” by creating various chatbot models.
- Comparing the outcomes and results of these models to discern the most effective approach for project specification understanding.

## Report Organization

This report is meticulously structured to offer readers a clear insight into the undertaken internship project:

- Chapter 1 delves into a general overview of the company and the scope of the project, encompassing the company’s objectives, partners, core services, and the project’s ambit.
- Chapter 2 provides a comprehensive review of existing literature and research, focusing on large transformer-based language models and their application in natural language understanding .
- Chapter 3 underscores the methodology and processes involved in developing the proof of concept, ADADGPT.

# Chapter 1

# General presentation of the company and the project

# Introduction : The Company Main goals and Presentation

Travel, airline and hospitality companies must reinvent themselves and modernize technology to keep up with changing market and challenges. M &C IT CONSULTING empower travel companies achieve this goal in the area of finance and accounting services .

## 1.1 Core services of the company

With years of proven Airline and Travel expertise, M&C IT consulting offers unique capabilities to develop, implement, support and operate optimized end-to-end digital financial applications, helping airlines reduce cost, complexity while improving speed and agility .

### 1.1.1 Intelligent Process Outsourcing: M&C's innovative approach :

M&C's IPO pushes the boundaries of traditional BPO models by leveraging Robotics and Artificial Intelligence with a continuous search for process optimization. Our automation driven IPO services enable airlines to turn revenue accounting and finance into a strategic function through the best combination of people, processes, and technology. Figure 1.2 explains the IPO approach used in the company .

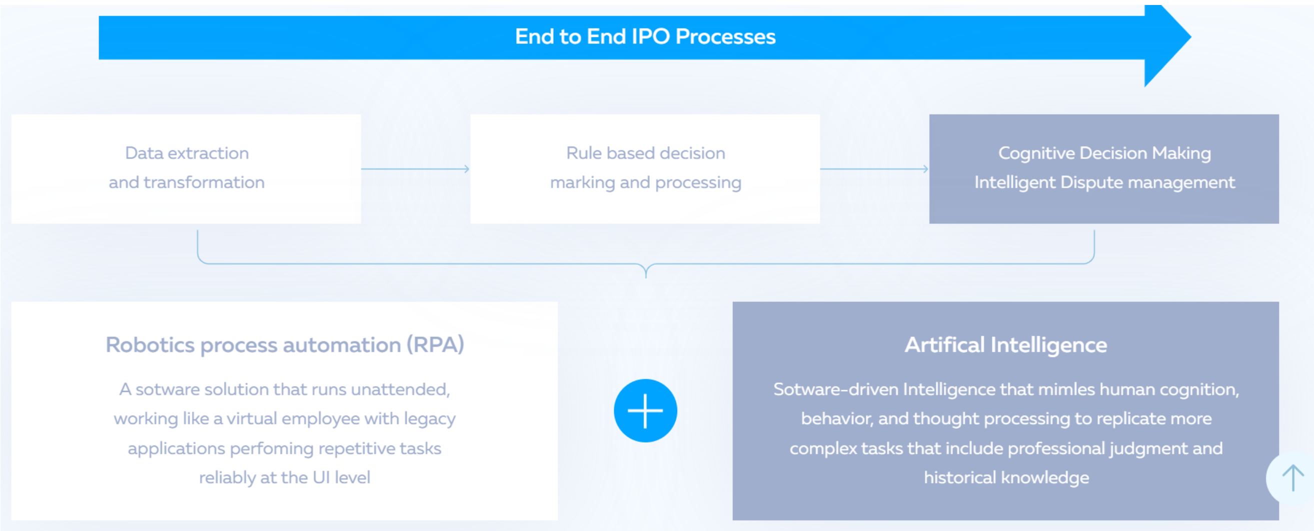


Figure 1.1: End-to-End IPO M&C's approach[4].

### 1.1.2 Robotics process automation (RPA)

Robotic process automation is a software technology that makes it easy to build, deploy, and manage software robots that emulate humans actions interacting with digital systems and software.
Figure 1.2 explains more how RPA works in general .

Figure 1.2: How RPA Works .

### 1.1.3 ADAD: Payment Reconciliation Platform

ADAD: Payment Reconciliation Platform is dedicated to the airline industry, ADAD payment reconciliation platform is an agnostic end to end payment reconciliation platform that can be easily integrated with the airline payment and finance ecosystem. ADAD automates the validation and reconciliation of transactions across multiple payment providers, bank accounts and platforms

Some of the first ADAD dashboards are illustrated in figure 1.2.

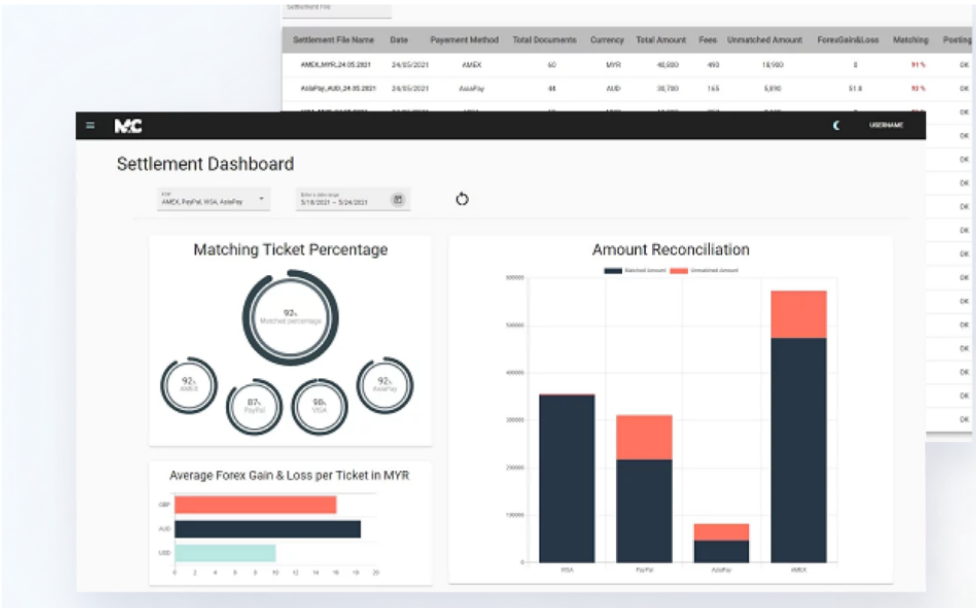


Figure 1.3: ADAD Settlement Dashboard.

### 1.1.4 M&C's Partners

M&C IT CONSULTING helped a lot of airlines in different countires either by ADAD , or RPA and IPO services.


Figure 1.4: Examples of Airlines helped by M&C locations.

## 1.2 The Internship project scope

During my internship at the company, I was positioned within the ADAD Management & Development department. This department was structured into three distinct teams: Functional, Development (Dev), and DevOps. The product owner introduced three innovative ideas, all rooted in Artificial Intelligence. These ideas were subsequently transformed into three separate internship opportunities. My specific role revolved around one of these ideas: conducting a comparative study on transformers-based large language models, specifically tailored to understanding ADAD specifications.

# Conclusion

In this first chapter, we've introduced the company, talking about its main goals, partners, and services. We also outlined what this project is about. This basic information sets the scene for the next chapters and helps understand the details of the internship project we'll discuss later.

# Chapter 2

# State-Of-The-Art: Reviewing Large Language Models Architectures and Evaluation Metrics

# Introduction

In this chapter, we will review some of the NLP research papers , how the attention mehcanism works , and LLMs architecture like : T5 , BERT , ALBERT and GPT .

## 2.1 Large Language Models

### 2.1.1 Attention Mechanism

Self-attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence. Self-attention has beenused successfully in a variety of tasks including reading comprehension, abstractive ,summarization,textual entailment and learning task-independent sentence representations [7].

Most competitive neural sequence transduction models have an encoder-decoder structure . Here, the encoder maps an input sequence of symbol representations (x1, ..., xN) to a sequence of continuous representations z = (z1, ..., zN). Given z, the decoder then generates an output sequence (y1, ..., yM) of symbols one element at a time. At each step the model is auto-regressive, consuming the previously generated symbols as additional input when generating the next [7].

The figure 2.1 shows the Tranformer's architecture [7].

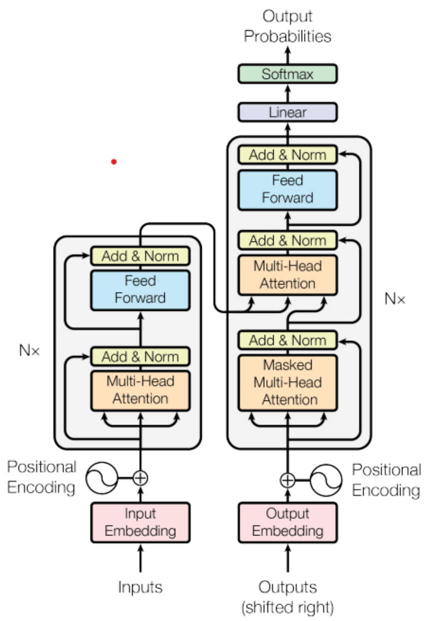


Figure 2.1: The Transformer - model architecture [7].

An attention function can be described as mapping a query and a set of key-value pairs to an output,where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key as shown in figure 2.2 [7].

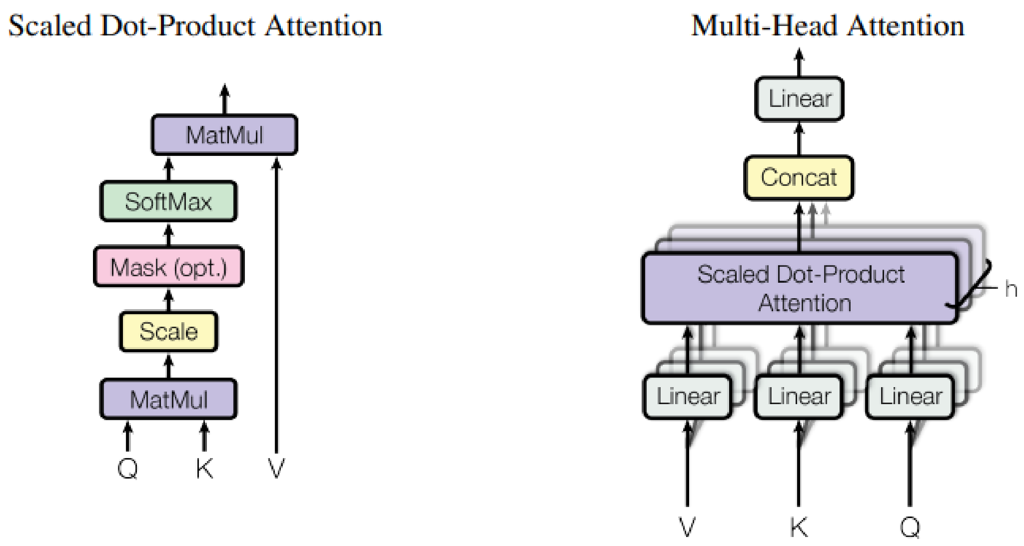


Figure 2.2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel [7].

### 2.1.2 BERT : Bidirectional Encoder Representations from Transformers

BERT (Bidirectional Encoder Representations from Transformers) is a revolutionary deep learning model designed for natural language processing, transforming the landscape of machine learning applications by enabling superior understanding and representation of contextualized word meanings in sentences [2].

We introduce BERT and its detailed implementation in this section. There are two steps in our framework: pre-training and fine-tuning. During pre-training, the model is trained on unlabeled data over different pre-training tasks. For finetuning, the BERT model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameter [2].

The figure 2.3 shows how BERT Was finetuned and pre-trained [2]:

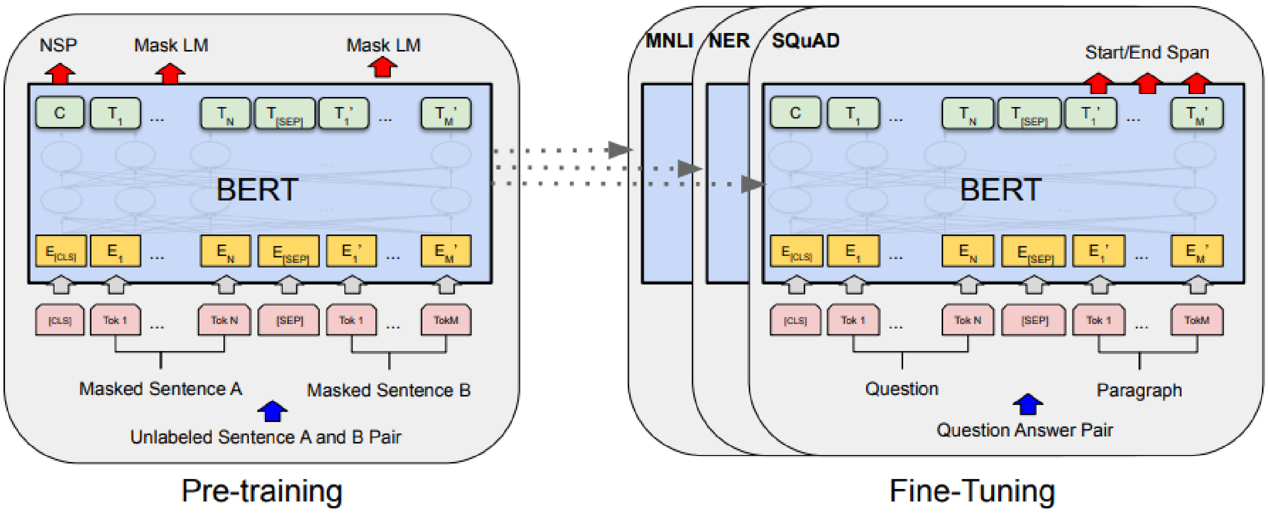


Figure 2.3: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers) [2].

BERT's model architecture is built on a multi-layer bidirectional Transformer encoder, closely aligned with the original design introduced in the tensor2tensor library. Given that Transformer architectures have become standard in the field and our implementation aligns closely with the original, we will not delve deeply into its intricate details. For a comprehensive understanding, we refer readers to Vaswani et al. (2017) and to insightful resources such as "The Annotated Transformer [2].

We can use the following notations [2]:

- **L** denotes the number of layers (i.e., Transformer blocks).
- **H** represents the hidden size.
- **A** stands for the number of self-attention heads.

Mainly, we present results on two model configurations [2]:

- **BERT-BASE**: L=12, H=768, A=12, with a total of 110M parameters.
- **BERT-LARGE**: L=24, H=1024, A=16, boasting 340M parameters in total.

BERT utilizes two pre-training tasks. The first is the "masked LM" (MLM) where a percentage of input tokens are randomly masked, and the model predicts these masked tokens. Instead of always replacing the masked word with the `[MASK]` token, variations are applied: 80% of the time it is replaced with `[MASK]`, 10% with a random token, and 10% remains unchanged. The goal is to predict the original token [2].

The second task is ”Next Sentence Prediction” (NSP). This aims at understanding the relationship between two sentences, essential for tasks like Question Answering and Natural Language Inference. In this, two sentences, A and B, are chosen. Half the time, B genuinely follows A (`IsNext`), while the other half, B is random (`NotNext`). This simple pre-training has proven beneficial for QA and NLI tasks [2].

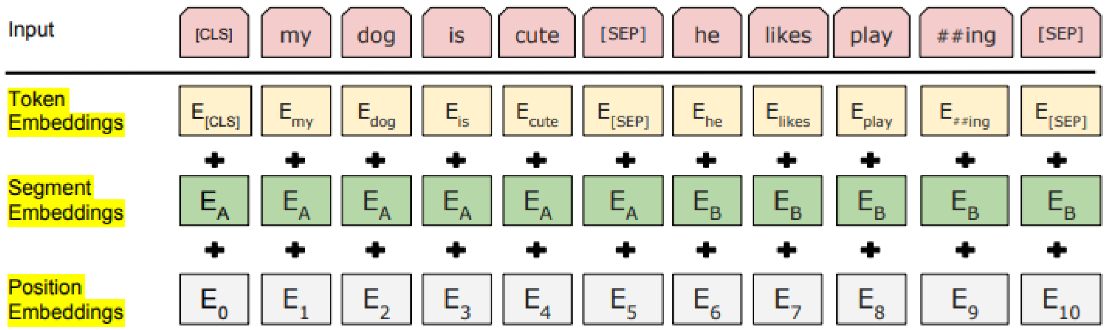


Figure 2.4: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings [2].

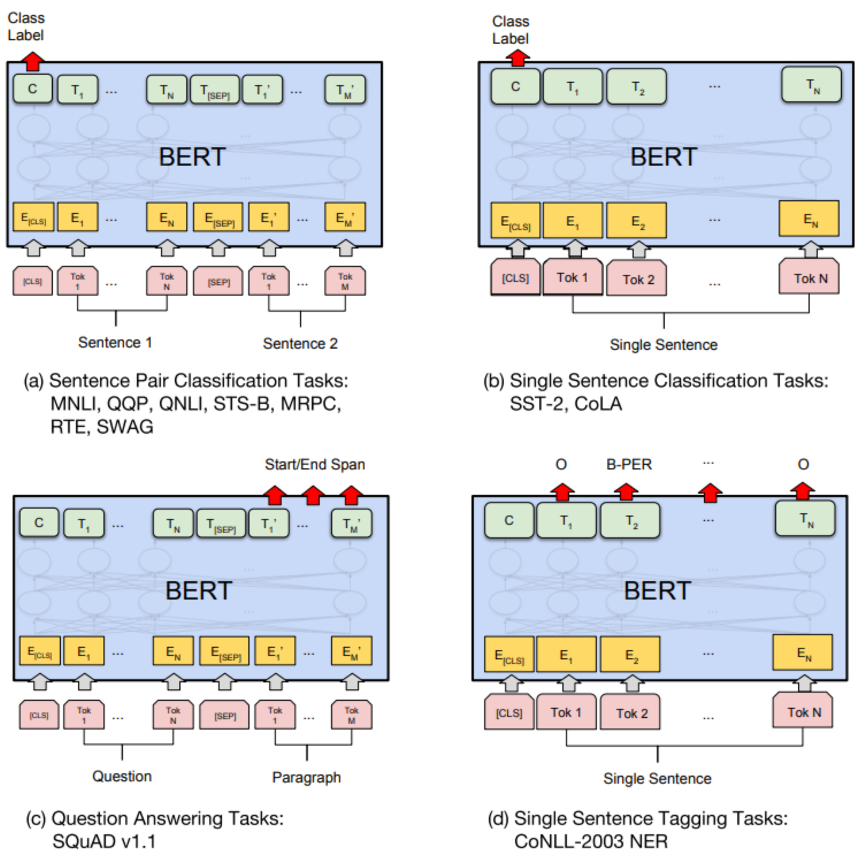


Figure 2.5: Illustrations of Fine-tuning BERT on Different Tasks [2].

### 2.1.3 ALBERT: A Lite BERT For Self-Supervised Learning Of Language Representations

The ALBERT architecture draws its foundation from BERT, utilizing a transformer encoder and GELU nonlinearities. In accordance with BERT's notation [3]:

- **E** represents the vocabulary embedding size.
- **L** stands for the number of encoder layers.
- **H** denotes the hidden size.

Following established practices, the feed-forward/filter size is set at (4H), while the number of attention heads is determined as (H/64) [3].

ALBERT incorporates a pretraining loss focused on predicting the sequence of two adjoining text segments. This concept of addressing discourse coherence in pretraining objectives isn't new and has roots in studies related to discourse coherence and cohesion. Among the myriad phenomena identified, the primary ones link adjacent text segments [3].

In practice, many effective objectives are relatively straightforward. For instance, models like Skip-thought and FastSent learn sentence embeddings by utilizing the encoding of a sentence to anticipate words in surrounding sentences. Some models lean towards predicting forthcoming sentences rather than just neighboring ones, while others hinge on predicting explicit discourse markers. ALBERT's loss shares similarities with the objective where sentence embeddings are learned to determine the sequence of two successive sentences. However, an essential distinction is that ALBERT's loss is applied to textual segments, not just sentences [3].

While BERT's loss hinges on predicting if the second segment in a pair has been exchanged with a segment from a different document, ALBERT's approach finds that sentence ordering provides a more rigorous pretraining task and proves more beneficial for specific subsequent tasks. Simultaneously, another model also explores predicting the sequence of two successive text segments, though they merge it with the original next sentence prediction, forming a three-class classification task instead of analytically contrasting the two [3].

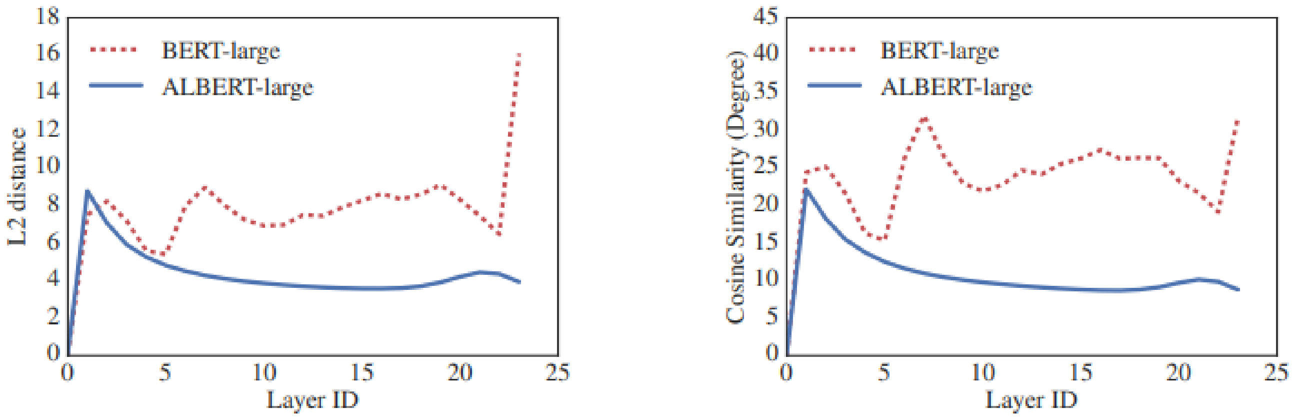


Figure 2.6: Shows the L2 distances and cosine similarity of the input and output embeddings for each layer, using BERT-large and ALBERT-large configurations . We observe that the transitions from layer to layer are much smoother for ALBERT than for BERT. These results show thatweight-sharing has an effect on stabilizing network parameters. Although there is a drop for both metrics compared to BERT, they nevertheless do not converge to 0 even after 24 layers. This shows that the solution space for ALBERT parameters is very different from the one found by DQE [3].

| Model | Parameters | Layers | Hidden | Embedding | Parameter-sharing |
|---|---|---|---|---|---|
| BERT base | 108M | 12 | 768 | 768 | False |
| BERT large | 334M | 24 | 1024 | 1024 | False |
| ALBERT base | 12M | 12 | 768 | 128 | True |
| ALBERT large | 18M | 24 | 1024 | 128 | True |
| ALBERT xlarge | 60M | 24 | 2048 | 128 | True |
| ALBERT xxlarge | 235M | 12 | 4096 | 128 | True |

Table 2.1: The configurations of the main BERT and ALBERT models analyzed in the reference paper [3].

### 2.1.4 T5 : Text-to-Text Transfer Transformer

Text-to-Text Transfer Transformer (T5) is a versatile machine learning model designed by Google Research, which interprets all NLP tasks as a text-to-text problem, converting input text sequences into target output text sequences [6].

The foundational concept behind T5 is the ”text-to-text” framework, where every text processing challenge is viewed as converting an input text into a corresponding output text. This universal approach finds its roots in previous methodologies that have proposed various unifying frameworks for NLP tasks. Some have suggested interpreting all text-related problems through the lens of question answering, language modeling, or span extraction tasks [6].

What makes the text-to-text paradigm so impactful is its uniformity: it allows for the consistent application of the model, objective, training methodology, and decoding process across a vast array of tasks. This consistent approach empowers T5 to be tested across diverse English-based NLP challenges, encompassing areas like question answering, document summarization, and sentiment classification, among others [6].

By embracing this unified methodology, there is the added advantage of drawing comparative insights on the efficacy of different transfer learning objectives, the use of various unlabeled datasets, and other influential factors. This exploration also paves the way for understanding the boundaries of transfer learning in NLP, especially when pushing the scales of models and datasets beyond conventional limits [6].

The figure 2.7 illustrates examples of T5 using in NLP tasks [6]:

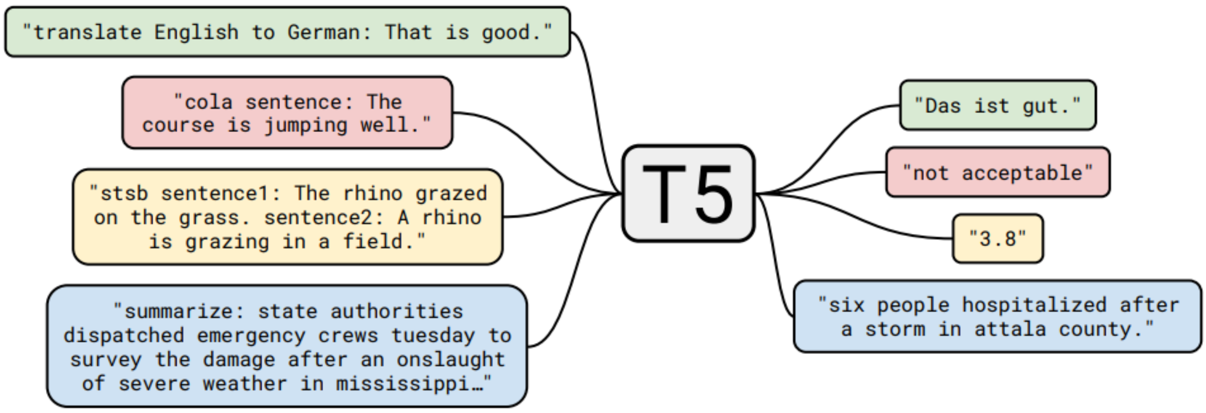


Figure 2.7: A diagram of our text-to-text framework. Every task we consider—including translation, question answering, and classification—is cast as feeding our model text as input and training it to generate some target text. This allows us to use the same model, loss function, hyperparameters, etc. across our diverse set of tasks. It also provides a standard testbed for the methods included in our empirical survey. “T5” refers to our model, which we dub the “Text-to-Text Transfer Transformer” [6].

### 2.1.5 GPT: Generative Pretrained Transformer

For The GPT architecture, authors used the Transformer, which has been shown to perform strongly on various tasks such as machine translation , document generation, and syntactic parsing .This model choice provides us with a more structured memory for handling long-term dependencies in text, compared to alternatives like recurrent networks,resulting in robust transfer performance across diverse tasks. During transfer, they utilize task-specific input adaptations derived from traversal-style approaches , which process structured text input as a single contiguous sequence of tokens. As they demonstrate in their experiments, these adaptations enable us to fine-tune effectively with minimal changes to the architecture of the pre-trained model [5].

The figure 2.8 illustrates how the GPT was fine tuned on different task [5].

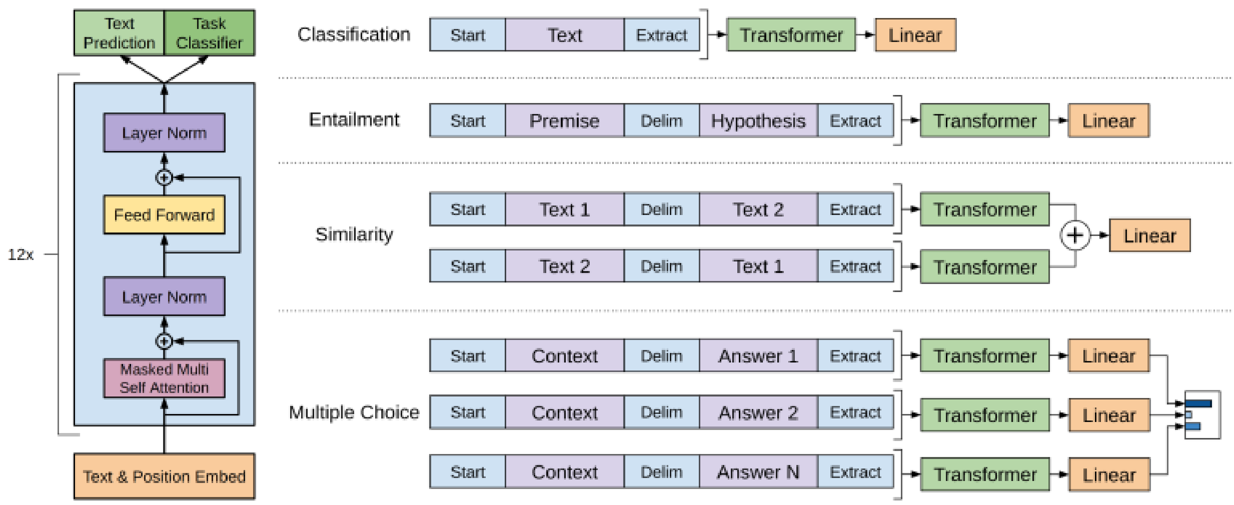


Figure 2.8: (left) Transformer architecture and training objectives used in this work. (right) Input transformations for fine-tuning on different tasks. We convert all structured inputs into token sequences to be processed by our pre-trained model, followed by a linear+softmax layer [5].

The foundational pre-training approach adopted achieved mainly by enhancing the model size, data volume and diversity, and extending the training duration. While the concept of in-context learning resembles , this research delves deeper into varying in-context learning settings. The intent is to determine how GPT-3 performs across these settings, especially when focusing on the reliance on task-specific data. These settings can be visualized as a spectrum, and four distinct points on this spectrum have been identified [1]:

- **Fine-Tuning (FT)**: This widespread technique modifies the weights of a pre-trained model using a task-specific supervised dataset, generally requiring thousands to hundreds of thousands of labeled instances. Its strengths lie in impressive benchmark performance. However, the downsides include the demand for extensive datasets for every task, risks of sub-par generalization, and reliance on potentially spurious training data features. Although GPT-3 isn't fine-tuned in this study, with the emphasis being on task-agnostic performance, it remains an intriguing avenue for future exploration [1].

- **Few-Shot (FS)**: In this approach, the model is conditioned with several task demonstrations during inference, but without any weight adjustments. A typical example might encompass a context and its corresponding completion. The "few-shot" method provides a handful of such context-completion examples, followed by a sole context, expecting the model to generate the completion. Advantages include significant reductions in task-specific data needs and decreased risk of narrow distribution learning. On the flip side, its results haven't matched fine-tuned models in performance [1].

- **One-Shot (1S)**: This mirrors the few-shot technique but is restricted to a single demonstration alongside a natural language task description. It's singled out because it often aligns with how tasks are relayed to humans, particularly in platforms like Mechanical Turk where a single task demonstration is customary [1].

The figure 2.9 illustrates the diference between , Fine-Tuning , Few-Shot and One-Shot

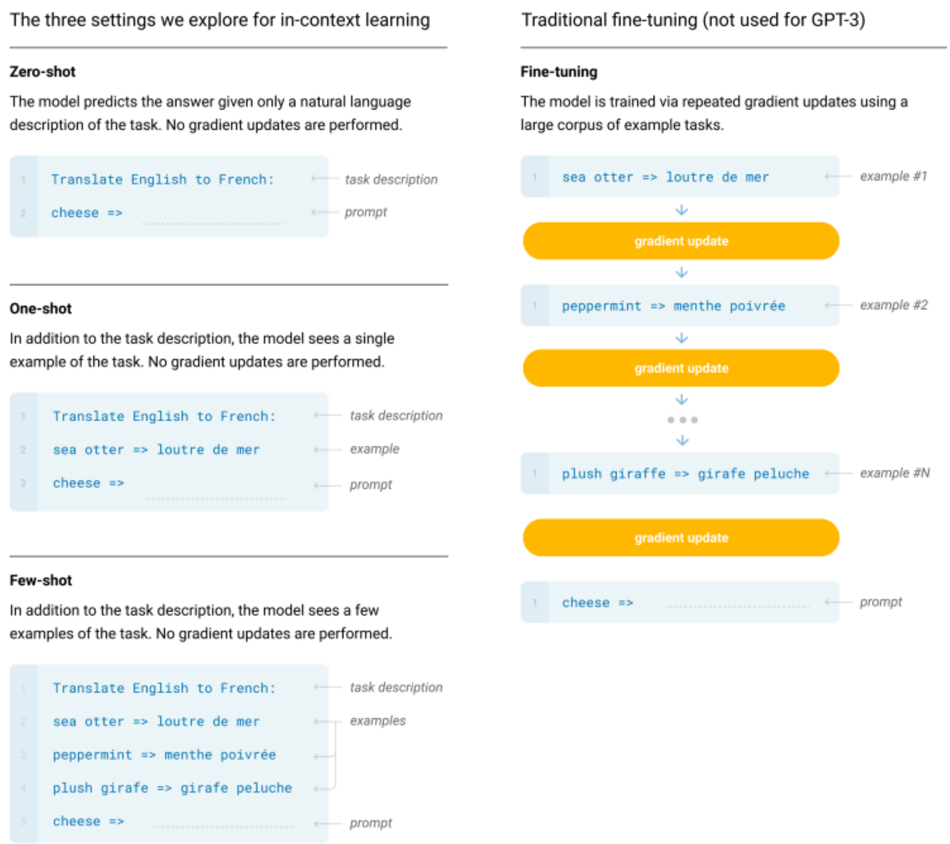


Figure 2.9: Zero-shot, one-shot and few-shot, contrasted with traditional fine-tuning. The panels above show four methods for performing a task with a language model – fine-tuning is the traditional method, whereas zero-, one-, and few-shot, which we study in this work, require the model to perform the task with only forward passes at test time [5].

## 2.2 Evaluation Metrics

### 2.2.1 BLEU SCORE

The BLEU (Bilingual Evaluation Understudy) score is a metric devised for evaluating the quality of machine-translated text compared to one or more reference translations. It examines the overlap of n-grams between the generated translation and the reference translation(s) to compute the score. The score ranges between 0 and 1, where 1 indicates a perfect match with the reference [4].

Given a candidate translation and a set of reference translations, the BLEU score is computed as:

$$\text{BLEU} = \text{BP} \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{2.1}$$

Where:

- BP is the brevity penalty. It ensures that shorter candidate translations don't get higher scores simply because they match fewer n-grams. It's given by:

  $$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-\frac{r}{c})} & \text{if } c \leq r \end{cases}$$

  where $c$ is the length of the candidate translation and $r$ is the effective reference corpus length [4].

- $w_n$ are the weights for each n-gram. Typically, for evaluating up to 4-grams, these weights are set to $\frac{1}{4}$ [4].

- $p_n$ is the precision for n-grams and is computed as the ratio of number of n-gram counts in both candidate and reference to the number of n-grams in the candidate [4].

A perfect BLEU score of 1 means the candidate translation matches a reference translation exactly [4].

### 2.2.2 ROUGE SCORE

The ROUGE (Recall-Oriented Understudy for Gisting Evaluation) score is a metric devised to evaluate the quality of summaries by comparing them to reference summaries. It's particularly popular in tasks like automatic text summarization to measure the overlap between the n-grams of the produced summary and a set of reference summaries.

One of the most commonly used versions of ROUGE is ROUGE-N, which measures the overlap of n-grams between the system and reference summaries. Mathematically, ROUGE-N is defined as:

$$\text{ROUGE-N} = \frac{\sum_{s \in \text{references}} \sum_{gram_n \in s} \min(\text{count}(gram_n, s), \text{count}(gram_n, \text{system}))}{\sum_{s \in \text{references}} \text{count}(gram_n, s)} \tag{2.2}$$

Where:

- $gram_n$ refers to the n-gram under consideration.
- $\text{count}(gram_n, s)$ is the count of the n-gram in summary $s$.
- system is the system-generated summary.
- references is the set of reference summaries.

Apart from ROUGE-N, there are other variants like ROUGE-L, which considers the longest common subsequence, and ROUGE-S, which measures skip-bigram statistics.

# Conclusion

In this chapter, we looked at some important models and methods used in modern text processing. We started by understanding the attention mechanism, which is a key part of many newer models. After that, we discussed different models like BERT, ALBERT, T5, and GPT, each with its own way of working with text. Finally, to know if these models are doing a good job, we learned about two ways to measure their performance: the BLEU and ROUGE scores. All in all, this chapter gives us a good foundation to understand how modern text models work and how to measure their success.

# Chapter 3

# Work Architecture and Implementation

# Introduction

Chapter 3 delves into the practical aspects of our project. Here, we explore the methods, tools, and processes adopted, giving a hands-on perspective of turning research into practical applications.

## 3.1 Design of Our research method

Our system's design is shown in Figure 3.1. When a user asks a question, we first check if it's similar to basic questions in the `intents.json` file. If it is, our PyTorch chatbot answers it, and that cycle ends. If not, tools like GPT, BERT, Albert, and T5 help process the question. After getting an answer, we compare it with correct answers from our data or from developers. If they're close, we update our data. After many questions, we wrap up with a summary table, ending that cycle.

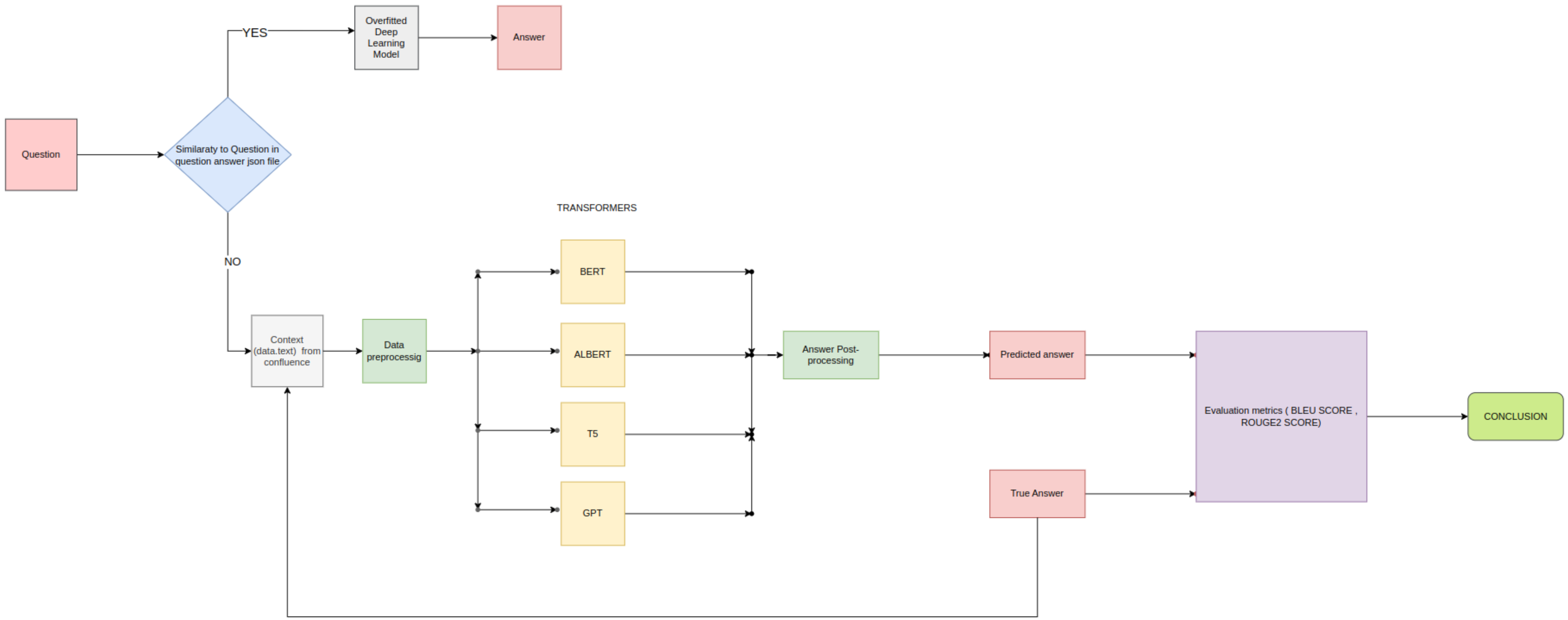


Figure 3.1: The overall architecture of the project.

## 3.2 Implementation

### 3.2.1 Hardware and Software Environments

**Jupyter Notebook :**

For the creation and manipulation our project , we chose Jupyter Notebook, which is an interactive development environment for the creation of documents containing code, visualizations and textual explanations.

Jupyter Notebook is particularly useful for data analysis, data manipulation and visualization creation.

**Hardware Environment :**

In order to carry out the work during the project, we used:

- Processor :Intel i5
- Memory: 16 GB
- Hard drive: 512 SSD
- Operating System: Windows 11

### 3.2.2 The Datasets

In our project, as illustrated in Figure 3.1, we rely on two primary datasets. The first is the `intents.json` file. This dataset contains keywords, questions, and answers designed for basic and day-to-day conversations. The second dataset consists of paragraphs from Atlassian Confluence. These paragraphs provide detailed specifications for the ADAD project, which is part of M&C's initiatives.

```
{
  "intents": [
    {
      "tag": "greeting",
      "patterns": [
        "Hi",
        "Hey",
        "How are you",
        "Is anyone there?",
        "Hello",
        "Good day"
      ],
      "responses": [
        "Hey :-)",
        "Hello, thanks for visiting",
        "Hi there, what can I do for you?",
        "Hi there, how can I help?"
      ]
    },
    {
      "tag": "goodbye",
      "patterns": ["Bye", "See you later", "Goodbye"],
      "responses": [
        "See you later, thanks for visiting",
        "Have a nice day",
        "Bye! Come back again soon."
      ]
    },
```

Figure 3.2: JSON data for simple chatbot

### 3.2.3 Examples of built chatbots

I've built different chatbots, and I want to share a few examples. First, I made a chatbot using PyTorch that can handle basic day-to-day conversations. To make sure the chatbot understands sentences correctly, I used a tool called NLTK to prepare and organize the data.

Then, I explored other advanced chatbot designs using the Transformers library from Hugging Face. With this library, I built three chatbots based on BERT, ALBERT, and T5 models. BERT helps the chatbot understand sentence context really well. ALBERT is like BERT but uses fewer resources, making it faster. T5 has a special way of changing all language tasks into a format where it changes one text into another.

I've included some figures next, which will show how each of these chatbots works and responds to what developers and clients ask.

```
ADAD Chatbot: Hello! I am ADAD Chatbot. Let's chat! (Type 'exit' to end the co
You: hello
ADAD Chatbot: Hi there, what can I do for you?
You: tell me a joke
ADAD Chatbot: Why did the hipster burn his mouth? He drank the coffee before i
You: thanks
ADAD Chatbot: Any time!
You: see you later
ADAD Chatbot: Have a nice day
You: quit
ADAD Chatbot: Goodbye! Have a great day!
```

Figure 3.3: Chatbot for basic and daily conversations

For the transformer-based chatbots, I introduced an additional feature. Alongside the answer provided by the transformer, I added an option to input the correct answer. If the correct answer is provided, the system automatically computes the ROUGE and BLEU scores by comparing the given answer with the correct one.

```
Question:
How the categorized disputes are uploaded into the database ?

Predicted Answer:
At specified intervals

True Answer:
The categorized disputes are uploaded into the database at specified intervals
BLEU score: 0.2727272727272727
ROUGE-2 score(F1_score): 0.33333333333333337
Results updated and saved.
Enter your question (type 'exit' to end): exit
```

Figure 3.4: Chatbot based on BERT

```
Enter your question (or type 'exit' to quit): who defines the accounting period duration
Enter the true answer or type 'skip': The accounting period duration is a parameter defined at i
ne.

convert squad examples to features: 100%|████████████████████████████████████████| 1/1 [
add example index and unique id: 100%|████████████████████████████████████████████████

Predicted Answer: The accounting period duration is a parameter defined at implementation time b
True Answer: The accounting period duration is a parameter defined at implementation time by the
BLEU Score: 0.9333333333333333
ROUGE Score: 1.0
Enter your question (or type 'exit' to quit): exit
Exiting...
```

Figure 3.5: Chatbot based on ALBERT

```
Enter your question (type 'exit' to end): what is the purpose of the Ai
Enter the true answer or type 'skip': Airline’s Referential Data have t
r) of the Airline in Adad, allowing it to better use adad
Predicted Answer: represent the ecosystem (payment and other) of the Ai
True Answer: Airline’s Referential Data have the purpose to represent t
allowing it to better use adad
BLEU Score: 0.5517241379310345
ROUGE Score: 0.6666666666666666
Results updated
Enter your question (type 'exit' to end): exit
```

Figure 3.6: Chatbot based on T5

## 3.3 Final results

I posed a vast number of questions related to ADAD specifications that developer or ariline can ask to chatbots built on T5, ALBERT, and BERT models. For every inquiry, I logged the question, the true answer, and the chatbot's predicted response into a CSV file. For each model, I visualized the outcomes by randomly selecting 10 entries from this file for examination.The following figures shows the results of each model sepratly.

| | Question | True Answer | Predicted Answer | BLEU Score | ROUGE-2 Score |
|---|---|---|---|---|---|
| 21 | When do the status "pending" apply | When no settlement has been linked to a sale | When no settlement has been linked to a sale | 1.000000 | 1.000000 |
| 8 | Give me example of the file name | 9429515695_epa_amex_20150818_0453389821883824_... | Ax_merchant_number_epa_amex_20150818 | 0.000000 | 0.307692 |
| 13 | What is unit test in a clean architecture | In clean architecture, a unit test is a type o... | A unit test is a type of automated test that i... | 0.615385 | 0.810811 |
| 19 | When the settlement is considered "settled" | When it has been successfully matched to one o... | When it has been successfully matched to one o... | 0.642857 | 0.761905 |
| 2 | What does etl stand for | Extract , transform , load | Extract, transform, and load | 0.818731 | 0.400000 |
| 25 | What is the purpose of the airline's referntia... | Airline's referential data have the purpose to... | Adad gl file monitoring dashboard helps airlin... | 0.034483 | 0.000000 |
| 18 | What is the purpose of matching rules module | The matching rules module enables the user to ... | Matching rules module enables the user to chan... | 0.888889 | 0.967742 |
| 9 | What format should the file name respect | Ax_merchant_number_epa_amex_$utc_{d}ate$_utc_time | Ax_merchant_number_epa_amex_$utc | 0.400000 | 0.769231 |
| 6 | What is the role of the etl configuration panel | The aim of the etl configuration panel is to p... | The aim of the etl configuration panel is to p... | 0.515152 | 0.744186 |
| 11 | What is scaling plan | A scaling plan is a set of rules and costs ass... | A scaling plan is a set of rules and costs ass... | 0.319149 | 0.491228 |

Figure 3.7: 10 Samples from T5's evaluation file

| | Question | True Answer | Predicted Answer | BLEU Score | ROUGE-2 Score |
|---|---|---|---|---|---|
| 21 | When do the status "pending" apply | When no settlement has been linked to a sale | When no settlement has been linked to a sale | 1.000000 | 1.000000 |
| 13 | What is unit test in a clean architecture | In clean architecture, a unit test is a type o... | A unit test is a type of automated test that i... | 0.615385 | 0.810811 |
| 22 | What happened when when the capture status of ... | The "n/a" (not applicable) status is assigned | "n/a" status is assigned | 0.545455 | 0.545455 |
| 9 | What format should the file name respect | Ax_merchant_number_epa_amex_$utc_{d}ate$_utc_time | Ax_merchant_number_epa_amex_$utc | 0.400000 | 0.769231 |
| 11 | What is scaling plan | A scaling plan is a set of rules and costs ass... | A scaling plan is a set of rules and costs ass... | 0.319149 | 0.491228 |
| 14 | Example of naming convention | Salesreport_{airlinecode}{salesreportid}{recep... | Salesreport_airlinecodesalesreportidreceptiondate | 0.000000 | 0.000000 |
| 7 | What is load test | Load test is the objective is to ensure that l... | Load test is the objective is to ensure that l... | 0.481481 | 0.631579 |
| 1 | What is sla | Is a contract between a service provider and i... | Service-level agreement (sla) is a contract be... | 0.300000 | 0.400000 |
| 23 | What does soc stand for | Summary of charge | Summary of charge | 1.000000 | 1.000000 |
| 18 | What is the purpose of matching rules module | The matching rules module enables the user to ... | Matching rules module enables the user to chan... | 0.888889 | 0.967742 |

Figure 3.8: 10 Samples from BERT's evaluation file

| | Question | True Answer | Predicted Answer | BLEU Score | ROUGE-2 Score |
|---|---|---|---|---|---|
| 71 | Who is responsible for transmitting the groupe... | The airline's officer | The airline's officer | 1.000000 | 1.000000 |
| 72 | What role does the airline's officer play in t... | THe airline's officer acts as the intermediaries | The airline's officer acts as the intermediaries | 1.000000 | 1.000000 |
| 7 | What is the type of testing preferred at adad | Black box testing (behavioral testing) | Black box testing | 0.428571 | 0.666667 |
| 66 | What role do the recokeys play in relation to ... | Common fields | Common fields | 1.000000 | 1.000000 |
| 25 | Why the end-to-end(e2e) tests are used | To test the full flow of the system, from the ... | To test the full flow of the system, from the ... | 1.000000 | 1.000000 |
| 45 | How does the airline define rules to link acco... | A dedicated accounting rule interface allows t... | A dedicated accounting rule interface | 0.238095 | 0.333333 |
| 37 | How do automatic task assignment methods work | Automatic task assignment methods do not need ... | Automatic task assignment methods do not need ... | 1.000000 | 1.000000 |
| 68 | What value does the settlement dispute managem... | It contributes to accurate financial reporting... | It contributes to accurate financial reporting... | 1.000000 | 1.000000 |
| 50 | How does the 'unaccounted events dashboard' as... | Help the user investigate the causes of the ac... | Investigate the causes of the accounting failu... | 0.769231 | 0.857143 |
| 0 | What is etl | A data integration process that combines data ... | NaN | 0.000000 | 0.000000 |

Figure 3.9: 10 Samples from ALBERT's evaluation file

Following our detailed analysis, where we visualized various samples, we synthesized the outcomes to provide a consolidated view. To achieve this, we compiled the results into a comprehensive table. This table showcases the performance of the three distinct models: BERT, T5, and ALBERT. For each model, we highlighted key metrics, including the average BLEU and ROUGE scores

The figure 3.10 showcases the final results of the three models :

| | Model name | AVG BLEU score | AVG ROUGE-2 score |
|---|---|---|---|
| 0 | ALBERT | 0.711513 | 0.777917 |
| 1 | BERT | 0.404006 | 0.373474 |
| 2 | T5 | 0.505599 | 0.602931 |

Figure 3.10: Final results

## 3.4 Special Use Case : Using GPT with Our Data

I got an API key from OpenAI and used it to build an app using python Langchain and OpenAI library. This app uses the GPT-3.5 Turbo model and works with the company's data from a .txt file. At first, I tested it using a basic terminal where I could type questions and get answers (see Figure 3.9). After that, I made a web version of the app using Streamlit, making it easier for users to interact with

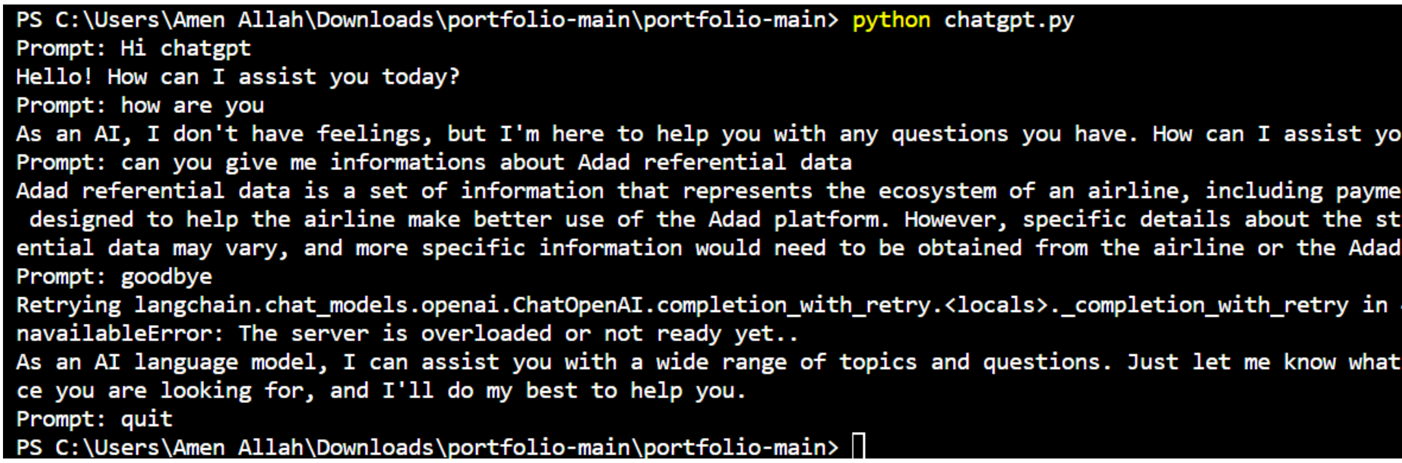


Figure 3.11: Testing the solution

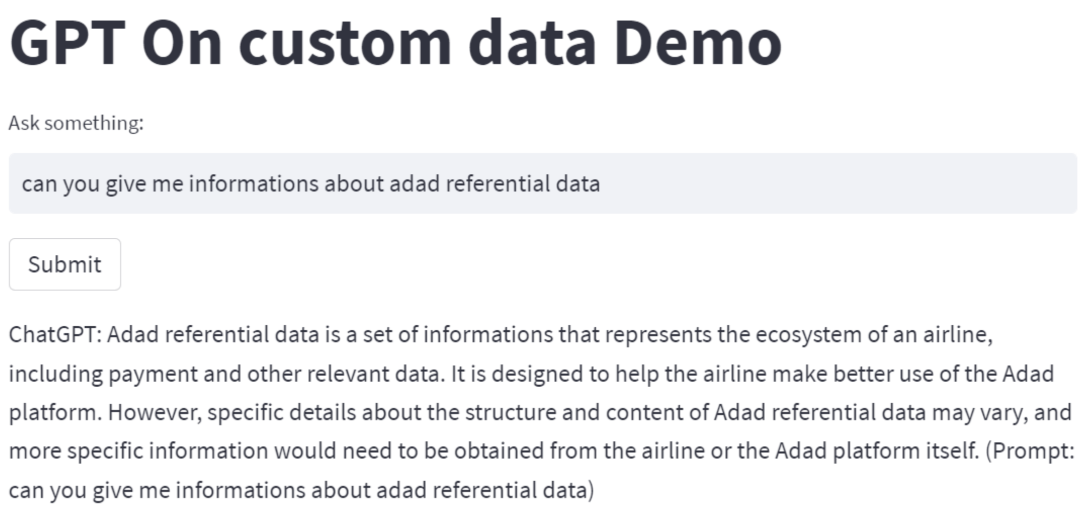


Figure 3.12: The solution deployed using streamlit

## Conclusion

Chapter 3 walked us through how I set up and ran our project. I talked about the research architectue , the data I used, and the chatbots I made. This chapter gives a clear picture of our hands-on work with AI.

# General Conclusion

Throughout this report, we embarked on a comprehensive journey exploring the intersection of large language models and their practical application within the company's ecosystem. We initiated our exploration with a deep dive into the company's goals, services, and the specific objectives of this internship. This foundational understanding directed our subsequent exploration into the latest advancements in language models.

We delved into the intricacies of attention mechanisms, scrutinized architectures like BERT, ALBERT, T5, and GPT, and explored the metrics that evaluate their performance. The latter stages of the report transitioned from theoretical knowledge to its application. We meticulously documented our research methods, the choices behind our datasets, and the practical challenges faced during implementation. The showcase of chatbots we built serves as a testament to the potential and adaptability of these models, especially when fine-tuned with custom data, as illustrated with our special use case of GPT.

Looking back, this project taught us a lot about what modern language models can and can't do. It showed us how they can be useful in real situations. As we move forward, what we learned will help shape our decisions about using AI in our business, aiming for tools that are smarter, easier to use, and more effective.

# References

[1] Tom Brown et al. "Language models are few-shot learners." In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[2] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." In: *arXiv preprint arXiv:1810.04805* (2018).

[3] Zhenzhong Lan et al. "Albert: A lite bert for self-supervised learning of language representations." In: *arXiv preprint arXiv:1909.11942* (2019).

[4] Kishore Papineni et al. "Bleu: a method for automatic evaluation of machine translation." In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002, pp. 311–318.

[5] Alec Radford et al. "Improving language understanding by generative pre-training." In: (2018).

[6] Colin Raffel et al. "Exploring the limits of transfer learning with a unified text-to-text transformer." In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 5485–5551.

[7] Ashish Vaswani et al. "Attention is all you need." In: *Advances in neural information processing systems* 30 (2017).