Politecnico di Torino

Corso di Laurea
A.a. 2023/2024
Sessione di Laurea Aprile 2024

Design, Implementation and Evaluation of a Chatbot for Accounting Firm: A Fine-Tuning Approach with Two Novel Dataset

Relatori:
Luca Ardito

Candidati:
Michele Basilico

Abstract

Artificial intelligence, particularly in the field of Chatbots, is fundamentally reshaping learning, communication, and work paradigms. This phenomenon has sparked growing interest among businesses, viewing Chatbots as a means to streamline processes, enhance customer support, and deliver increasingly efficient services. The proposed thesis work focuses on applying these tools within the legal, fiscal, and commercial sectors to meet the needs of an accounting firm, where managing a large volume of questions and information requires innovative solutions. The thesis aims to investigate, design, develop, and evaluate a Chatbot capable of providing quick, effective, and high-quality responses. To achieve this goal, an analysis of state-of-the-art modern Open-Source Large Language Models was initially conducted, followed by a fine-tuning process to specialize the base model. Finally, the results were analyzed and compared with models currently available on the market. To make possible model fine-tuning, two new datasets were created and introduced: ADE, containing questions and answers from the "Agenzia Delle Entrate" portal, and ITACA, containing synthetically generated questions and answers using information provided through articles and official documents. Specifically, ITACA was developed using a tool designed and implemented during the proposed work, named LLMDSGenerator.

Table of Contents

TABLE OF CONTENTS 4

1. INTRODUCTION 8

2. BACKGROUND 11

2.1 Basic introduction to Artificial Intelligence 12

2.1.1 AI, Machine Learning & Deep Learning 12

2.1.2 AI - State of the art 14

2.1.3 AI – The challenges to face up 15

2.2 Natural Language Processing 16

2.2.1 NLP: Goal and challenges 16

2.2.2 NLP Solution: Transformers 17

2.3 Large Language Model 19

2.3.1 LLM – What is it? 19

2.3.2 LLM – How it works? 19

2.3.3 LLM – Why a large language model? 21

2.3.4 LLM – Common Use Cases 21

2.4 AI for financial services 23

2.4.1 How AI can be used in finance and legal 23

2.4.2 The challenges of AI in finance 25

3. LARGE LANGUAGE MODEL DEVELOPMENT 27

3.1 Machine Learning Lifecycle 28

3.1.1 The step to build a ML 28

3.1.2 Selection and configuration of an architecture/model 29

3.1.3 Data Preparation 29

3.1.4 Training Process 30

3.1.5 Inference Process 31

3.1.6 Evaluation 32

3.1.7 Deploy in production 33

3.2 How to choose a model? 35

3.2.1 Model Size and capabilities 35

3.2.2 Performance and Latency 36

3.2.3 Pretraining Data and Knowledge Cutoff 37

3.2.4 FineTuning and Customizability 38

3.2.5 Open-source model & Licence 39

3.2.6 Accessibility and cost 39

3.2.7 Ethical and Responsible AI Considerations 40

3.3 Different approaches to train a model 41

3.3.1 Training from scratch 41

3.3.2 Training with Transfer Learning 42

3.3.3 Training with finetuning 42

3. DATASET 45

3.1 Dataset Construction 46

3.1.1 Challenges and Considerations: 46

3.1.2 Strategies to Overcome Challenges: 47

3.1.3 Dataset Choices 48

3.2 ADE (Agenzia Delle Entrate) DATASET 48

3.2.1 What is ADE Dataset 48

3.2.1 The process behind ADE dataset 50

3.2.3 Data Choices and Cleaning 53

3.3.4 ADE Samples 54

3.3 ITACA DATASET 55

3.3.1 What is the ITACA DATASET? 55

3.1.2 ITACA Building Process 56

3.1.3 LLMDSGenerator (Large Language Model DataSet Generator) 57

3.1.4 Data Choices and Cleaning 60

3.1.5 ITACA Dataset Samples 62

3.1.6 ITACA Dataset v2 63

5. METHODOLOGY 65

5.1 Selection of the Base Large Language Model 66

5.1.1 Context and exclusion: 66

5.1.2 State-of-the-arte LLM: 67
5.1.3 Base LLM Comparation: 72
5.1.4 LLM Italian Evaluation 74
5.1.4 Analysis of result & Final Choice: 78

5.2 Data Loading, Cleaning and Preprocessing 79
5.2.1 Loading and merging datasets: 79
5.2.2 Data Cleaning: 81
5.2.3 Data preprocessing: Format and tokenize the text. 83

5.3 Fine-Tuning Process with (Q)LoRA 86
5.3.1 LoRA and QloRA Introduction 87
5.3.2 How does LoRA works? 88
5.3.3 Make more efficient LoRA with QloRA 89

6. EXPERIMENTS, EVALUATION AND COMPARATION 91

6.1 Environments and dependencies 92
6.1.1 Environment and Power Computing 92
6.1.2 Hugging Face 92
6.1.3 Unsloth 94
6.1.4 WandB 95

6.2 Hyperparameters Tuning 95
6.2.1 Hyperameter Tuning Method 96
6.2.2 Standard Hyperparameters 97
6.2.3 Lora Adapter Hyperparameters 98

6.4 Experiments & Analysis: 99
6.4.1 Hyperparameter Proposed Configuration 99
6.4.2 Configuration #0 100
6.4.3 Configuration #1 102
6.4.4 Configuration #3 104
6.4.5 Configuration #4 104
6.4.6 ITACA Finetuning 106

6.5 Inference & Comparation 107
6.5.1 Society 108
6.5.2 Tax collection 110
6.5.3 Professions and work regimes 112

6.5.4 Fisco Generale 114
6.5.5 Out Of Scope 116
6.5.6 Human evaluation 118

7. ITACA DEPLOYMENT 119

7.1 Deployment of LLM Models 120

7.2 Required Hardware and Architecture 121

7.3 Overview of Deployment Options 121

7.4 Hugging Face for LLM Deployment 122

7.5 Gradio UI Experience 125

8. RESULT 128

8.1 A look back to the project 128

8.2 Limits and challenges 129

8.3 Possible Improvements 130

BIBLIOGRAPHY 132

1. Introduction

In recent years, conversational agents, commonly known as chatbots, have emerged as powerful tools across various domains, fundamentally altering the way people interact with technology. No longer limited to scripted responses, modern chatbots leverage sophisticated algorithms and artificial intelligence techniques to understand and respond to user queries in a manner that mimics human conversation. Their impact has been profound in a lot of sectors including customer service, healthcare, and education, reshaping conventional paradigms. The rise of large language models (LLMs) like GPT-4 by OpenAI, LLama2 by Meta, Gemini by Google, and others has amplified this trend, swiftly embedding chatbots into the fabric of daily life, research, and professional environments.

However, there are some fields requires paying particular attention such as finance, tax and, legal ones. Accounting firms are trying to grapple with manual tasks and client interactions, seeking ways to optimize operations and enhance experiences. In this world, it's crucial to handle data with utmost care due to its sensitivity and the potential for creating problems and uncomfortable situations. The legal and financial sectors face unique challenges in this regard. Unlike many other industries where a one-size-fits-all approach might suffice, the complexity of legal and financial matters demands a more tailored solution. Each country has its own distinct set of laws, regulations, financial and business practices, bureaucratic processes, and legal nuances. Therefore, a generic model that works well across languages may not be sufficient for these sectors. Instead, it's essential to customize the model to the specific area and country where it will be utilized. This entails going beyond a basic model to create a highly specialized version tailored to the legal and financial landscape of the target country. By taking this extra step, organizations can ensure compliance with local regulations, address unique business practices, and navigate bureaucratic timelines effectively. This level of customization is crucial for maintaining the integrity and reliability of systems operating in these sensitive sectors.

The problem to specialize a model to a specific and smaller target is not new and has been faced successfully in the last year. Starting from an approach like transfer learning a modern technique has been explored named FineTuning. While transfer learning occurs when we use knowledge that was gained from solving one problem and apply it to a new but related problem, *fine-tuning* is the process of taking a pretrained machine learning model and further training it on a smaller, targeted data set. The aim of fine-tuning is to maintain the original capabilities of a pretrained model while adapting it to suit more specialized use cases. This approach is especially beneficial when computational resources are limited, or relevant data is scarce.

This thesis addresses this need by focusing on the development and implementation of a conversational agent specialized in Italian legal and tax contexts for an accounting firm. To make it available, the thesis analyses the state-of-art of LLM, the different approach to finetune them, realize and define dataset and the flow to follow in order to achieve the best result and performance. Also, another issues need to be taken in consideration. While a standard LLM focuses on global and available data, in legal and tax field is necessary to explore a set of private data and personal documents. So, it's mandatory explore not just the technique to finetune the model, but also trying to understand how to make it possible without give private data out of our hand.

A fundamental obstacle in creating such a system lies in the scarcity of relevant training data, especially specific to Italian legal and tax conversations. To overcome this hurdle, two novel datasets tailored to these domains have been meticulously constructed. In pursuit of this goal, innovative approaches have been explored, including the release of a new tool called LLMDSGenerator, which combines the power of modern LLMs with web scraping techniques to generate synthetic but affordable datasets, such as the ITACA Dataset. Additionally, available tools have been leveraged to create another dataset named ADE.

To better understand all the work done and process used it's fundamental know all the terms and state of art in AI field. This is the goal of the first chapter. To lay the groundwork for an effective AI-driven assistant, understanding various types of chatbots, design methodologies, and best practices is crucial, as explored in the second chapter. Subsequently, important decisions regarding model choices and algorithms are discussed in the third chapter. The methodology and approach used for fine-tuning the model are then detailed, with results explored and analyzed in the fourth chapter. Finally, we will see all the achieved result and make the comparison with state-of-the-art LLM.

In summary, through the utilization of state-of-the-art natural language processing techniques and machine learning algorithms, this thesis aims to advance conversational agent technology in the specific contexts of Italian law and taxation. By developing novel datasets and implementing innovative approaches, it seeks to pave the way for more effective and user-friendly solutions tailored to the unique needs of Italian-speaking individuals within legal and tax frameworks.

2. Background

2.1 Basic introduction to Artificial Intelligence

The world today is undergoing a period of technological change unlike any seen in many years. Artificial Intelligence is not a new concept; we just need to think that the discipline was born in 1956. So, what has made AI successful these days?

This credit is certainly due to the development that this discipline has seen in recent years with the advance of many tools directly available to the public, such as ChatGPT, capable of automatically generating texts, or StableDiffusion, capable instead of generating images and videos. Thanks to these tools, the discipline, and generally AI, has entered everyday usage, and try to solve the common tasks of daily routine without these tools may seems expensive and out-of-date.

In this paragraph, we will briefly introduce all the concepts related to the world of AI that we need to know before delving into the thesis objective.

2.1.1 AI, Machine Learning & Deep Learning

Artificial Intelligence (AI) has witnessed unprecedented growth and development in recent years, revolutionizing various domains ranging from healthcare and finance to manufacturing and entertainment. AI and machine learning are often used interchangeably, but it's important to recognize that machine learning constitutes a subset of the broader category of AI. Within the realm of AI systems, several key concepts contribute to their functionality, including machine learning, deep learning, neural networks, computer vision, and natural language processing.

Artificial Intelligence characterises a paradigm within computer science aimed at creating machines capable of replicating tasks requiring human-like intelligence. This encompasses a broad spectrum of activities, spanning from problem-solving and pattern recognition to decision-making and adaptive behaviour. Examples range from smart assistants like Alexa to advanced applications such as self-driving cars.

Machine learning is only a branch of artificial intelligence, focuses on leveraging data and algorithms to simulate human learning processes, thereby enhancing accuracy. Technological advancements have led to impactful applications like Netflix's recommendation engine and autonomous vehicles. It serves as a vital component of data science, employing statistical methods to train algorithms for classification, prediction, and insights, thereby influencing decision-making across various domains.

Machine learning encompasses four main categories: **supervised**, **unsupervised**, and **semi-supervised learning**, along with **reinforcement learning**, each with distinct applications. Supervised learning utilizes labeled datasets to predict outcomes, while unsupervised learning clusters unlabeled data. Semi-supervised learning employs a smaller labeled set to guide classification, whereas reinforcement learning learns through trial and error, generating recommendations or policies.

Common machine learning algorithms include **neural networks**, linear regression, logistic regression, clustering, decision trees, and random forests. Real-world applications span from speech recognition and customer service chatbots to computer vision, recommendation engines, automated stock trading, and fraud detection.

Deep Learning represents a transformative paradigm within machine learning, showcasing its efficacy in handling diverse data resources with minimal human intervention while achieving superior accuracy compared to traditional methods. At the core of deep learning are neural networks, inspired by the intricate interactions of human brain neurons, enabling data processing through multiple iterations to uncover intricate features and make informed determinations.

Several significant variants exist within Deep Learning, such as **Convolutional Neural Networks** (CNNs), tailored for perceptual tasks like image processing. CNNs interpret pixel collections and identify distinctive features within hidden layers, facilitating image classification based on learned attributes.

Recurrent Neural Networks (RNNs) feature loop connections within their structure, allowing data to move in both forward and backward directions through previous layers. RNNs excel in tasks like sentiment prediction or sequence endings, making them invaluable for handling extensive sequences of text, speech, or images. For instance, in fraud detection within the banking sector, RNNs analyze transactional behavior history to enhance fraud detection beyond traditional methods.

Understanding the characteristics and applications of these artificial neural network variants is crucial as we advance in the development of Large Language Models (LLMs). It underscores the significance of deep learning in enhancing natural language processing and highlights the transformative potential of LLMs across diverse application scenarios.

2.1.2 AI - State of the art

Artificial Intelligence (AI) has seen significant advancements in recent years, with state-of-the-art technologies pushing the boundaries of what machines can achieve. Here are some key areas where AI has made remarkable progress:

1. Natural Language Processing (NLP): NLP has seen significant advancements with models like GPT-4, which has outperformed other Language Learning Models (LLMs) on various benchmarks. These models have been successful in tasks such as language translation, question answering, and text generation.

2. Computer Vision: Computer vision has also seen substantial progress with models excelling in tasks such as semantic segmentation, image classification, and object detection. These models have been used in various applications, from autonomous vehicles to medical imaging.

3. Reinforcement Learning: Reinforcement Learning (RL) has been a key area of focus, with models being used for tasks ranging from game playing to autonomous navigation. RL models learn by interacting with their environment, making them particularly useful for tasks where explicit supervision is not possible.

4. Generative AI: Generative models have also gained popularity, with applications in areas such as video, text, and code generation. These models can generate new data that is similar to the training data, opening up possibilities for creative applications.

5. Graph Learning: Graph learning models have been used for tasks such as link prediction, node classification, and graph embedding. These models are particularly useful for tasks that involve relational data, such as social network analysis.

6. Safety and Ethics in AI: As AI models become more capable, there is an increasing focus on safety and ethics. This includes efforts to mitigate the risks posed by highly-capable future AI systems, as well as addressing issues related to fairness and transparency.

7. Computational Efficiency: There is also a growing emphasis on making AI models more computationally efficient. This includes efforts to reduce the computational resources required to train and deploy AI models, which is particularly important given the environmental impact of large-scale AI training.

2.1.3 AI – The challenges to face up

Despite the remarkable strides AI has made, it faces significant challenges that necessitate careful consideration and responsible development. Some of the most pressing challenges confronting AI today encompass:

Data Bias: Data bias represents a critical obstacle in the AI landscape, with far-reaching consequences. Biased data used for training can propagate discrimination and inequality in algorithmic decision-making processes. This issue is particularly pronounced in domains like facial recognition technology, where biased datasets can result in erroneous identifications and unjust repercussions. Addressing data bias requires meticulous dataset curation, robust bias detection mechanisms, and ongoing vigilance to ensure fairness and impartiality in AI applications.

Ethical Concerns & Human-AI Collaboration: AI's expansion into various sectors raises ethical concerns that demand thorough examination. Privacy, autonomy, and accountability are at the forefront of these concerns. For example, when AI systems play a role in critical decisions such as hiring or loan approvals, the potential for unfair outcomes arises if the algorithms harbor biases or flaws. A responsible approach to AI involves fostering transparency and accountability, ensuring that the algorithms' inner workings are understandable and explainable to humans. Human-AI collaboration necessitates user-friendly interfaces and clear communication channels, enabling individuals to comprehend AI decisions and, if necessary, intervene to ensure just outcomes.

Addressing these challenges is essential to harness AI's potential for positive transformation while mitigating its risks. Striking a balance between innovation, ethical considerations, and effective human-AI interaction is crucial. As AI continues to integrate into our lives, it is our collective responsibility to navigate these challenges with a commitment to fairness, inclusivity, and the betterment of society. By prioritizing transparency, fairness, and ethical use, we can unlock AI's potential as a powerful tool for progress, innovation, and positive impact while mitigating the risks it poses.

2.2 Natural Language Processing

Natural language processing, or NLP, combines computational linguistics—rule-based modelling of human language—with statistical and machine learning models to enable computers and digital devices to recognize, understand and generate text and speech. It's at the core of tools we use every day – from translation software, chatbots, spam filters, and search engines, to grammar correction software, voice assistants, and social media monitoring tools.

2.2.1 NLP: Goal and challenges

The aim of NLP tasks is not only to understand single words individually, but to be able to understand the context of those words. NLP is used to understand the structure and meaning of human language by analyzing different aspects like syntax, semantics, pragmatics, and morphology. Then, computer science transforms this linguistic knowledge into rule-based, machine learning algorithms that can solve specific problems and perform desired tasks.

Using text vectorization, NLP tools transform text into something a machine can understand, then machine learning algorithms are fed training data and expected outputs (tags) to train machines to make associations between a particular input and its corresponding output. Machines then use statistical analysis methods to build their own "knowledge bank" and discern which features best represent the texts, before making predictions for unseen data (new texts).

The following is a list of common NLP tasks, with some examples of each:

- **Classifying whole sentences**: Getting the sentiment of a review, detecting if an email is spam, determining if a sentence is grammatically correct or whether two sentences are logically related or not
- **Classifying each word in a sentence**: Identifying the grammatical components of a sentence (noun, verb, adjective), or the named entities (person, location, organization)
- **Generating text content**: Completing a prompt with auto-generated text, filling in the blanks in a text with masked words
- **Extracting an answer from a text**: Given a question and a context, extracting the answer to the question based on the information provided in the context

- **Generating a new sentence from an input text**: Translating a text into another language, summarizing a text

NLP isn't limited to written text though. It also tackles complex challenges in speech recognition and computer vision, such as generating a transcript of an audio sample or a description of an image.

2.2.2 NLP Solution: Transformers

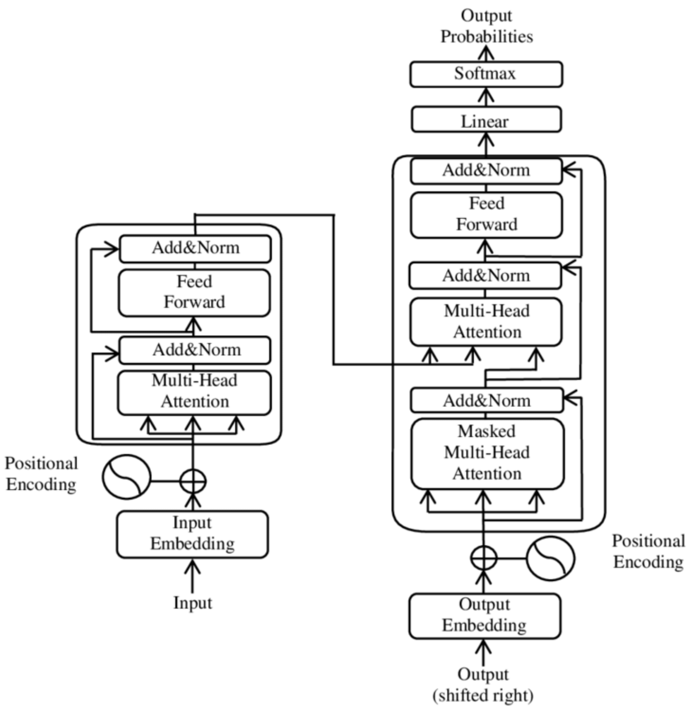
The Transformer, introduced by Vaswani et al. in their groundbreaking 2017 paper "Attention Is All You Need" (Ashish Vaswani, 2017) stands as a pivotal architecture in artificial intelligence, particularly in the realm of natural language processing (NLP). Its profound impact on how machines comprehend and generate human language has made it a cornerstone in various AI contests and competitions, owing to its remarkable performance and versatility.

At its heart lies the self-attention mechanism, also known as scaled dot-product attention, which enables the model to assess the significance of individual words within a sentence or sequence. Unlike traditional recurrent neural network (RNN) or convolutional neural network (CNN) architectures, the Transformer effectively captures long-range dependencies, thus overcoming previous limitations.

In earlier sequence-to-sequence tasks, such as neural machine translation, RNN-based encoder-decoder models struggled with retaining information from lengthy sequences. This issue stemmed from their reliance on the last hidden state, which often overlooked crucial elements at the beginning of the sequence. To address this, the attention mechanism was introduced, allowing the decoder to access all encoder states and prioritize relevant information for each output element prediction.

However, this method posed its own challenge: processing sequences one element at a time, leading to inefficiencies, especially with large datasets. The Transformer overcomes this hurdle by employing self-attention to extract features for each word, without relying on recurrent units. This design not only enhances parallelization but also boosts efficiency.

Comprising encoder and decoder components, each equipped with multiple layers of self-attention mechanisms and feed-forward neural networks, the Transformer architecture excels in capturing contextual relationships between words during encoding. In the decoding phase, it generates output sequences by attending to the encoded representations, predicting each token progressively.



One of its standout features is parallelizability, facilitating faster training compared to sequential models like RNNs. Moreover, Transformers excel at capturing dependencies across long distances within input sequences, making them ideal for tasks like language translation, text summarization, and question answering.

In AI contests, the Transformer's exceptional performance on benchmark NLP tasks, including machine translation, sentiment analysis, and language modelling, has made it a top choice among participants. Its ability to generalize across languages and domains, combined with its scalability and ease of implementation, has solidified its position as the preferred architecture for addressing diverse NLP challenges.

2.3 Large Language Model

2.3.1 LLM – What is it?

A large language model is a type of artificial intelligence (AI) model designed to understand and generate human language. Is a specific type of transformer that has been trained on vast amounts of text data.

A language model takes in a sequence of words as input and predicts the probability distribution of the next word or sequence of words. It learns from large amounts of text data, such as books, articles, websites, and other sources, to capture the statistical patterns and relationships between words. By analyzing these patterns, a language model can generate human-like text based on the context and input it receives.

Now, a large language model, like GPT-3 (which is one of the largest language models to date), refers to a language model with an extensive number of parameters. Parameters are the internal variables that the model uses to make predictions and store information. The more parameters a model has, the more complex and nuanced its understanding of language can be.

Large language models like GPT-3 are trained on massive datasets containing billions of sentences to develop a deep understanding of grammar, syntax, and semantics. This training allows them to generate text that is remarkably coherent and contextually relevant. These models can be fine-tuned for specific tasks, such as translation, summarization, question-answering, and more.

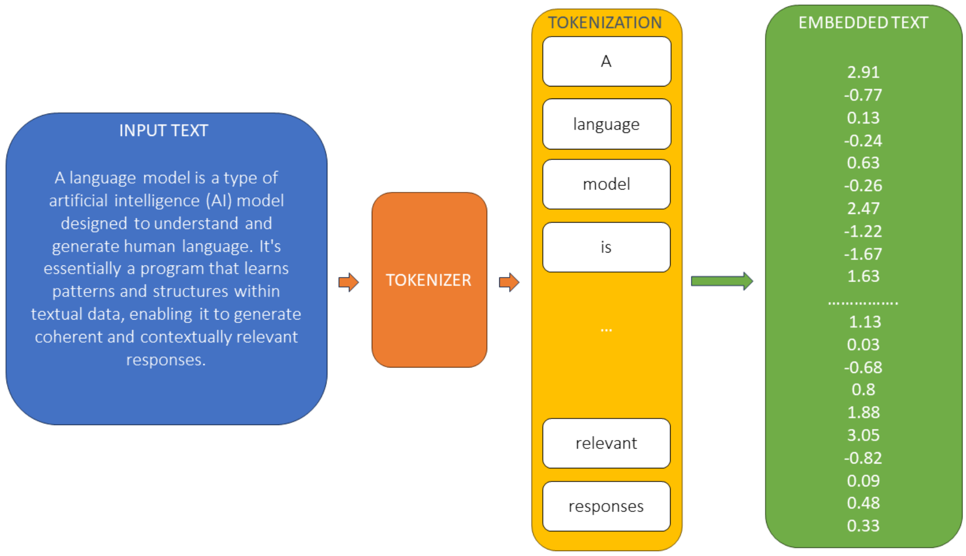
2.3.2 LLM – How it works?

As humans, we perceive text as a collection of words and documents structured into chapters, sections, and paragraphs. However, for computers, text is essentially a series of characters. To bridge this gap and enable machines to truly comprehend text, a revolutionary model based on recurrent neural networks can be developed. This model processes each word or character sequentially, generating output once the entire input text has been processed. While this approach works well, it sometimes "forgets" information from the beginning of the sequence when it reaches the end.

In 2017, Vaswani et al. introduced a game-changing paper titled "Attention is All You Need," presenting the transformer model, built upon the innovative attention mechanism. In contrast to recurrent neural networks, the attention mechanism permits the model to grasp an entire sentence or even a paragraph at once, rather than processing one word at a time. This distinctive feature empowers the transformer model to better understand the context of words, leading to its widespread adoption in state-of-the-art language processing models.

To process text using a transformer model, the initial step involves tokenization, transforming the text into a sequence of words or subwords. These tokens are then converted into numerical embeddings, creating vector-space representations that preserve their underlying meaning. The transformer's encoder subsequently transforms these token embeddings into a comprehensive context vector.

Consider the following example of a text string, its tokenization, and the vector embedding.



The context vector encapsulates the essence of the entire input. Leveraging this vector, the transformer decoder generates output based on contextual clues. For instance, you can provide the original input as a clue and prompt the transformer decoder to produce the next word that naturally follows. Repeating this process allows you to generate an entire paragraph, starting from an initial sentence.

This approach, known as autoregressive generation, underpins the workings of large language models. These models, based on the transformer architecture, can handle lengthy input texts, boasting a large context vector capable of tackling complex concepts, and featuring numerous layers in both the encoder and decoder.

2.3.3 LLM – Why a large language model?

Historically, AI models had been focused on perception and understanding.

However, large language models, which are trained on internet-scale datasets with hundreds of billions of parameters, have now unlocked an AI model's ability to generate human-like content.

Models can read, write, code, draw, and create in a credible fashion and augment human creativity and improve productivity across industries to solve the world's toughest problems.

The applications for these LLMs span across a plethora of use cases. For example, an AI system can learn the language of protein sequences to provide viable compounds that will help scientists develop groundbreaking, life-saving vaccines.

Or computers can help humans do what they do best—be creative, communicate, and create. A writer suffering from writer's block can use a large language model to help spark their creativity.

Or a software programmer can be more productive, leveraging LLMs to generate code based on natural language descriptions.

2.3.4 LLM – Common Use Cases

Large Language Models (LLMs) have a wide range of use cases due to their ability to understand and generate human-like text based on the extensive training data they've been exposed to. Here's a detailed presentation of some key LLM use cases:

1. **Text Generation (Generative Use Cases):**

    - **Creative Writing**: LLMs can generate creative pieces of writing, including poetry, short stories, and even novels. They can assist writers by providing inspiration and generating content.

    - **Content Creation**: LLMs can be used to automatically generate articles, blog posts, and other forms of written content, which can be valuable for content marketers and publishers.

- **Code Generation**: LLMs can generate code snippets in various programming languages based on high-level descriptions or requirements, aiding software developers.

- **Data Augmentation**: LLMs can generate additional training data for machine learning models, helping improve the performance of various AI applications.

2. **Natural Language Understanding (NLU Use Cases):**

   - **Chatbots**: LLMs power conversational AI by understanding user queries and generating human-like responses. They're used in customer support, virtual assistants, and more.

   - **Sentiment Analysis**: LLMs can determine the sentiment (positive, negative, neutral) of text, which is valuable for understanding customer opinions and market trends.

   - **Named Entity Recognition**: LLMs can extract specific information, such as names, locations, dates, and organizations, from text, which is useful in various data analysis tasks.

   - **Language Translation**: LLMs excel at translating text from one language to another, enabling real-time language translation in various applications.

3. **Text Summarization and Information Retrieval:**

   - **Text Summarization**: LLMs can generate concise summaries of long articles or documents, which is valuable for quick information retrieval and content curation.

   - **Search Engines**: LLMs can improve search engine results by understanding user queries better and retrieving more relevant documents or web pages.

4. **Text Classification and Sentiment Analysis:**

   - **Topic Classification**: LLMs can classify documents or text snippets into predefined categories, aiding in content organization and information retrieval.

   - **Spam Detection**: LLMs can be used to identify spam emails, comments, or other types of unwanted content, enhancing cybersecurity.

5. **Personalization and Recommendations:**

    - **Personalized Recommendations**: LLMs can analyze user preferences and behaviors to make personalized product, content, or service recommendations.

    - **Content Tagging**: LLMs can automatically tag and categorize content, making it easier to organize and recommend relevant items to users.

6. **Academic and Scientific Research:**

    - **Research Assistance**: LLMs can assist researchers in finding relevant papers, summarizing research findings, and generating hypotheses.

7. **Accessibility and Inclusivity:**

    - **Text-to-Speech**: LLMs can convert written text into spoken words, making content accessible to visually impaired individuals.

    - **Language Generation for Non-Native Speakers**: LLMs can help non-native speakers generate more fluent and accurate text in a given language.

These are just a few examples of the many use cases for Large Language Models. The versatility and capabilities of LLMs continue to expand as research and development in this field progress.

2.4 AI for financial services

2.4.1 How AI can be used in finance and legal

AI is used extensively in both the finance and legal sectors, leveraging its capabilities to enhance efficiency, accuracy, and decision-making. As concern as AI in Finance, AI can be used to the following task:

1. **Algorithmic Trading:** AI is employed to develop sophisticated trading algorithms that analyze vast amounts of market data in real-time. These algorithms can make rapid

trading decisions based on predefined criteria, leading to increased efficiency and potentially higher returns.

2. **Risk Assessment:** AI models can analyze historical data to assess risk factors and predict potential market fluctuations. This is crucial for portfolio management and investment decision-making.

3. **Credit Scoring:** AI algorithms can assess creditworthiness by analyzing various data points, leading to more accurate and consistent credit scoring models.

4. **Fraud Detection:** AI is used to detect unusual patterns in financial transactions that could indicate fraudulent activities. This helps financial institutions protect themselves and their customers from fraudulent behavior.

5. **Customer Service:** AI-powered chatbots and virtual assistants are used to handle customer inquiries, provide basic financial advice, and assist with account management.

6. **Personalized Financial Services:** AI analyzes customer preferences, spending habits, and financial goals to provide personalized investment advice and financial planning.

Also, in Legal Sector AI comes to help for a lot of tasks:

1. **Document Review:** AI can review and analyze large volumes of legal documents quickly, which is particularly useful in tasks such as due diligence, contract review, and e-discovery during litigation.

2. **Legal Research:** AI tools can assist lawyers in finding relevant case law, statutes, and legal precedents, helping them save time and improve the quality of their legal research.

3. **Predictive Analytics:** AI can predict case outcomes based on historical data, helping lawyers assess the viability of a case and potentially settle disputes outside of court.

4. **Contract Analysis:** AI can extract and analyze key information from contracts, ensuring compliance and identifying potential risks.

5. **Natural Language Processing (NLP):** NLP-based AI tools can help automate the drafting of legal documents, generate client communication, and assist with regulatory compliance.

6. **Administrative Tasks:** AI can automate administrative tasks in law firms, such as appointment scheduling, document organization, and client communication, freeing up time for lawyers to focus on more complex legal work.

2.4.2 The challenges of AI in finance

AI has brought transformative possibilities to the realms of finance and law, promising efficiency, accuracy, and innovation. However, along with these benefits, significant challenges need careful consideration.

Both the financial and legal sectors operate within intricate regulatory frameworks. Implementing AI systems that adhere to these rules and stay current with evolving laws can be a daunting task. Ensuring that AI-driven decisions align with legal requirements while navigating a dynamic landscape is a significant challenge.

In addition, AI relies heavily on data, and in these sectors, data privacy is paramount. Financial and legal information is sensitive, and maintaining strict confidentiality is vital. AI systems must be designed to protect this data from breaches, unauthorized access, and misuse.

AI systems can also inherit biases from the data they're trained on, leading to unfair outcomes. In finance and law, where impartiality is crucial, biased algorithms can perpetuate or even exacerbate inequalities. Recognizing and mitigating these biases is essential to ensure fairness.

AI algorithms, particularly in complex deep learning models, can be challenging to interpret. This lack of transparency can be problematic, especially in sectors where clear explanations for decisions are required. Striking a balance between powerful predictive capabilities and understandable outcomes is a challenge.

Introducing AI into well-established industries can face resistance. Traditional practices, longstanding norms, and concerns about job displacement can hinder the adoption of AI-driven solutions. Convincing stakeholders of the benefits and providing training to work alongside AI are essential for successful integration.

AI decisions might raise ethical dilemmas, especially in finance and law, where human judgment has significant implications. Determining the right balance between automated decision-making and human oversight is a complex challenge that requires careful thought.

AI systems can sometimes behave unpredictably, leading to unforeseen risks. In high-stakes sectors like finance and law, these risks can have severe consequences. Thorough testing, monitoring, and contingency planning are necessary to mitigate such unexpected issues.

In overcoming these challenges, collaboration between experts in AI, finance, and law is crucial. Striking the right balance between technological advancement, regulatory compliance, and ethical considerations is essential to harness the full potential of AI while safeguarding the integrity and fairness of these industries.

3. Large Language Model Development

3.1 Machine Learning Lifecycle

3.1.1 The step to build a ML

The development of ML models and their delivery to the user is governed by the Machine Learning life cycle. It is a process that involves the preparation of data, training (building) models, and deploying them. While it enables businesses to acquire value, it aids them in managing their resources. These resources could range from business assets like customer data and capital to human resources like data scientists, ML engineers, and DevOps collaborating to make this process successful.

The data science development life cycle consists of three main stages: Data preparation, modelling and deployment.

Before this cyclic process commences, businesses need to define the problem they want to solve, create a roadmap, set objectives, and metrics to measure success or failure. It could be customer segmentation for their coffee business using K-means clustering to increase the consumer conversion rate or recommendation systems to enable customers easily find what they may want to buy on their site. All these have to be figured out so that it creates a clear direction for the teams involved.

The ML project life cycle can generally be divided into three main stages: data preparation, model creation, and deployment. All three of these components are essential for creating quality models that will bring added value to your business. It is called a *cycle* because, when properly executed, the insights gained from the existing model will direct and define the next model to be deployed.

Machine learning works in two main phases: training and inference. In the training phase, a developer feeds their model a curated dataset so that it can "learn" everything it needs to about the type of data it will analyze. Then, in the inference phase, the model can make predictions based on live data to produce actionable results.

3.1.2 Selection and configuration of an architecture/model

Begin by selecting a *baseline* architecture. This should be a relatively simple model, which is expected to have solid results with minimal effort. This model can later be compared to the more complex models that are trained later. You may want to start with vanilla classical ML solutions (e.g. logistic regression, xgboost) when possible, as they require minimal training resources and experimentation.

Later on, you may want to experiment with more complex DL architectures, ensembles, complex feature engineering, and feature selection. These methods will require more experimentation to find what best matches the problem you attempt to solve. Training these could be quite expensive, so limiting the space to explore by starting from well-established settings is a good idea.

3.1.3 Data Preparation

Training machine learning models is heavily reliant on high-quality data. The data used must be accurate, devoid of inconsistencies, and directly relevant to the intended task the model is meant to excel at. Ensuring proper data preparation is a critical step in the ML development process, as it can significantly reduce the need for extensive model debugging later on. This preparatory work is typically accomplished through a structured data pipeline, which comprises a series of data processing stages starting from data collection to its eventual storage in a designated repository such as a data lake or warehouse, tailored to the specific project requirements.

The primary components of a comprehensive data preparation process include the following:

1. **Data Collection and Labelling:** In ML, the complexity of the task at hand often dictates the amount of data required. Whenever feasible, it's advantageous to seek existing datasets that align with your needs, as generating new datasets can be resource-intensive. Even if existing data doesn't align perfectly with your target task, considering transfer learning approaches can help mitigate the need for an extensive dataset (a popular strategy in Natural Language Processing, for instance). When you must create your own dataset, certain factors should be taken into account, such as whether you can use "natural" data and annotate it, or if a synthetic dataset is necessary. Additionally, if data labeling requires domain expertise, outsourcing to platforms like Amazon Mechanical Turk can be a cost-effective solution.

2. **Data Augmentation:** When data scarcity is an issue, data augmentation techniques can be employed to expand the dataset. This involves applying automated alterations to the data. For instance, rotating an image of a cat still preserves the essence of a cat image. More advanced augmentation may involve altering labels as well. For example, in sentiment analysis, introducing negations to a positive movie review could transform the label to "negative."

3. **Data Cleaning:** Datasets commonly exhibit missing values, incorrect data types, or values that fall outside the expected range. Similar to the shifting of an entry in a spreadsheet, such issues can disrupt the integrity of the dataset. Removing redundant features also aids the training process. While data cleaning can be labor-intensive, proper implementation and automation significantly enhance data quality with minimal effort.

4. **Data Validation:** The data validation process is crucial for ensuring the quality, integrity, and accuracy of the source data before it's integrated into a database. This step is essential for preventing future complications in the model development process or issues like data drift that could degrade model performance. Teams can employ tools like Great Expectations to set data quality expectations or leverage tools like deepchecks to validate labels, dimensions, and data distributions, guaranteeing that the data is suitable for the task.

5. **Data Management:** Databases evolve over time, and as new data sources become available, it may be necessary to incorporate additional columns or tables. ETL (Extract, Transform, Load) processes are commonly used to bring data to its final usable format. Proper management and maintenance of the data are critical for building high-quality models. Data versioning can be employed to track historical data, new sources, and any changes made to the database, similar to code versioning. This approach ensures the availability of a stable version of the data, which can be invaluable in case of errors or to preserve a reliable product history.

3.1.4 Training Process

The model is usually trained using a process called unsupervised learning. The objective is to predict the probability distribution of the next word or sequence of words given a context. The model learns to minimize the difference between its predictions and the actual next words in the training data.

The model is trained iteratively on the tokenized dataset. During each iteration, it processes a sequence of tokens and tries to predict the next token. The model's parameters are updated based on the error between its predictions and the ground truth.

Backpropagation is used to calculate gradients that indicate how the model's parameters should be adjusted to reduce prediction errors. Optimization algorithms, such as stochastic gradient descent (SGD) or its variants, are employed to update the model's parameters in the direction that minimizes the training loss.

Training a large language model like GPT-3 requires parallel processing across multiple machines or specialized hardware like graphics processing units (GPUs) or tensor processing units (TPUs). The training process may take days or even weeks to complete.

Training large language models is a resource-intensive process that requires expertise in machine learning, access to extensive computational resources, and careful consideration of ethical considerations, such as data privacy, biases, and potential misuse.

3.1.5 Inference Process

The inference process is a fundamental concept in Machine Learning (ML) and Large Language Models (LLMs) that involves applying a trained model's learned knowledge to new, unseen data. This process enables the model to make predictions, classifications, or generate meaningful output based on the patterns it has learned during its training phase.

In the context of ML, once a machine learning model has been trained using a dataset containing input features and corresponding target labels, it undergoes an inference phase. During inference, the model is presented with new input data for which it needs to make predictions or classifications. The model applies the patterns and relationships it has learned from the training data to the new data and produces predictions or outputs. For instance, in a model trained to classify images of animals, during inference, the model takes an image as input and predicts the type of animal depicted.

Large Language Models (LLMs), like the GPT series developed by OpenAI, are a specialized type of ML model that excel in understanding and generating human language. In the context of LLMs, the inference process involves inputting a sequence of words or tokens to the model, which then generates coherent and contextually appropriate text as output. This output can be used for a variety of tasks, such as completing sentences, answering questions, translating languages, summarizing text, and more. LLMs leverage their understanding of language structure, grammar, and context to produce text that often closely resembles human-generated content.

In both ML and LLMs, the inference process is guided by the model's internal parameters and the patterns it has learned during training. It's important to note that while models strive to generate accurate and relevant output, they do so based on statistical patterns rather than true comprehension. This means that although the output might seem remarkably human-like, the model doesn't possess actual understanding or consciousness.

In summary, the inference process in Machine Learning and Large Language Models involves the application of trained models to new data for the purpose of generating predictions, classifications, or text outputs. This process leverages the learned patterns to bridge the gap between training and real-world application, enabling models to provide valuable insights and generate human-like language.

3.1.6 Evaluation

The evaluation process is a crucial step in the development and deployment of Machine Learning (ML) models, including Large Language Models (LLMs). It involves assessing the performance and capabilities of a trained model to understand how well it generalizes to new, unseen data or tasks.

In the realm of ML, once a model has been trained on a dataset, it's essential to evaluate its performance to ensure that it can make accurate predictions on real-world data. This involves using an evaluation dataset separate from the training data. The model is presented with this evaluation dataset, and its predictions or classifications are compared to the ground truth labels. Common evaluation metrics vary depending on the type of problem being solved. For example:

- **Classification Problems**: Metrics like accuracy, precision, recall, F1-score, and confusion matrices are used to measure how well the model classifies data into different categories.

- **Regression Problems**: Metrics such as mean squared error (MSE) or root mean squared error (RMSE) quantify the deviation between predicted and actual numeric values.

In the context of LLMs, evaluation involves assessing the quality of the generated text. Since LLMs often generate responses or completions, evaluating their output requires a combination of human judgment and automated measures. For instance, in open-ended tasks like text generation, human evaluators might assess factors like fluency, coherence, relevance, and grammaticality. Automated measures like BLEU (Bilingual Evaluation Understudy) and ROUGE (Recall-Oriented Understudy for Gisting Evaluation) are also employed to quantify the similarity between model-generated text and reference text.

The evaluation process serves several important purposes:

1. **Model Selection**: It helps in choosing the best model among different variations or algorithms by comparing their performance on the evaluation dataset.

2. **Generalization**: It provides insights into how well the model can perform on unseen data, indicating its ability to generalize beyond the training set.

3. **Hyperparameter Tuning**: It assists in tuning hyperparameters to enhance the model's performance.

4. **Bias and Fairness**: It helps identify potential biases or fairness issues in the model's predictions or outputs.

It's important to note that the evaluation process is iterative. Models might need to be retrained or fine-tuned based on evaluation results to achieve better performance. Additionally, evaluations are context-dependent; what's considered good performance varies based on the application and domain.

In summary, the evaluation process in Machine Learning and Large Language Models involves assessing a model's performance on new data or tasks. This process helps determine how well the model generalizes, aids in selecting the best model, and guides improvements to enhance overall performance and capabilities.

3.1.7 Deploy in production

The deployment process involves transitioning a trained model from the development environment to a production environment, where it can be used to make real-world predictions, generate outputs, or provide valuable insights.

After training and evaluation, the ML or LLM model is exported from the development environment, typically saved as a file that contains the model's architecture, parameters, and learned patterns.

In many cases, real-world data needs preprocessing before being fed into the model. This could involve data normalization, tokenization, scaling, or other transformations. Setting up a reliable data pipeline ensures that incoming data is properly processed and ready for the model.

In a production environment, the model needs to handle a potentially large volume of requests efficiently. Depending on the use case, this might involve optimizing the model's architecture, utilizing hardware accelerators (like GPUs or TPUs), and designing systems that can handle concurrent requests.

To make predictions or generate outputs, the model needs to be accessible through an Application Programming Interface (API) or as a service. This involves creating a server or service that can receive input data, send it to the model, and return the model's predictions or generated text back to the user or application.

Deployed models require continuous monitoring to ensure they are performing as expected. This includes tracking metrics like response time, accuracy, and resource utilization. Comprehensive logging helps in diagnosing issues and understanding how the model is being used.

Security measures must be in place to protect both the model and the data it processes. This might involve implementing authentication mechanisms, encryption, and access controls to prevent unauthorized access.

As models improve over time, it's essential to have a system for versioning deployed models. This allows for easy tracking of changes and the ability to roll back to previous versions if necessary. Additionally, updating the model in production requires careful testing to avoid disruptions.

Deployed models should have fail-safe mechanisms in place to handle unexpected errors or anomalies. This might involve setting default behaviors or responses when the model encounters data it hasn't been trained on.

Depending on the application and industry, models might need to adhere to specific regulations and compliance standards. Ensuring that the deployed model meets these requirements is critical.

The deployment process is not a one-time event. It's an ongoing cycle of monitoring, updating, and improving the deployed model based on real-world usage and feedback.

Successful deployment bridges the gap between the model's development and its practical application, enabling it to deliver value in various domains.

3.2 How to choose a model?

3.2.1 Model Size and capabilities

It's essential to carefully evaluate both the model's size and its capabilities, as they directly impact the model's performance, resource consumption, and overall suitability for your specific use case.

Language models come in a wide range of sizes, from relatively compact ones with a few hundred million parameters to massive models with several billion parameters. Larger models often offer expanded capabilities, allowing them to understand context, generate coherent responses, and handle complex language tasks more effectively. However, the size of the model also comes with a cost — increased computational requirements.

Larger models demand more computational power for various tasks, including training, fine-tuning, and inference. This translates to longer processing times and higher hardware costs. Furthermore, the storage demands for model checkpoints and deployment might become a consideration, particularly if your application needs to run on devices with limited space. Therefore, it's crucial to weigh the benefits of a larger model against the resources available to you.

The capabilities of a language model define its suitability for specific tasks. Different models excel in various areas, such as text generation, translation, sentiment analysis, or question answering. When evaluating the capabilities of a model, it's essential to align them with your application's requirements. Choosing a model with the right pre-trained abilities can significantly reduce the need for extensive fine-tuning, making the development process more efficient.

The selection process involves finding the optimal balance between model size, capabilities, and the resources at your disposal. For applications where real-time responses are crucial, a smaller model might be the better choice, even if it comes with slightly fewer capabilities. On the other hand, if your project demands cutting-edge language understanding and generation, a larger model may be the right fit, provided you can manage the increased computational demands.

3.2.2 Performance and Latency

These factors directly impact the model's ability to deliver accurate results and timely responses, making them critical considerations for a wide range of use cases.

Begin by examining the model's performance on relevant benchmarks or tasks. Metrics such as accuracy, precision, and recall provide valuable insights into the model's effectiveness in handling specific language processing tasks. It's crucial to assess the model's performance across different use cases to ensure it aligns with your application's requirements. For tasks like sentiment analysis, translation, or question answering, a model that consistently achieves high performance metrics is preferable.

Beyond performance, the response time or latency of the model is particularly significant, especially if your application demands real-time interactions or operates in low-latency environments. The response time measures how quickly the model can process input and produce an output. In scenarios like customer support chatbots, live translations, or voice assistants, minimizing latency is crucial to provide a seamless user experience. A model with low-latency capabilities ensures that users receive prompt and smooth responses, enhancing user satisfaction and engagement.

As you evaluate performance and latency, it's essential to strike the right balance based on your application's specific use cases. Some applications may prioritize high accuracy over low latency, while others require a swift response time at the cost of slightly reduced accuracy. Understanding the trade-offs between these factors is crucial for making an informed decision that aligns with your application's goals.

Additionally, consider the model's scalability. Will the model maintain consistent performance and latency as the workload increases? Scalability is essential for applications with varying traffic levels or those expected to handle spikes in user interactions.

3.2.3 Pretraining Data and Knowledge Cutoff

Understanding the pretraining data and knowledge cutoff of a language model is paramount when selecting the right model for your application. These factors directly impact the model's understanding of the world, its ability to handle recent information, and its relevance to current events and trends.

Language models leverage large and diverse datasets during pretraining to learn the intricacies of language. The quality and comprehensiveness of this pretraining data significantly influence the model's language understanding and generation capabilities. Consider the sources, domains, and languages included in the pretraining data to ensure they align with the context of your application. Models trained on a broad range of data sources might be more versatile and adaptable to different use cases.

The knowledge cutoff refers to the date until which the model has been exposed to data during pretraining. It's essential to recognize that models trained on data only up to a specific date might lack information from recent sources or events. If your application requires up-to-date information, such as real-time news analysis, current market trends, or recent research findings, selecting a model with a more recent knowledge cutoff is critical. This ensures that the model's knowledge remains current and relevant to the latest developments in the field.

Evaluate the temporal relevance required for your application. Some use cases, such as sentiment analysis of historical documents, may not demand the most recent information, making models with older knowledge cutoffs acceptable. However, applications like real-time news summarization or social media sentiment tracking necessitate models with up-to-the-minute knowledge to provide accurate and relevant insights.

In some cases, you might have the opportunity to fine-tune the model using recent data after its initial pretraining. This can extend the model's knowledge beyond its original cutoff date, allowing you to incorporate the latest information relevant to your domain.

It's important to strike a balance between knowledge cutoff and other factors, such as model size, capabilities, and resource requirements. A model with a very recent knowledge cutoff might be ideal for some applications but might come with trade-offs in terms of resource consumption or response time.

3.2.4 FineTuning and Customizability

Fine-tuning is a powerful capability that can significantly enhance the suitability of a large language model for your specific use case. When evaluating language models for your application, it's essential to consider the availability of fine-tuning options and the level of customizability they offer.

Fine-tuning enables you to adapt a pretrained model to better perform on your specific domain or task. This process involves training the model on a task-specific dataset, allowing it to learn the nuances of your data. Fine-tuning can improve the model's accuracy, relevance, and performance in the context of your application. Understanding the fine-tuning process and its requirements, such as the amount and quality of the training data, is crucial to harness its benefits effectively.

The ability to customize the model is paramount, especially if your use case demands domain-specific understanding or requires the model to address specific requirements. Models that support fine-tuning provide you with the flexibility to tailor the model to your unique needs. Whether you're working with medical data, legal documents, or any specialized domain, the model's customizability ensures that it aligns with the intricacies of your application.

Evaluate whether the model can be fine-tuned for the specific task you have in mind. Some models are better suited for particular tasks or domains. Ensure that the model's architecture and pretraining data are conducive to the task you intend to tackle.

While customization is valuable, it's essential to strike a balance between customizability and the pretrained knowledge the model brings. A model with strong pretraining can offer a solid foundation for a wide range of language understanding tasks. Leveraging both pretraining and fine-tuning can yield optimal results, enhancing the model's performance while retaining its broad language capabilities.

Check whether the model's documentation provides clear guidance on the fine-tuning process. A model with comprehensive documentation and a supportive community can make the fine-tuning journey smoother and more effective.

3.2.5 Open-source model & Licence

The distinction between open-source and proprietary models is a significant factor to weigh when selecting a language model for your project. This consideration can impact your ability to customize, integrate, and leverage the model for your specific needs.

Open-source models offer several key advantages. First and foremost, they are freely available, allowing developers to access, use, and modify the model without incurring licensing fees. This makes open-source models an attractive option for individuals or organizations with budget constraints or those seeking cost-effective solutions.

Open-source models empower developers with the flexibility to customize the model according to their requirements. You can fine-tune the model on your own domain-specific data, adapt it to unique tasks, or tailor it to fit the nuances of your application. This customization capability is especially valuable when your use case demands a high degree of specialization or when you want to refine the model's performance for specific contexts.

Open-source models facilitate seamless integration into your existing systems, whether it's embedding the model in a mobile application, deploying it in a cloud-based service, or integrating it with other AI technologies. The availability of open-source code and resources makes it easier to work with the model, and a community of developers often surrounds these models, fostering collaboration and innovation.

When dealing with open-source models, it's essential to review the specific open-source license under which the model is released. Different licenses have varying terms and may impose certain requirements, such as attributions or share-alike clauses. Make sure the license aligns with your intended usage and that you understand any obligations it entails.

On the other hand, proprietary models may come with licensing restrictions, limiting how you can use or modify the model. These models might be subject to fees, usage limitations, or access restrictions, which can impact your ability to fully leverage the model for your project.

3.2.6 Accessibility and cost

The accessibility and cost of a language model are crucial factors to consider when selecting the right model for your project. These considerations impact not only the feasibility of using the model but also the long-term sustainability of your application within your budget.

Some language models are open source, meaning they are freely accessible for use, modification, and distribution. Open-source models are an excellent choice for developers seeking cost-effective solutions. They eliminate the need for licensing fees, making them particularly attractive for individuals or organizations with limited financial resources.

Open-source models not only provide affordability but also offer flexibility and customization. You can adapt the model to your specific use case, fine-tune it on your own data, and even contribute to the model's development. This level of customization ensures that the model aligns precisely with your application's requirements.

In contrast, some models may come with licenses or usage costs. These models might have associated fees based on the number of requests, the amount of data processed, or the level of usage. It's crucial to carefully review the licensing terms and pricing structure to understand the potential financial implications over time.

Evaluate your budget and available resources. Consider not only the initial costs but also the ongoing expenses, especially if you expect your application to scale or require continuous usage. Models with licensing fees or usage costs should be assessed based on how well they fit within your financial constraints.

While cost is a significant consideration, it's essential to strike a balance between affordability and the model's performance. Sometimes, investing in a higher-quality model with a moderate cost might yield better results and long-term benefits compared to opting for a completely free but less effective model.

Assess the sustainability of your chosen model within your budget. Ensure that you can maintain and scale your application without encountering unexpected financial hurdles due to high licensing costs or usage fees.

3.2.7 Ethical and Responsible AI Considerations

As the use of large language models becomes more prevalent, it's essential to prioritize ethical and responsible AI practices when selecting a model for your project. The impact of these models on content generation, user interactions, and societal norms cannot be understated. Here's why ethical considerations should play a central role in your decision-making process:

One of the most significant concerns with language models is the potential for generating biased or inappropriate content. Language models learn from vast datasets, which may contain biases present in the real world. It's crucial to evaluate whether the model has undergone bias analysis and if the development team actively addresses and mitigates bias in its outputs.

Choose a model that has been developed with responsible AI practices. Look for models created by organizations that emphasize ethical considerations, transparency, and accountability. Models developed with responsible AI principles are more likely to have been subject to rigorous evaluation for fairness, transparency, and ethical use.

Look for models that provide clear guidelines and usage policies. These guidelines can help you understand the model's intended use, ethical boundaries, and restrictions. Transparent guidelines enable you to ensure that the model aligns with your ethical values and intended application.

Models that actively address bias mitigation are preferable. These models employ techniques to reduce biases in their outputs, making them more suitable for applications where fairness and inclusivity are paramount, such as chatbots used in customer support or language models in educational contexts.

Consider how the model ensures user safety. Models that filter inappropriate or harmful content, or those that can be fine-tuned to suit your specific content moderation needs, are valuable in creating a safe environment for users.

A model with an active and engaged community can be indicative of responsible development practices. A model with a strong user community is more likely to identify and address ethical concerns, share best practices, and collaborate on improving the model's ethical aspects.

3.3 Different approaches to train a model

3.3.1 Training from scratch

Learning from scratch refers to the process of training a machine learning model or algorithm without leveraging pre-existing knowledge or pre-trained models. When learning from scratch, the model starts with no prior information or understanding of the task at hand and learns directly from the provided training data. The model is initialized with random parameters or weights, then it gets trained on a labelled dataset specific to the task. In practice, the model iteratively processes the training examples, makes predictions based on its current parameters, and compares those predictions with the true labels. It then adjusts its parameters through optimization techniques like gradient descent, aiming to minimize the difference between its predictions and the true labels. With each iteration, the model updates its parameters to improve its performance on the task. Learning from scratch is often more time-consuming and computationally demanding compared to leveraging pre-trained models or transfer learning approaches because the model needs to learn from a relatively small amount of data without any prior knowledge or guidance. Additionally, learning from scratch may require a larger dataset to achieve comparable performance to pre-trained models, as it needs to capture all the necessary patterns and representations solely from the provided training data. It can be advantageous especially when the task or domain is highly specific or unique, and pre-existing models or knowledge are not readily available or relevant. It allows the model to develop task-specific representations and understanding directly from the training data, potentially leading to more tailored and specialized performance.

3.3.2 Training with Transfer Learning

Transfer learning is a method that enhances machine learning performance on a new task by leveraging knowledge from a pre-trained model. Instead of starting from scratch, the pre-trained model's learned representations and features are utilized as a foundation. This approach saves time and data, improves accuracy, and allows the model to adapt to new tasks or domains more effectively. By transferring knowledge, the model benefits from previous learning and achieves better results on the target task.

3.3.3 Training with finetuning

Fine-tuning is an important step in training large language models that allows customization for specific tasks or domains. Here's an overview of the fine-tuning process:

Task-specific dataset: To fine-tune a language model, a task-specific dataset is required. This dataset should be relevant to the desired task and should ideally contain a sufficient number of examples to train the model effectively. The dataset could include labeled examples for supervised tasks or paired data for tasks like machine translation.

Model initialization: The pre-trained language model, such as GPT-3, is used as a starting point for fine-tuning. The model has already learned a broad understanding of language from its pre-training phase, which provides a strong foundation for the fine-tuning process.

Architecture modification: Depending on the task, the model's architecture may need to be modified or extended. Additional layers or specific modules can be added to adapt the model to the requirements of the task. For example, for a sentiment analysis task, a classification head may be added on top of the language model.

Training objective: A task-specific objective is defined for fine-tuning. This objective varies based on the nature of the task. For example, for a classification task, the objective may involve minimizing the cross-entropy loss between predicted labels and true labels. For a machine translation task, the objective may be to minimize the translation loss.

Training process: The task-specific dataset is used to train the modified model. The model is presented with input sequences from the dataset, and the parameters are updated through backpropagation and optimization algorithms to minimize the defined training objective. The training process typically involves multiple iterations or epochs over the dataset.

Hyperparameter tuning: During fine-tuning, various hyperparameters, such as learning rate, batch size, regularization techniques, and optimization algorithms, need to be carefully tuned. Hyperparameter tuning aims to find the optimal configuration that maximizes the performance of the fine-tuned model.

Evaluation: After training, the fine-tuned model is evaluated on a separate validation or test set. This evaluation measures the model's performance on the specific task, providing insights into its effectiveness and potential areas for improvement.

Iterative refinement: Based on the evaluation results, further iterations of fine-tuning and evaluation may be performed to improve the model's performance. This process allows for the refinement of the fine-tuned model to achieve better results.

Generalization and deployment: Once the fine-tuned model achieves satisfactory performance on the evaluation metrics, it can be deployed for real-world applications. The fine-tuned model can be used to generate text, classify inputs, perform translation, or assist in various other language-related tasks specific to the fine-tuning objective.

It's worth noting that fine-tuning a language model requires a sufficient amount of task-specific data, and the performance of the fine-tuned model heavily depends on the quality and representativeness of the dataset. Additionally, ethical considerations should be taken into account during the fine-tuning process to address biases and potential issues related to fairness and inclusivity.

3. DATASET

In this chapter, we delve into the pivotal process of selecting datasets for fine-tuning a Large Language Model (LLM). Fine-tuning plays a vital role in unlocking the full potential of these models. During the fine-tuning process of an LLM, as well as in their initial training, the accurate selection of data to use emerges as a crucial step. Although it may seem less relevant compared to the selection of the base model or the fine-tuning process itself it constitutes the essential core of the entire procedure.

Like any other fundamentals step, dataset building requires an impressive effort in terms of capabilities, time and resources, an effort that will directly influence the outcome.

Therefore, this chapter meticulously examines the datasets used for fine-tuning, exploring their construction methods, strengths, and potential weaknesses.

3.1 Dataset Construction

The process of constructing a dataset involves several intricate steps aimed at gathering, organizing, and preparing relevant data to meet specific objectives. We need to select the scope and purpose of the dataset, identifying the target domain and the types of data required. Next, data collection methods are devised, which may include web scraping, manual annotation, or collaboration with domain experts to ensure data authenticity and relevance. Once the raw data is amassed, it undergoes preprocessing to remove noise, standardize formats, and address any inconsistencies.

Through this meticulous process, a well-constructed dataset forms the foundation for robust machine learning models and facilitates advancements in various domains.

3.1.1 Challenges and Considerations:

The construction of datasets for machine learning poses several challenges and requires careful consideration at various stages. Let's introduce the key aspects about the process:

1. **Domain Relevance:** Ensuring the relevance of the finetuning data to the specific task or domain is paramount. Optimal performance hinges on gathering data that authentically mirrors real-world scenarios the model will encounter. Strive to eliminate mismatches that could lead to suboptimal performance or unexpected outputs.
2. **Data Diversity:** Beyond mere quantity, the quality of diversity within the dataset is crucial for model robustness. A diverse dataset, encompassing variations in language, style, and perspectives, guards against biased or one-sided outputs. Aim for inclusivity and a comprehensive representation of possible inputs to enhance the model's adaptability.
3. **Data Size:** While larger datasets contribute to improved model performance, managing, and processing massive amounts of data presents technical challenges. Consider storage, computational resources, and processing time when determining the dataset size. A baseline of at least 1000 samples is recommended, with the actual size influenced by the choice of the foundational model for finetuning.
4. **Data Cleaning and Preprocessing:** Raw data often contains noise, errors, and inconsistencies. Implementing proper preprocessing steps, such as text normalization,

spell-checking, and removal of irrelevant content, is crucial to ensure the model learns from clean and coherent inputs.

5. **Data Annotation:** In tasks requiring labeled data, accurate and consistent annotation is paramount. Address ambiguities in labeling guidelines and inter-annotator disagreements to enhance the quality of the model's learning.

6. **Handling Rare Cases:** Acknowledge the significance of rare cases in task performance. Ensuring these infrequent but critical scenarios are well-represented in the data facilitates better generalization by the model, leading to accurate outputs in real-world applications.

7. **Ethical Considerations:** Vigilantly vet the data for potential biases or harmful content. Biases present in the training data can be perpetuated by the finetuned model, resulting in unintended consequences.

3.1.2 Strategies to Overcome Challenges:

Managing the challenges inherent in dataset construction requires a multifaceted approach that integrates best practices, ethical considerations, and technological solutions. There are some common indications and best practices to follow:

1. **Curate and Verify:** Invest time in curating a dataset aligned with the task's objectives. Manual verification of a subset of the data ensures adherence to quality standards. High-quality datasets, achieved through human effort, contribute to a cost-effective solution, avoiding the pitfalls of "garbage-in, garbage-out."

2. **Data Augmentation:** Enhance dataset diversity through augmentation techniques like synonym replacement, paraphrasing, or back-translation. These techniques expand the model's capacity to handle a broader range of inputs.

3. **Balancing Act:** Strike a balance between domain-specific data and more general language data. This ensures the model retains linguistic prowess while excelling in the target task.

4. **Iterative Refinement:** Acknowledge finetuning as an iterative process. Train, evaluate, and fine-tune the model and data iteratively to address gaps and correct shortcomings, optimizing the model's overall performance.

3.1.3 Dataset Choices

A high-quality dataset is the foundation upon which successful training/finetuning is based. The vastness and diversity of the data in the dataset directly affect the model's ability to understand and generate text in a consistent and contextually accurate manner. The use of a well-maintained dataset allows the model to learn in a more thorough and versatile manner, enabling better generalization to new linguistic situations.

In the context of this project, it was not possible to rely on any existing datasets. Indeed, there are not many datasets in the field of Large Language Models developed in languages other than English. Even more challenging is finding a dataset that not only has been developed in Italian but also pertains to the legal and fiscal domain, which is the focus of this project.

For this reason, it was decided to proceed by constructing two new datasets, ITACA and ADE, which constitute the only two datasets currently available in the legal domain and in Italian language, accessible online.To fulfill this task, there was obviously no resource to rely on, which led to considering some compromises that must be accepted when there is no access to an expert in the examined field. It was decided to define the two datasets through automation.

In particular, for ADE, only the technique of Web Scraping was utilized, while for ITACA, Artificial Intelligence was also employed, using an innovative technique that we will analyze below. In this way, we managed to overcome the daunting challenge of combining minimal resources with a dataset that is qualitative and reliable.

Let's now explore the processes and choices that led to the construction of the two datasets used.

3.2 ADE (Agenzia Delle Entrate) DATASET

3.2.1 What is ADE Dataset

ADE constitutes the most accurate and stable dataset used for fine-tuning the ITACA model.

ADE takes its name from the source used during the construction process, namely the website of the ”Agenzia Delle Entrate”. The idea was to use this portal as the primary source of information since the topic under consideration needs special attention. Tax and legal fields are areas where any inaccuracy can lead to not inconsiderable problems.

ADE is therefore a dataset constructed using the FiscoOggi.it portal, which represents the online magazine of the “Agenzia Delle Entrate”. The portal was created to offer updates on the activities of the Administration and its central and peripheral offices, comments on regulations, practices, and tax case law. It is an information tool available to taxpayers and professionals.

On the portal of the Agency, in addition to the sections where all the news of the moment is published and where all the available articles and guides are, there is a particularly interesting section for our purpose and on which we want to place particular emphasis, “La posta.”

This is an interesting service made available to all users to ask any kind of question in the tax and legal field. It is possible to ask what to do in particular tax situations, how to get bonuses or what to do to avoid IRS assessments and anything else related to tax, regulations and so on.

And for this reason, that it is particularly suitable for constructing a dataset for training a Large Language Model. In fact, each "article" is composed of a question, the one posed by the user in trouble, and an answer, coming from an authoritative source, namely the “Agenzia delle Entrate”.

The resulting advantages are many:

1) Speed of construction:

Since each article is already represented by a question and an answer, it is possible to define and use an automation to retrieve through the so-called "Web Scraping" mechanism all the articles that will compose our dataset.

2) Authoritativeness and reliability:

Answers come from the main source of information in the field, i.e., Italian Authority. This minimizes the possibility of introducing inaccurate or partially correct facts.

3) Language and Style:

It is important to pay attention to the fact that these are not generic questions, but specific questions composed specifically by the user for his or her financial and legal situation. This allows us to generalize our model and train it in particular situations as well. In addition, the answers are very carefully crafted by people who work in the field and know what they are talking about. This aspect is by no means to be overlooked when constructing the dataset, since very often, it is even more so in the academic field, it is extremely difficult for the person or team in charge of constructing a dataset in a particular field to also be trained in that field under consideration:

Size	3280
Format	JSON
Domain	Legal & Fiscal
Language	Italian
Source	Agenzia delle Entrate

3.2.1 The process behind ADE dataset

ADE is a dataset constructed entirely from a single source, namely that of FiscoOggi.it, a portal made available by the Internal Revenue Service.

As discussed extensively above, the portal provides an internally mailed section where questions posed by portal users can be found. However, although the site makes this information available to everyone, there are no APIs made available to the portal to be able to work on some automation to quickly build the dataset.

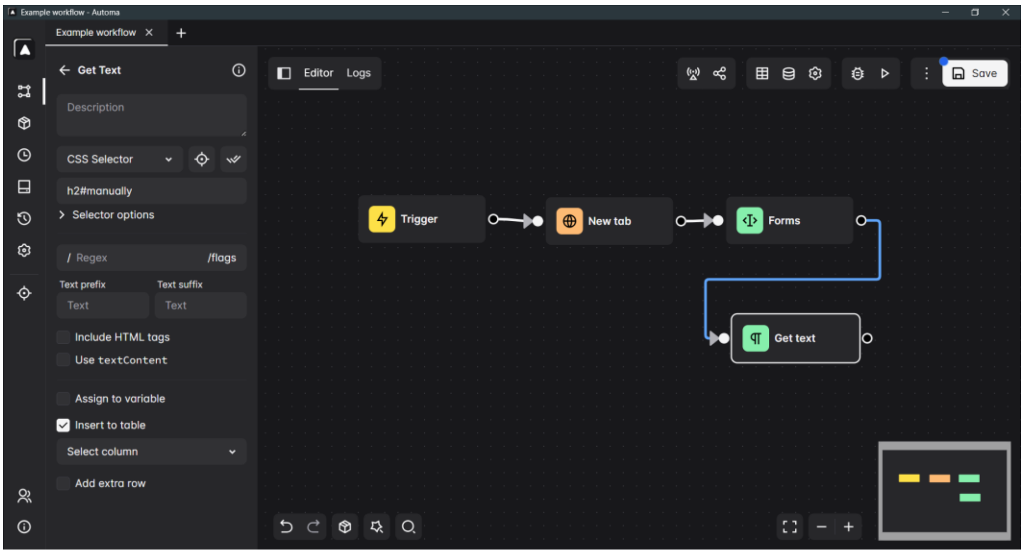
And for this reason, to perform Web Scraping, it was decided to take advantage of an external tool, the purpose of which is precisely to speed up the aforementioned process. The automation was designed using the Chrome browser extension, **Automa**.

Automa is a browser extension for automations on the browser. With this extension you can fill forms, define and execute repetitive tasks, open web pages, capture data, images and more, and generate tables and structured files. It is a very easy-to-use and very powerful tool for performing so-called web data scraping, i.e., capturing data from websites automatically and converting it in the form of structured data. Through this tool you can also collect them via CSV, JSON, Google Sheets and more.

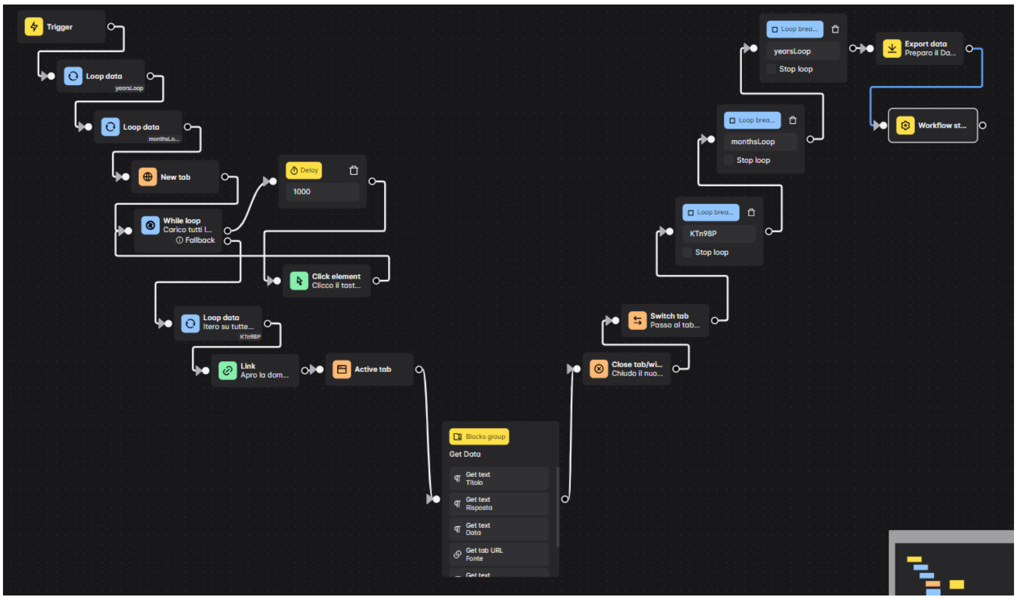
Automa provides an editor for defining its Workflows. A Workflow is a collection of connected blocks that define and automate a certain process. A workflow is executed from a trigger block, executing the blocks that follow it step by step.

The editor provides an additional tool for defining a workflow through screen recording. In fact, by taking advantage of this mode, actions performed on the browser will be recorded and automatically important within a new workflow.

Finally, it is possible to follow step by step the execution of a workflow through logs, which can be used to visualize the current point and the variables involved. There is also a debugging tool if you want to go into more detail.



Let us look in more detail at the workflow that enabled its definition:



Listed below are what steps are defined by automation to retrieve information and build ADE.

1. Definition of some global variables, such as years and months to iterate.
2. Iteration on the predefined years.
3. Iteration on the predefined months.
4. Opening mail page with the articles of the selected year and month.
5. Loading of all articles on the page.
6. Iteration on all articles present for the selected year and month.
7. Opening the link in a new tab.
8. SCRAPING: Retrieving all necessary information such as Title, Question, Answer, Date, Source through the use of CSS Selector. The information is retrieved and transferred to a table.
9. Closing the tab and selecting the previous one.
10. Once the process is finished, the table with all the retrieved data is converted into a JSON and downloaded.

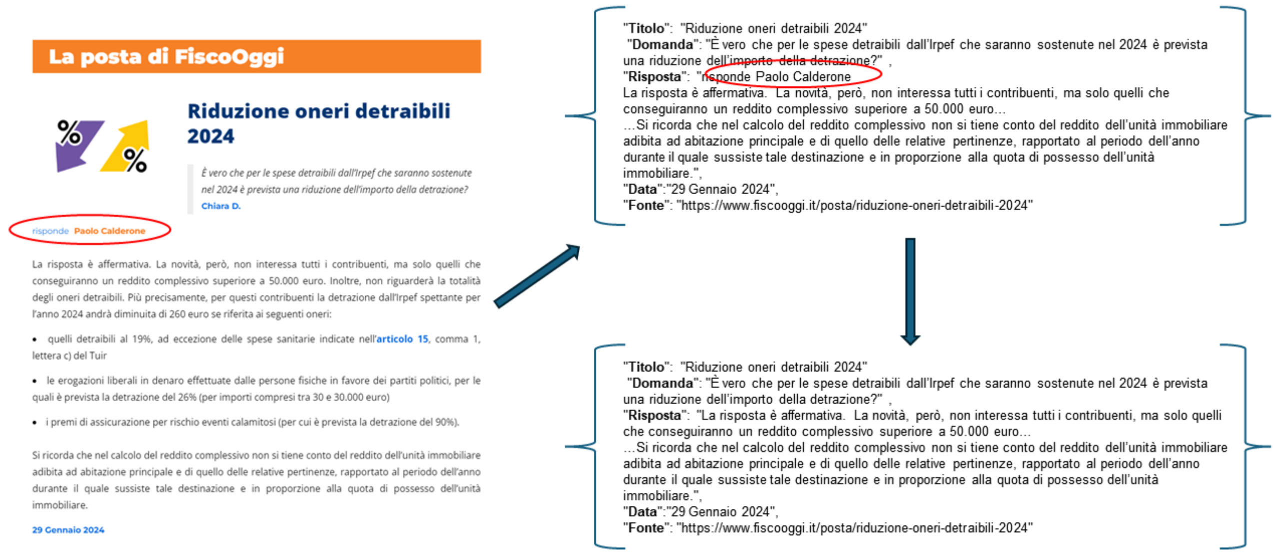
The workflow under consideration can be executed when triggered manually, it allows access to the FiscoOggi.it portal, to the section related to its mail, and iterating for each year and for each put, access to all the queries asked during the various years. The individual request is then opened on a separate page, and the data is retrieved, inserted into a table and then downloaded in JSON format at the end of the process.

3.2.3 Data Choices and Cleaning

The choices that were made during the definition of the dataset were determined by the fact that the web scraping procedure for this type of article had no contraindications.

In fact, no article was discarded for some reason, due to the fact that in this type of article, no source of noise could influence the generation of the dataset element. There are no videos or images in these articles, nor are there any particular limitations related to accessing articles rather than others. In this sense, all questions and answers available on the site since 2008 have been used for the purpose.

Regarding the pre-processing and cleaning of the collected data, it was chosen not to modify or alter any kind of information. Given the manner in which the data were collected, there are no duplicates or items that for some other reason need to be selected and eliminated. The only process of transformation they underwent concerns the response, in which the name and date of the person considering the question and responding was always present.



3.3.4 ADE Samples

Dataset Sample 1:

Titolo	Società estinta: notifica dell'accertamento
Domanda	La Srl di cui ero socio si è estinta nel 2012. A dicembre 2014 è stato notificato al nostro liquidatore un accertamento per il 2011. È legittimo? Chi è tenuto a ricorrere, noi soci o il liquidatore?
Risposta	Le società indicate all'articolo 2495 del codice civile cancellate dal registro delle imprese sono responsabili dei debiti fiscali e contributivi per cinque anni dopo la loro estinzione (articolo 28, comma 4, Dlgs 175/2014). Tale norma, in quanto procedurale, ha applicazione retroattiva (circolare 31/E/2014). Pertanto, a partire dal 13 dicembre 2014, data di entrata in vigore del decreto, l'avviso di accertamento contenente la rettifica della dichiarazione della società cancellata dal registro delle imprese deve essere emesso nei confronti della società cancellata e notificato alla stessa presso la sede dell'ultimo domicilio fiscale in quanto, a tal fine, l'effetto dell'estinzione si produrrà solo dopo cinque anni dalla data della cancellazione. L'avviso sarà impugnabile sia dai soci che dal liquidatore, entrambi soggetti responsabili ai sensi degli articoli 2495 del codice civile e 36 del Dpr 602/1973 (circolare 6/E/2015, paragrafo 13.4).
Data	31 Marzo 2015
Fonte	https://www.fiscooggi.it/posta/societa-estinta-notifica-dellaccertamento

Dataset Sample 2:

Titolo	Forfettari e fatturazione elettronica
Domanda	Un contribuente nel 2021 era in regime forfettario e ha conseguito ricavi inferiori a 25.000 euro. L'anno successivo, invece, l'ammontare dei ricavi ha superato di poco questo limite. Mi confermate che l'obbligo di emettere fattura elettronica scatterà per lui dal 2024 e non dal 1°gennaio 2023?
Risposta	Si conferma che per il contribuente in regime forfettario, che si trova nella situazione descritta nel quesito, l'obbligo di emettere fattura elettronica decorrerà dal 1° gennaio 2024.\nL'obbligo di fatturazione elettronica è già scattato dal 1° luglio 2022 per tutti i soggetti, precedentemente esclusi, che nell'anno precedente hanno conseguito ricavi o percepito compensi (ragguagliati ad anno) superiori a 25.000. Per tutti gli altri soggetti forfettari, invece, l'obbligo decorrerà dal 1° gennaio 2024, indipendentemente dai ricavi o compensi conseguiti nel 2022 (articolo 18, comma 3, del decreto legge n. 36/2022 e circolare dell'Agenzia delle entrate n. 26/2022).
Data	15 Febbraio 2023
Fonte	https://www.fiscooggi.it/posta/forfettari-e-fatturazione-elettronica

Dataset Sample 3:

Titolo	Comunicazioni di irregolarità: pagamento con lieve ritardo
Domanda	Se un contribuente effettua il pagamento delle somme richieste dall'Agenzia delle entrate con una comunicazione di irregolarità dopo due giorni dalla scadenza prevista per usufruire della sanzione ridotta, non ha più diritto a tale riduzione?
Risposta	Il contribuente che riceve una comunicazione di irregolarità, se vuole evitare l'iscrizione a ruolo e usufruire della riduzione delle sanzioni amministrative a un terzo deve effettuare il pagamento delle somme dovute entro trenta giorni dal ricevimento della stessa comunicazione (articolo 2, comma 2, del decreto legislativo n. 462/1997). In tal caso, inoltre, gli interessi sono dovuti fino all'ultimo giorno del mese antecedente a quello di elaborazione della comunicazione.\nPer effetto di quanto previsto dal Dpr 602/1973 (articolo 15-ter, commi 3 e 4), gli stessi benefici permangono anche nel caso in cui si effettui il pagamento con "lieve ritardo", non superiore a sette giorni. Pertanto, il contribuente che versa le somme richieste con la comunicazione di irregolarità entro 37 giorni dal ricevimento della stessa conserva il diritto alla riduzione della sanzione e degli interessi.
Data	7 Gennaio 2022
Fonte	https://www.fiscooggi.it/posta/comunicazioni-irregolarita-pagamento-lieve-ritardo

3.3 ITACA DATASET

3.3.1 What is the ITACA DATASET?

ITACA is a dataset named like the Large Language Model for which it was created. It is a dataset created using an innovative and modern approach, exploiting the combination of two different techniques, Web Scraping and Artificial Intelligence.

Two versions of the ITACA Dataset were born during the project, since the development progress achieved with the generator used for its building.

ITACA v1 contains more than 11,000 samples of questions and answers in the tax and legal fields while ITACA v2 contains about 10.500 samples in the same area. ITACA represents together with ADE, the only Dataset entirely in Italian in the legal and tax field available to date.

Although it was implemented using other artificial intelligence models, specifically using ChatGPT 3.5 Turbo, it has the peculiarity of being highly accurate and reliable due to the technique used in its implementation.

Contrary to what some may assume, questions aren't directed straight to ChatGPT. Instead of relying on its knowledge or stored data, we will use a different approach harnesses a key strength of modern language models (LLMs) on the market today: their ability to extract information from given text. In fact, these models are skilled at summarizing text or, in our case, turning it into a series of questions and answers.

In this sense, a dedicated tool, *LLMDSGenerator*, has been developed. It is a project that aims to to be used for the generation of a Q&A dataset for any LLM model. As we said, this tool attempts to combine two techniques already used to date for dataset creation, such as Web Scraping and the generation of synthetic datasets using artificial intelligence models.

3.1.2 ITACA Building Process

ITACA was completely developed using the LLMDSGenerator. This project aims to blend the speed and capabilities of artificial intelligence with models like ChatGPT4, ChatGPT 3.5 Turbo, Gemini, LLAMA2, or other local models combining them with authoritative information we provide to them. In practice, the tool analyses web pages and documents with an automation, clear, format and finally shares them with a LLM that will generate question and answer sequences based on the provided text.

LLMDSGenerator aims to create a clean and high-quality dataset, even when significant resources are lacking, and manual creation isn't feasible due to a lack of experts in the relevant field. Despite these challenges, the tool remains dependable and of high quality. It achieves this by utilizing authoritative sources such as web resources, private documents, regulatory texts, laws, and more.

The ITACA Dataset was conceived with the ambitious goal of creating a fully synthetic dataset using only artificial intelligence following the root tracked by Alpaca, which was constructed entirely through the APIs provided by OpenAI.

However, several factors render this approach suboptimal for our project and others of its kind. Let's delve into these considerations:

Language Barrier:

While the concept of synthesizing datasets isn't novel, challenges arise when dealing with languages other than English. Most Large Language Models are predominantly trained on English data, with only a minimal portion dedicated to other languages. This discrepancy is even more pronounced for languages like Italian. Consequently, relying solely on artificial intelligence to construct a synthetic dataset in Italian can lead to the formation of incorrect syntactic and grammatical structures, posing a significant risk.

Scope and Knowledge Limitations:

Large Language Models are trained on vast amounts of data and can generally respond to a wide array of queries. However, the quality of responses may diminish when questions become highly specific or delve into niche topics. The model's knowledge may be too generalized to provide accurate answers in such cases. As questions narrow down to microtopics, the reliability and quality of responses tend to decrease. To mitigate this, specialized models tailored to domains like tax and legal fields are preferred, or fine-tuning techniques are employed to enhance performance.

Out-of-Date Knowledge:

It's crucial to recognize that most commercially available models possess knowledge up to a certain date. In dynamic fields like tax and legal domains, relying on outdated data risks producing models that are obsolete and incapable of addressing recent developments. Such models may provide incorrect or irrelevant responses, undermining their utility.

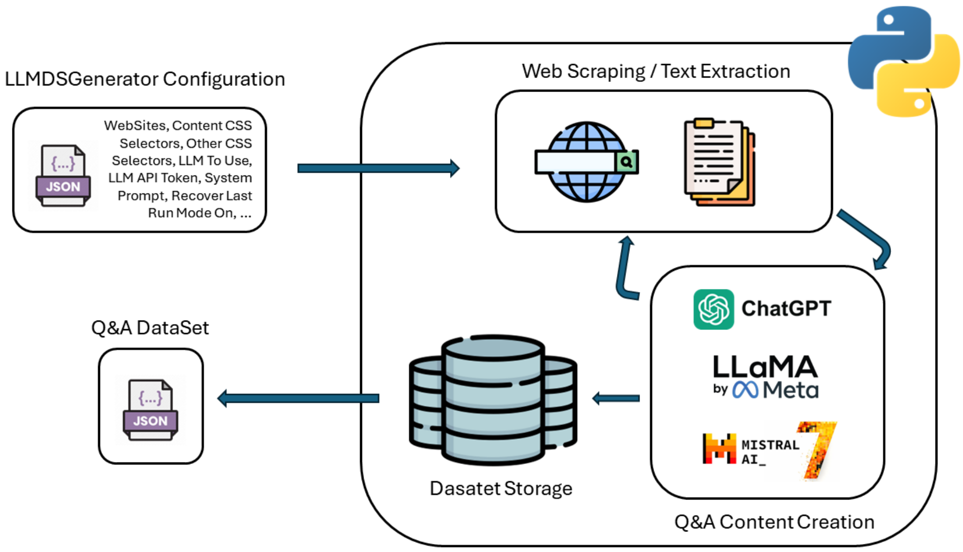
These challenges have spurred the development of our alternative project, LLMDSGenerator. It endeavors to achieve the same objectives and benefits while circumventing the inherent limitations of the conventional approach.

3.1.3 LLMDSGenerator (Large Language Model DataSet Generator)

LLMDSGenerator is a project developed within the ITACA initiative. Its primary objective is to provide a tool capable of generating Instruction datasets for Large Language Models, even with limited resources. More significantly, it facilitates this process across diverse domains, even in areas where expertise may be lacking.

To achieve this goal, the project harnesses the capabilities of artificial intelligence, blending it with the credibility of data sourced from authoritative and reliable sources on the internet. Users can explicitly specify these sources, ensuring the reliability of the generated datasets.

The project is entirely implemented in Python. LLMDSGenerator provides an interface versus all the modern Large Language Model, like ChatGPT, Gemini or Llama2.



The process is straightforward. By editing a configuration file, users can specify websites or documents to be scrapped. Additionally, they must provide various CSS selectors to extract information, a potential URL for traversing multiple pages, the communication token for the designated LLM, and the prompt for dataset creation.

You can specify to scrape both document or webpage depending from what you want to read contest. You can also decide to generate only one question/answer per request or multiple of them per request. The first option is very qualitative and affordable, while the second option is cheaper and effective.

Once initiated, the tool reviews the configuration file and proceeds to scrape the specified websites or the requested document. For each website, it gathers all visible articles using the provided selector. Upon selecting an article, it retrieves additional details such as title, date, and source based on the configuration file's selectors.

Subsequently, the main text is extracted using the appropriate selector. This text is then forwarded to the designated LLM, which generates a sequence of questions and answers based on its content.

```json
{
  "base_config": {
    "scraping_mode": "web",
    "workflow_mode": "n_question_per_request",
    "request_per_questions_generation": true,
    "data_augmentation": false,
    "questions_n": 5,
    "time_sleep_per_request": 1
  },
  "questions_gen_model_config": {
    "model_name": "gpt-3.5-turbo-0125",
    "max_tokens": 1024,
    "temperature": 1,
    "frequency_penalty": 0.3,
    "presence_penalty": 0,
    "top_p": 1,
    "sys_prompt": "QuestionsGeneratorPrompt.txt"
  },
  "answers_gen_model_config": {
    "model_name": "gpt-3.5-turbo-0125",
    "max_tokens": 4000,
    "temperature": 0.8,
    "frequency_penalty": 0.2,
    "presence_penalty": 0,
    "top_p": 1,
    "sample_sys_prompt": "AnswerGeneratorPrompt.txt",
    "samples_sys_prompt": "AnswersGeneratorPrompt.txt"
  },
  "model_prices": {
    "gpt-3.5-turbo-0125": { "input": 0.0000005, "output": 0.0000015 },
    "gpt-4-0125-preview": { "input": 0.00001, "output": 0.00003 }
  },
  "doc_mode_config": {
    "docs": []
  },
  "web_mode_config": {
    "resume_last_workflow": false,
    "sites": [
      {
        "name": "fiscoetasse",
        "description": "Regime Forfettario",
        "base_url":
"https://www.fiscoetasse.com/regime_forfettario_contribuenti_minimi/tipo/speciali",
        "skip_site": true,
        "page_iterator_url": "?pagina={page_number}",
        "start_page": 1,
        "article_links_selectors": ["a.card-title"],
        "start_link": 0,
        "title_selectors": ["h1.text-primary"],
        "date_selectors": [],
        "content_selectors": ["div.main-text"],
        "content_cleanup_selectors": [],
        "skip_article_selectors": []
      }
    ]
}
```

The system's prompt is built starting from a starting template in order to inform and assure the LLM to follow some guide lines and that the response should be in question-and-answer format in JSON. The other part of the template is customizable, allowing for the inclusion of any desired additional information.

Moreover, users have the option to specify pages or articles to skip based on predefined CSS elements. Additionally, parts of articles that could potentially detract from text quality, such as images, tables, and advertisements, can be removed.

An essential feature of the tool is its capability to resume from a previous run. Interruptions, such as connection failures or limits imposed by the LLM, are common occurrences during the process. Consequently, the tool automatically saves a backup of the dataset and a configuration file to facilitate seamless resumption of the operation.

Once the pages are processed and the dataset is compiled, a JSON file containing all generated questions and answers is produced.

3.1.4 Data Choices and Cleaning

For the construction of the ITACA dataset, several reference and authoritative sources in the tax and legal fields were used.

Any elements that could interfere with the text, such as advertisements and external figures, were removed from all sites and documents used.

The system prompt used for generating the dataset underwent careful selection after a series of experiments and considerations. Emphasis was placed on adhering to specific key points, following best practices outlined by OpenAI to optimize model performance.

Let's examine and analyze some of these practices:

Clear Instruction:

It's crucial to provide clear instructions to the model to ensure accurate responses. This includes specifying details in queries, guiding the model to adopt a specific persona, using delimiters to indicate distinct input parts, specifying task steps, providing examples, and defining the desired output length.

Provide Reference Text:

To reduce fabrication and improve response accuracy, reference text should be provided to the model. Instructing the model to answer using or citing from this reference text enhances response quality.

Split Complex Tasks:

Complex tasks should be broken down into simpler subtasks, mirroring modular design principles in software engineering. This helps reduce error rates and improves model performance. Techniques like intent classification, dialogue summarization, and document summarization can aid in this process.

Test Changes Systematically:

Systematic testing is essential for evaluating performance improvements. Modifications to prompts should be rigorously tested across various examples to ensure net positive performance gains.

By following these best practices, the system message analyzed for the implementation of ITACA aimed to optimize dataset generation. The message, along with predefined initial and final segments, was integrated directly into the dataset development tool.

Now let us analyze the system message that was analyzed for the implementation of ITACA and try to understand how the best practices analyzed earlier were followed. Recall that to this message must then be added the init and final part of the message that is predefined and inserted directly within the tool used to develop the dataset.

Il tuo compito è di formulare delle domande riguardanti l'articolo e fornire delle risposte.

I requisiti sono:

1)Domande e risposte devono utilizzare solo informazioni presenti nell'articolo;

2)Domande e risposte non devono mai citare l'articolo direttamente;

3)Ogni coppia domanda/risposta deve essere indipendente dalle altre;

4)Domande e risposte devono riproporre tutti i concetti chiave dell'articolo;

5)Le risposte non devono essere brevi ma dettagliate ed esaustive e contenere un minimo di 200 parole ed un massimo di 1000;

6)Se le informazioni presenti nell'articolo non sono sufficienti per una risposta dettagliata, aggiungi: <<Se vuoi saperne di più, rivolgiti ad uno dei nostri esperti.>>

7)Quando la domanda e/o la risposta fanno riferimento a delle date, invita a controllarne la veridicità su fonti ufficiali, aggiungendo alla risposta <<Ti invitiamo a verificare sempre su fonti ufficiali le informazioni riportate o a rivolgerti ad uno dei nostri esperti. Alcune informazioni potrebbero essere cambiate nel corso del tempo o non essere più veritiere.>>"

The system prompt aims for clarity, ensuring all necessary information is included, even if it seems obvious at first. It specifies what to say when information is scarce or when referencing dates.

Introducing text input enhances model performance by providing reliable sources. Breaking down the message into steps aids comprehension for both us and the model, ensuring each request is fully understood and distinct.

The system message underwent extensive testing, revealing limitations of previous versions. For instance, specifying word count in responses was necessary as the model tended to provide brief answers without explanations. Similarly, explicit requests to only use information from provided articles and to add disclaimers when needed were implemented to prevent the model from adding unverified information.

3.1.5 ITACA Dataset Samples

Dataset Sample 1:

Titolo	*Uscita dal forfettario e passaggio ad SRL: come funziona*
Domanda	*Quando è consigliabile passare dal regime forfettario a una SRL?*
Risposta	*Il passaggio a una SRL è consigliabile quando l'attività supera i limiti di fatturato del regime forfettario, quando si prevede una crescita significativa, o quando si desidera limitare la responsabilità personale e aumentare la credibilità aziendale. Inoltre, se si vuole attrarre investimenti e partecipare a gare d'appalto, la costituzione di una SRL può offrire dei vantaggi. Si consiglia di valutare attentamente la propria situazione e consultare un commercialista esperto per prendere la decisione più adatta.*
Data	*25 Gennaio 2024*
Fonte	*https://fiscomania.com/passaggio-forfettario-srl/*

Dataset Sample 2:

Titolo	*Revoca cedolare secca e aggiornamento canone locazione*
Domanda	*Cosa prevede la cedolare secca in termini di tassazione per chi percepisce redditi dalla locazione di immobili?*
Risposta	*La cedolare secca è un regime di tassazione facoltativo caratterizzato dal pagamento di un'imposta sostitutiva dell'IRPEF e delle addizionali. Possono esercitare l'opzione per la cedolare secca tutti coloro che percepiscono redditi dalla locazione di immobili o che godono di diritti reali su di essi*
Data	*17-gen-24*
Fonte	*https://fiscomania.com/revoca-cedolare-secca/*

3.1.6 ITACA Dataset v2

With the evolution of LLDSGenerator, several new features have been integrated, many of which have had a significant impact on the potential of the dataset created. For this reason, it was decided to perform a second run of the generator, this time using a new execution flow provided by the generator.

While in the first case, the question and answer were generated simultaneously, using a single request, the evolution of the generator has introduced the possibility to specify whether to generate a prompt for each request. Furthermore, another mode has been made available. It is now possible to decide to generate questions and answers using two separate prompts and requests.

This new technique not only saves compared to the previously seen mode, which requires one request for each question, but also increases the reliability and correctness of the questions and answers provided.

It is possible to decide how many questions to generate. The request is sent to the LLM, which uses the context first to generate the questions, dedicating a separate request for this activity, resulting in higher quality questions. Subsequently, these questions are sent back to the Large Language Model, packaged in the number defined by the user, who can choose to generate a request for each question, or to generate them all at once, thus saving a significant amount of time and resources required for generation.

Finally, version 2 of the ITACA Dataset represents an important step forward in terms of quality.

5. METHODOLOGY

As we've discussed in detail, our primary aim is to develop a Large Language Model that can effectively respond to common questions in Italian regarding taxation, law, and commerce. However, it's essential to emphasize that our project extends beyond this specific goal. We aspire to showcase how remarkable outcomes can be achieved using open-source models and to explore cutting-edge techniques that empower us to tackle this challenging task with limited resources and within a defined timeframe.

We've ruled out the option of entrusting our sensitive data to third-party frameworks and models, as our objective is to maintain control over our proprietary data. While fine-tuning ChatGPT might seem like a straightforward solution, it doesn't align with our core purpose. Instead, we're committed to achieving success with our modest resources while safeguarding our custom and private data.

In this chapter, we'll embark on a comprehensive exploration of various Large Language Models, evaluating them across different metrics to identify the most suitable options available with commercial licenses. Additionally, we'll introduce the techniques and approaches we've employed to fine-tune the model, aiming to streamline the process and conserve resources and time.

Now, let's delve into the essence of our project.

5.1 Selection of the Base Large Language Model

Once the datasets have been designed and created, the next step toward our goal is to select a basic model from which to start. In order to perform the better choice, it is important to evaluate and define the state-of-the-art in LLM environment. This is an extremely important choice because a basic model that does not fit well with our goal will result in something that is not up to par, also with a good dataset and training.

5.1.1 Context and exclusion:

The emergence of OpenAI and ChatGPT in the modern landscape has truly revolutionized our era, introducing chatbots to a wide audience and reshaping their applications across various fields. However, the internet offers an extensive suite of models, each possessing distinct features, advantages, and drawbacks. Hence, it becomes imperative to sift through these options to identify those that align closely with the specific requirements of our project.

Initiating our selection process with models like OpenAI's ChatGPT4, Antropic's ClaudeAI, or Google's Gemini would position us at an exceptionally high starting point. These companies often provide user-friendly tools to streamline the fine-tuning process, making them seemingly attractive options. However, such a choice is not viable for us. Our objective is to train the model without compromising on privacy by avoiding third-party data sharing and retaining complete control over it. Moreover, these models come with usage restrictions, making them unsuitable for commercial purposes and imposing intricate and limiting regulations. These factors render them unsuitable for our project's objectives.

Furthermore, a secondary filter is necessary based on the available resources at our disposal. While some open-source models rival industry giants in performance, the idea to get these models and fine-tuning them remains a lofty aspiration. Training models like Llama70B, Falcon140B, and Mistral8x7B demands an immense amount of data and computational power, far beyond what we have at our disposal. One of our goals focus in minimizing resource utilization during training, coupled with the challenge of insufficient data to mitigate overfitting risks.

Consequently, models with parameters exceeding 70 billion, such as LLama2 70B, Mistral8x7B, Falcon70B, Falcon 170B, MTP70B and many others like them must be excluded from our consideration.

5.1.2 State-of-the-arte LLM:

Despite the extensive filtering process we conducted earlier, there are still numerous models awaiting exploration. Our attention will now shift to open-source models, which offer unrestricted usage for both research and commercial endeavors. Of particular importance will be the consideration of the model's parameter count and the resources required for fine-tuning.

Let's now introduce a few models that we've meticulously examined:

MPT 7B

MPT-7B, developed by MosaicML, stands as a decoder-style transformer model meticulously pretrained from scratch on a vast corpus of 1 trillion tokens encompassing both English text and code. This model represents a pivotal member of the MosaicPretrainedTransformer (MPT) family, distinguished by its tailored transformer architecture engineered for optimized training efficiency and inference performance.

The architecture of MPT models incorporates several strategic enhancements, including performance-optimized layer implementations and the elimination of context length restrictions. Notably, positional embeddings are replaced with Attention with Linear Biases (ALiBi[1]), enabling MPT models to process inputs without constraint on length. This architectural innovation not only facilitates high-throughput training but also ensures stable convergence during model optimization. Moreover, MPT models exhibit remarkable efficiency in both training and serving, seamlessly compatible with standard HuggingFace pipelines [2]and NVIDIA's FasterTransformer [3]framework.

[1] A novel positional representation method which biases query-key attention scores with a penalty proportional to their distance, enabling efficient extrapolation for transformer models at inference time.

[2] Pipelines from Hugging Face are a great and easy way to use models for inference other NLP tasks.

[3] NVIDIA's FasterTrasformer is a library allowing a high-performance training and inference by GPU usage.

MPT-7B boasts several noteworthy attributes:

- Licensing for commercial usage, setting it apart from models like LLaMA.
- Extensive training data comprising 1 trillion tokens, surpassing datasets used for other models such as Pythia, OpenLLaMA, and StableLM.
- Exceptional capability to handle significantly long inputs, facilitated by ALiBi. For instance, MPT-7B-StoryWriter-65k+ has been fine-tuned to process inputs up to 65k tokens, surpassing the limited capacity of other open-source models.
- Swift training and inference enabled by FlashAttention and FasterTransformer.
- Availability of highly efficient open-source training code through the llm-foundry repository.

MPT-7B emerges as a versatile and potent tool, equipped with cutting-edge architecture and capabilities tailored for diverse applications in natural language processing and beyond.

STABLE LM 7B

StableLM-Base-Alpha represents a suite of decoder-only language models boasting 3B and 7B parameters, meticulously pre-trained on a diverse array of English datasets. Developed by Stability AI, a trailblazer in the generative AI domain spearheaded by Emad Mostaque since late 2020, these models are engineered to transcend the context window constraints often encountered in existing open-source language models.

Distinguished by their robust architecture, StableLM-Base-Alpha models leverage a sequence length of 4096, enabling them to delve deeper into text understanding and generation. Notably, they are pre-trained on an innovative experimental dataset derived from The Pile[4], significantly expanding their scope with an unprecedented token count of approximately 1.5 trillion tokens.

Employing mixed-precision (FP16) training and optimized with Adam, these models are trained using the NeoX tokenizer, boasting a vocabulary size of 50,257. Detailed hyperparameter selections are meticulously documented and available in the project's GitHub repository, ensuring transparency and reproducibility.

[4] An 825 GiB English text corpus comprised of 22 diverse high-quality subsets, aimed at enhancing the general cross-domain knowledge and downstream generalization capability of large-scale language models.

One of the key features of StableLM-Base-Alpha models is their versatility, designed to serve as foundational models adaptable for fine-tuning in various application-specific contexts. Importantly, there are no stringent limitations imposed on their commercial utilization, making them accessible to a wide range of individuals and enterprises alike.

DOLLY 2.0

Databricks, renowned for its unified data analytics platform, has recently unveiled Dolly 2.0, an open-source large language model (LLM) that echoes the interactive capabilities of ChatGPT. Sporting a staggering 12 billion parameters, Dolly 2.0 is derived from EleutherAI's Pythia-12b and finely honed using a corpus of approximately 15,000 instructional records curated by Databricks personnel. The model undergoes training across a spectrum of skill domains, encompassing brainstorming, classification, closed QA, generation, information extraction, open QA, and summarization.

Dolly 2.0 distinguishes itself with its exceptional proficiency in adhering to instructions, a testament to its meticulous training process. Despite its reliance on a comparatively modest dataset, the model impressively showcases high-quality instruction-following capabilities.

Regarding fine-tuning, Dolly 2.0 presents numerous merits. It boasts an open-source framework and is commercially licensed, affording organizations the freedom to craft, customize, and own potent LLMs sans the constraints of API access fees or data-sharing obligations. However, it's crucial to acknowledge the potential pitfalls of fine-tuning on a limited dataset, namely the risk of overfitting.

LLAMA 7B

Llama 2 represents a comprehensive suite of generative text models, meticulously fine-tuned by Meta AI, the pioneering force behind Facebook's advancements in artificial intelligence. This groundbreaking collection, designed with the aim of democratizing access to AI resources, stands as a testament to Meta AI's commitment to innovation in the field.

Comprising models of varying scales—from 7 billion to a staggering 70 billion parameters—LLama 2 offers versatility and scalability to cater to diverse needs. Trained on a vast amalgamation of publicly available datasets, totaling an impressive two trillion tokens, these models are meticulously crafted to excel in dialogue-based scenarios. Their performance rivals that of prominent closed-source models like ChatGPT and PaLM, further solidifying LLama 2's reputation as a formidable contender in the realm of natural language processing.

Powered by an optimized transformer architecture, LLama2 operates as an auto-regressive language model, capable of generating contextually coherent text with remarkable fluency and accuracy. Targeted primarily towards commercial and research applications in English, LLama 2 offers two distinct variants: tuned models tailored for assistant-like chat functionalities and pretrained models adaptable to a myriad of natural language generation tasks.

A custom commercial license is available, ensuring compliance with legal and regulatory frameworks. However, LLlama2 is intended solely for use within the bounds of applicable laws and regulations, particularly emphasizing adherence to trade compliance laws. Furthermore, its usage is restricted to the English language, underscoring the importance of linguistic context and appropriateness in deployment.

FALCON 7B:

The Falcon LLM, developed by the Technology Innovation Institute in Abu Dhabi, stands as a pioneering achievement in AI language processing, redefining the landscape with its exceptional capabilities. The Falcon family comprises models such as Falcon-180B, Falcon-40B, Falcon-7.5B, and Falcon-1.3B, each bringing unique strengths to the forefront and establishing the Falcon LLM as an innovative and adaptable tool across various applications.

These models showcase remarkable proficiency across a broad spectrum of language tasks, including question-answering, reasoning, and generating human-like text. Notably, models like Falcon-40B and Falcon-7B surpass the performance benchmarks set by other renowned language models such as LLaMA, StableLM, RedPajama, and MPT. The secret to Falcon's superiority lies in its meticulously crafted architecture and advanced training methodology.

Falcon-7B, for instance, is a 7-billion-parameter causal decoder-only model developed by TII and trained on an extensive dataset comprising 1,500 billion tokens sourced from RefinedWeb[5],

[5] The Falcon RefinedWeb dataset, developed by TII and licensed under ODC-By 1.0, is a substantial English web dataset resulting from rigorous filtering and extensive deduplication of CommonCrawl.

bolstered by curated corpora. This model is generously offered under the permissive Apache 2.0 license.

Let's explore the key advantages offered by Falcon:

- It consistently outperforms comparable open-source models like MPT-7B, StableLM, and RedPajama, owing to its robust training on the richly diverse dataset of 1,500 billion tokens from RefinedWeb, as evidenced by the OpenLLM Leaderboard.
- It boasts an architecture optimized for inference, featuring cutting-edge components like FlashAttention [6](Dao et al., 2022) and multiquery (Shazeer et al., 2019), enhancing its efficiency and effectiveness in real-world applications.
- The model's availability under the Apache 2.0 license ensures unrestricted commercial use, devoid of any royalties or constraints, making it an ideal choice for a wide array of projects and endeavors.

The Falcon-7B model is designed to serve as a valuable asset for research on large language models and as a foundational platform for further specialization and fine-tuning tailored to specific use cases, such as summarization, text generation, and chatbots.

It is essential to note that Falcon-7B is trained exclusively on English and French data and may not demonstrate optimal performance in other languages. Additionally, as it draws from a vast corpora representative of the web, it may inadvertently perpetuate stereotypes and biases prevalent in online content. Hence, careful consideration is warranted when deploying the model in diverse contexts.

MISTRAL 7B:

Mistral AI, a French company specializing in artificial intelligence, was founded in April 2023 by former researchers from Meta and Google DeepMind: Arthur Mensch, Timothée Lacroix, and Guillaume Lample. The company has developed an advanced platform tailored for training, serving, and evaluating large language models.

---

[6] FlashAttention presents an attention mechanism designed to enhance the computational efficiency of transformer-based models by dynamically adapting attention computation based on the relevance of input tokens

Among Mistral AI's notable achievements is the creation of Mistral 7B, a robust language model boasting 7.3 billion parameters. This model represents a significant leap forward in large language model capabilities, surpassing the performance of the 13 billion parameter Llama 2 model across all tasks and even outstripping the 34 billion parameter Llama 1 on many benchmarks.

An exceptional feature of Mistral 7B is its balanced performance across a diverse range of tasks. Leveraging Grouped-query Attention (GQA[7]) for accelerated inference times and Sliding Window Attention (SWA[8]) for managing lengthy text sequences at a minimal computational cost, Mistral 7B emerges as a versatile solution suitable for various applications. Furthermore, its availability under the permissive Apache 2.0 license promotes open and unrestricted access to this powerful AI resource.

However, like any model, Mistral 7B is not devoid of limitations. Despite being relatively smaller compared to some competitors, its size can still pose challenges, especially for tasks requiring low-latency responses. Additionally, while adept at handling a wide spectrum of tasks, its performance may exhibit variations depending on the specific task being addressed. Nevertheless, Mistral 7B stands as a formidable tool in the realm of AI and machine learning, offering immense potential for advancing research and application development.

5.1.3 Base LLM Comparation:

In today's rapidly evolving landscape of large language models (LLMs) and chatbots, it's increasingly challenging to discern genuine advancements amidst the deluge of releases, each boasting remarkable performance claims.

To better navigate this maze of innovation effectively, a HuggingFace space called *open_llm_leaderboard* has been lunched aimed at identifying the true state-of-the-art models within the open-source community.

---

[7] Grouped Query Attention simplifies how LLMs understand large amounts of text by bundling similar pieces together. This makes the model faster and smarter, as it can focus on groups of words at a time instead of each word individually.

[8] Sliding window attention is a computational mechanism used in natural language processing and computer vision tasks, wherein attention is focused on a subset of input tokens or image regions at a time, enabling efficient processing of large inputs while maintaining contextuality and performance.

Their evaluation methodology revolves around six key benchmarks, facilitated by the Eleuther AI Language Model Evaluation Harness. These benchmarks include:

- AI2 Reasoning Challenge (25-shot): Assessing performance on grade-school science questions.
- HellaSwag (10-shot): Testing commonsense inference abilities, challenging even for state-of-the-art models.
- MMLU (5-shot): Evaluating multitask accuracy across 57 diverse tasks spanning subjects like mathematics, history, computer science, and law.
- TruthfulQA (0-shot): Gauging a model's inclination to reproduce falsehoods prevalent online, even in the absence of prior training data.
- Winogrande (5-shot): Confronting models with adversarial Winograd benchmarks to evaluate commonsense reasoning skills at scale.
- GSM8k (5-shot): Presenting a series of grade school math word problems to assess a model's proficiency in solving multi-step mathematical reasoning challenges.

In all these evaluations, a higher score signifies superior performance. We selected these benchmarks for their ability to test a wide range of reasoning abilities and general knowledge, both in zero-shot and few-shot settings. Through this comprehensive evaluation framework, we aim to uncover the true champions among the myriad of LLMs, facilitating informed decisions and driving progress in natural language understanding and generation.

Model	#Params (B)	AVG	ARC	HellaSwag	MMLU	TruthfulQA	Winogrande	GSM8K
Zephyr 7b	7	62,03	84,36	61,07	57,45	77,74	29,04	62,03
Mistral 7B v0.2	7	60,34	60,49	82,94	63,42	41,8	78,69	34,72
Llama-7b	7	53,07	78,59	46,87	38,76	74,03	14,48	53,07
Falcon-7b	7	47,87	78,13	27,79	34,26	72,38	4,62	47,87
stablelm7b-v2	7	47,35	77,08	45,1	36,46	68,51	2,58	47,35
Mpt-7b	7	//	//	//	//	//	//	//

5.1.4 LLM Italian Evaluation

One of the main limitations found in state-of-the-art modern models is their ability to adapt to languages different from those predominantly used during training. Despite the existence of numerous models, some of which can compete with the largest ones available, they often lack proficiency in using the Italian language simply because it has been underutilized, if not entirely absent, during the training phase.

To ensure that we start with the most suitable model for this purpose, we conducted evaluations on the performance of the examined models by executing some simple tasks in the Italian language. The tests were conducted using HuggingFace Hub, which provides some of the most used models in their Instruction or Chat versions for free.

We defined certain configuration parameters for inference to avoid excessively long responses and ensure that a too high temperature does not cause the model to fail to use a different language correctly. Here they are:

Parameter	Value
Max-Tokens	512
Temperature	0.7
System Prompt	Rispondi brevemente alla domanda

Question #1:

Qual è il senso della vita?	
Mistral-7B-Instruct-v0.2	La risposta al significato della vita è una domanda filosofica che ha ispirato molte teorie e discussioni attraverso i secoli. Non esiste una risposta definitiva o universale a questa domanda, poichè la sua interpretazione varia in base alle diverse culture, religioni, filosofie e personalità. Alcune persone credono che la vita abbia un scopo specifico o un fine, mentre altre creano loro stessi il significato della propria esistenza. In generale, però, si può dire che la maggior parte delle persone cerca di vivere felici, soddisfatte e contribuire positivamente ai propri circondari e alla società in grande.
LLama7b Chat	Il senso della vita è una domanda che ha intriguato filosofi, religiosi e persone comuni per secoli. Non c'è un'unica risposta definitiva, poiché è un tema muy personale e subjectivo. Tuttavia, alcune possibili risposte potrebbero essere: • La realizzazione di sé stessi e del proprio potenziale;

	• L'amore e le relazioni con gli altri; • La scoperta di uno scopo o missione maggiore; • La ricerca della felicità e del benessere; • La crescita spirituale e la comprensione del mondo. In sintesi, il senso della vita può variare da persona a persona, ma generalmente si riferisce all'idea che ogni cosa nobilita e dà significato alla nostra esistenza.
Zephyr 7b Chat	The meaning of life is a philosophical and existential question that has been debated by scholars, philosophers, and individuals for centuries. There is no single definitive answer that applies universally to every person. For some, the meaning of life may be found in personal fulfillment, such as pursuing happiness, love, and personal growth. Others may find meaning in making a positive impact on the world through acts of kindness, altruism, and social activism. In many religious and spiritual traditions, the meaning of life is tied to a higher purpose, such as serving a deity, fulfilling a moral code, or achieving salvation. Ultimately, the meaning of life is a deeply personal and subjective question that each individual must explore and answer for themselves. It may evolve and change throughout one's life as they gain new experiences and perspectives.

Question #2:

Consigliami un pasto a base di pesce per il pranzo di oggi	
Mistral-7B-Instruct-v0.2	Suggestion for a fish-based lunch: Grilled salmon with quinoa, steamed broccoli, and avocado salsa. It's both delicious and nutritious!
LLama7b Chat	Certamente! Per un pasto delizioso e salutare a base di pesce, ti consiglio "Fritti di pesce con patate e verdure" oppure "Pasticcio di tonno con spinaci e formaggio". Spero che questo te sia stato utile!
Zephyr 7b Chat	Certamente! Ecco una proposta per un gustoso pasto a base di pesce per il pranzo di oggi: Insalata di tonno, avocado e pomodorini Ingredienti: * 200 g di tonno in scatola (soffritta o alla olio di oliva)* 1 avocado* 100 g di pomodorini pachino* 1/2 cetriolo* 1/2 lime* 1/2rugula* 1/2 cipollotto* 2-3 fogli di coriandolo fresco* Sale e pepe nero fresco * Olio di oliva extravergine

	Inizia preparando l'insalata: taglieggia il cetriolo e i pomodorini a pezzetti, schiacciandoli delicatamente per favorire l'assorbimento del succo. Taglia a julienne l'avocado, quindi sfilettando la metà con la pellicola, e aggiungi il tonno sbriciolato. In una ciotola, mescola tutti gli ingredienti, aggiungendo il lime spremuto, larugula, il cipollotto triturato, il coriandolo e una generousizza di olio di oliva. Sale e pepe a piacere. Questo piatto è fresco, leggero e brimmingofavorisce il consumo di pesce, ricche di sostanze nutrienti. Potrai accompagnare l'insalata con una spremuta di arancia o un tè freddo per una pausa pranzo sana e gustosa.

Question #3:

Consigliami un posto da visitare in Italia	
Mistral-7B-Instruct-v0.2	I'd be happy to help you with that! Italy is a beautiful country filled with amazing places to visit. One place that I would highly recommend is the Amalfi Coast. Known for its stunning cliffs, colorful towns, and crystal-clear waters, the Amalfi Coast offers breathtaking views and unique experiences. You can explore charming villages like Positano, Ravello, and Amalfi itself, enjoy delicious Italian cuisine, or take a boat tour along the coastline. It's truly a must-visit destination in Italy.Prendersi una cena alla Trattoria da Remigio e degustare il cacio e pepe alla romana.
LLama7b Chat	Oh, that's a tough choice! Italy has so many amazing places to visit. But if I had to recommend just one place, I would say Rome. It's the Eternal City and home to incredible history, architecture, art, food... everything! You could spend weeks there and still not see everything. Must-see attractions include the Colosseum, Vatican City, Pantheon, Trevi Fountain, and of course, indulging in delicious Italian cuisine. Buon viaggio!
Zephyr 7b Chat	Italy is a beautiful country with countless places to visit, so choosing just one can be difficult! However, one place that is often considered a must-see is the city of Rome. Rome is the capital of Italy and is known for its rich history, stunning architecture, and vibrant culture. Some of the must-see attractions in Rome include the Colosseum, the Roman Forum, the Pantheon, the Trevi Fountain, and the Vatican City, home to St. Peter's Basilica and the Sistine Chapel. In addition, Rome is also known for its delicious food, including pizza, pasta, and gelato. If you're looking for a unique experience, you might also consider visiting Cinque Terre, a group of five colorful coastal towns on the Italian Riviera. T.hese towns are known for their picturesque landscapes, stunning sunsets, and delicious seafood

	Ultimately, the best place to visit in Italy will depend on your interests and preferences. Consider your budget, the time of year you're visiting, and what you're hoping to experience during your trip. No matter where you choose, Italy is sure to be an unforgettable destination.

Question #4:

Che cosa è una SRL?	
Mistral-7B-Instruct-v0.2	Una Società a Responsabilità Limitata (Srl) è un tipo di società commerciale italiana, simile ad una Ltd in Inghilterra o ad una LLC negli Stati Uniti. La responsabilità delle azioni dei soci è limitata alle loro quote di capitale sociale. Questo significa che i soci non possono essere tenuti responsabili per debiti e obbligazioni della società oltre al proprio contributo di capitalizzazione.
LLama7b Chat	Una Srl (Società a Responsabilità Limitata) è un tipo di societario italiano che rappresenta una società privata con il limitamento della responsabilità dei Soci (capi soci o azionisti). In altre parole, i Soci sono solidali col patrimonio sociale e possono essere tenuti civili per le obbligazioni sociali e finanziarie, ma la loro responsabilità economica è limitata ai mezzi sociali disponibili.
Zephyr 7b Chat	SRL (Società a Responsabilità Limitata) è una forma societaria adottata in diversi paesi europei, tra cui Italia, Francia, Svizzera e Grecia. Si tratta di una società che presenta alcune similitudini con la società a responsabilità limitata (Ltd) in Gran Bretagna e con la società a responsabilità limitata (GmbH) in Germania. La SRL è una società privata, con azioni non quotate in borsa, e ha uno o più soci (azionisti), che possono essere anche soci-dirigenti. I soci sono responsabili solo per le proprie azioni e sono tutelati dalla separazione tra il patrimonio personale e quello aziendale, ovvero il loro patrimonio personale non può essere requisito per soddisfare eventuali debiti della società. L'amministrazione e la gestione quotidiana della SRL sono affidata a uno o più soci-dirigenti, detti anche amministratori, che sono responsabili della gestione delle attività della società e delle sue decisioni finanziarie. La SRL è una forma societaria relativamente flessibile, che offre una certa tutela ai soci e una certa autonomia nella gestione aziendale. Tuttavia, non offre la possibilità di poter avere soci esterni alla società, né la possibilità di ottenere quote azionarie senza partecipare attivamente alla gestione della società. In sintesi, la SRL è una forma societaria con responsabilità limitata che offre ai soci una certa tutela personale, con un'amministrazione e una gestione affidata a soci-dirigenti che sono responsabili della gestione delle attività della società e delle sue decisioni finanziarie.

Question #5:

Quanto fa 10 + 10 + 20?	
Mistral-7B-Instruct-v0.2	La risposta è 30. In matematica, quando si sommano due o più numeri insieme, si può usare la regola della somma per ottenere la risultato finale. Quindi, 10 + 10 = 20 e quindi 10 + 10 + 20 = 30.
LLama7b Chat	Sure! The answer to 10 + 10 + 20 is 40.
Zephyr 7b Chat	La somma di 10 + 10 + 20 è uguale a 40.

5.1.4 Analysis of result & Final Choice:

Before discussing the choice and results, some considerations need to be made. Firstly, the benchmark results should be taken with a grain of salt for these models due to various factors. This is because the datasets used for evaluations may be more suitable for one model than another, depending on the specifics, but this does not directly mean that models with worse results are less reliable or trustworthy.

A second consideration should be made regarding tests conducted on models in the Italian language. Indeed, these models represent a version trained for the specific purpose of fulfilling instructions starting from pre-trained versions. This means that everything depends on how the model fine-tuning was performed and the datasets used, and therefore, they do not constitute a definitive solution but only examples of what is possible to achieve with the base model.

This means that the choice of the model, despite various precautions, may not be the best in terms of performance, and we must rely on our collective knowledge of the various models and not just on the results obtained through benchmarks and evaluations.

For this reason, considering that Mistral7B, LLama2, and Zephyr7B represent the state-of-the-art in open-source Large Language Models, and considering that each has its advantages and disadvantages, the proposed solution is to use the Mistral model to fine-tune ITACA.

Mistral, in fact, is the most recently developed model trained from scratch. Zephyr, on the other hand, is simply another model already fine-tuned and prepared for user interaction but still uses Mistral as its base model. Since the knowledge injected into Zephyr may not be necessary for our purposes (see the evaluation through inference in paragraph 5.1.4), it has been discarded as a solution, despite achieving the best performance in terms of benchmarks.

LLama2, although more or less on par with Mistral, still represents its predecessor. Moreover, while LLama2 was developed by an American company, Mistral2 comes from a French company, which may have been more committed to using European languages during the training phase. This is confirmed directly from their official documentation, where we can see that LLama2 is not recommended for fine-tuning in languages other than English:

*“**Out-of-scope Uses** Use in any manner that violates applicable laws or regulations (including trade compliance laws).Use in languages other than English. Use in any other way that is prohibited by the Acceptable Use Policy and Licensing Agreement for Llama 2.”*

Furthermore, even though Llama2 has a commercial license, it uses a proprietary one, which could potentially pose issues in case of commercial use of the fine-tuned model.

Finally, we are going to the next step with Mistral 7b Finetuning.

5.2 Data Loading, Cleaning and Preprocessing

Although we have invested a significant amount of time in constructing the two datasets on which we will train the model, we cannot inject them directly as they are. No dataset can be fed into training or fine-tuning a model without first being processed to make it understandable and suitable for a specific task.

In this paragraph, we will delve into the techniques used for data preparation and cleaning. We will then discuss the concepts of text formatting and tokenization, both essential in the context of an Instruction-type Large Language Model.

5.2.1 Loading and merging datasets:

The datasets were loaded using the Pandas library, which allows us to easily read and load files in JSON format, taking care to replace any possible errors during decoding (utf-8 by default).

```
import pandas as pd
import unicodedata

df_ADE = pd.read_json('ADE.json',encoding_errors='replace')
df_ITACA_V1 = pd.read_json('ITACA_DATASET_V1.json',
encoding_errors='replace')
```

```
df_ITACA_V2 = pd.read_json('ITACA_DATASET_V2.json',
encoding_errors='replace')
df_ITACA_NEGATIVE = pd.read_json('ITACA_NEGATIVE.json',
encoding_errors='replace')
```

The datasets were then preprocessed to obtain the final version that we will use during the fine-tuning phase:

```
df_ADE["Dataset"] = "ADE"
df_ITACA_V1["Dataset"] = "ITACA_V1"
df_ITACA_V2["Dataset"] = "ITACA_V2"
df_ITACA_NEGATIVE["Dataset"] = "ITACA_OUTOFSCOPE"
df = pd.concat([df_ADE,df_ITACA_V1,df_ITACA_V2, df_ITACA_NEGATIVE],
ignore_index=True)
```

Result is shown here:

```
Size = 179767, Shape = (25681, 7)
```

Below, we can see some examples that compose it. Note that the "Argomento" field was added only later, in the second version of the ITACA dataset, which is why it has no value for the other datasets.

	Titolo	Risposta	Data	Fonte	Domanda	Dataset	Argomento
1248	Ivie: calcolo e versamento	Le persone fisiche residenti in Italia devono ...	20 Gennaio 2017	https://www.fiscooggi.it/posta/ivie-calcolo-e-...	Sono un cittadino italiano, residente in Itali...	ADE	NaN
2512	Interessi mutui agrari: detrazione Irpef	Dall'Irpef lorda è possibile detrarre un impo...	8 Gennaio 2019	https://www.fiscooggi.it/posta/interessi-mutui...	Per la detrazione relativa agli interessi di u...	ADE	NaN
15179	Detrazione premi assicurazione al 19: limiti e...	Il codice da riportare nella dichiarazione dei...	2 Maggio 2023	https://fiscomania.com/detrazione-polizze-assi...	Quale codice deve essere riportato nella dichi...	ITACA_V2	Dichiarazione dei redditi
17146	Cash pooling: di cosa si tratta?	Si possono notare tutti i benefici e vantaggi ...	11 Febbraio 2022	https://fiscomania.com/cash-pooling/	Quando si può notare tutti i benefici e i vant...	ITACA_V2	NaN
20233	Assegno circolare: cos'è e come utilizzarlo	Per incassare immediatamente un assegno circol...	15 Marzo 2024	https://fiscomania.com/assegno-circolare/	Come posso incassare immediatamente un assegno...	ITACA_V2	Fisco

5.2.2 Data Cleaning:

Often, both when data is retrieved through automated tools and when it is manually collected, small errors are made that, however, in excessive quantities, can compromise the outcome of fine-tuning. It is therefore extremely necessary to ensure that such errors are corrected before moving on to subsequent phases.

Consider, for example, ChatGPT3, for which an initial dataset of over 45 Terabytes was constructed. Once the cleaning phase was completed, the size of the dataset shrank to the "insignificant" figure of 570GB, practically 1% of the starting point.

The data cleaning process can vary from situation to situation, depending on how the dataset data was retrieved and constructed, but generally aims to normalize the text, remove HTML tags, filter keywords, eliminate duplicates, correct spelling errors, etc.

In many cases, since this data comes from all over the web, and very often it is not from reliable sources, this phase is used to try to eliminate those samples with clear false or imprecise information. In our case, using our LLMDSGenerator allowed us to start from an advantageous position, being able to rely on the fact that only we had defined, from the beginning, the sources from which to draw for the creation of the synthetic dataset.

Therefore, leaving aside this point, first, we can remove any duplicates. There are many techniques, both simple and very complex, for this phase. For example, often one not only deals with eliminating duplicates but also with eliminating samples that are highly similar to each other.

For our case, it was not deemed necessary to go into this detail given the modest size of the starting dataset and the diversity of topics covered. However, we limited ourselves to finding and eliminating those duplicates consisting of the same question and answer:

```
df_droplog = pd.DataFrame()
mask = df.duplicated(keep=False)
df_keep = df.loc[~mask]
df_droplog = pd.concat([df_droplog,df.loc[mask]])
df_droplog.head(6)

df = df.drop_duplicates()
```

	Titolo	Risposta	Data	Fonte	Domanda	Dataset	Argomento
3758	Pensione anticipata 2024: cos'è e come ottenerla?	La Legge di Bilancio 2024 ha deciso di esclude...	18 Dicembre 2023	https://fiscomania.com/pensione-anticipata/	Quali sono le modifiche introdotte dalla Legge...	ITACA_V1	NaN
3759	Pensione anticipata 2024: cos'è e come ottenerla?	La Legge di Bilancio 2024 ha reintegrato la po...	18 Dicembre 2023	https://fiscomania.com/pensione-anticipata/	Come è possibile accedere alla pensione antici...	ITACA_V1	NaN
3763	Pensione anticipata 2024: cos'è e come ottenerla?	La Legge di Bilancio 2024 ha deciso di esclude...	18 Dicembre 2023	https://fiscomania.com/pensione-anticipata/	Quali sono le modifiche introdotte dalla Legge...	ITACA_V1	NaN
3764	Pensione anticipata 2024: cos'è e come ottenerla?	La Legge di Bilancio 2024 ha reintegrato la po...	18 Dicembre 2023	https://fiscomania.com/pensione-anticipata/	Come è possibile accedere alla pensione antici...	ITACA_V1	NaN
3864	Assegno unico 2024: calcolo, importi e benefic...	L'assegno unico ed universale per i figli a ca...	11 Dicembre 2023	https://fiscomania.com/assegno-unico/	Qual è lo scopo dell'assegno unico ed universa...	ITACA_V1	NaN
3874	Assegno unico 2024: calcolo, importi e benefic...	L'assegno unico ed universale per i figli a ca...	11 Dicembre 2023	https://fiscomania.com/assegno-unico/	Qual è lo scopo dell'assegno unico ed universa...	ITACA_V1	NaN

Subsequently, all examples with at least one of the two fields necessary to create the text to inject into the model, not filled in, namely the "Domanda" and "Risposta" fields, were deleted.

```
df = df.dropna(subset=['Domanda','Risposta'])
```

Finally, we remove all samples containing certain keywords that we know we do not want to use for training. Additionally, we eliminate all samples that have dirty characters not convertible to ASCII format:

```
# Remove samples containing missleading information
remove_dictionary = ["*****", "******", … , "********"]
mask = df.apply(lambda row: any(word in row['Domanda'].lower() or word in
row['Risposta'].lower() for word in remove_dictionary), axis=1)
df = df[~mask]
# Remove any un-readable characters
df.loc[df['Domanda'].str.contains(r'[^\x00-\x7F]+') == False]
```

5.2.3 Data preprocessing: Format and tokenize the text.

We have seen how important it is to have a clean dataset to obtain a reliable and quality result, but it is not the only thing to consider during the preparation of our data. The dataset cannot yet be injected, or at least not to achieve our ultimate goal, which is to train the model so that it is able to respond to user requests.

We must remember that models are not born ready to solve instructions or chat with the end user, but they are Natural Language Programming (NLP) models, which we discussed in detail in Chapter 2, that is, models that have the ability to predict the next word based on the reference context.

For this reason, it is necessary to prepare the model so that the task is read similarly to completing a text, but at the same time, train it so that the result appears as fulfilling a request.

In this sense, we will not inject a question-answer pair, but a single text to provide to the model, which will correspond to the concatenation of the question and the answer.

Since the starting base for fine-tuning the model is not already a model based on instructions, it is not necessary to follow a specific format type, because we will train it so that it can follow a certain format indicated by us.

However, we refer to some of the most common formats currently available, namely the ALPACA format and the MISTRAL format:

Alpaca Format:
{***Instruction***}
Input:
{***prompt***}
Response:
{***completion***}

Mistral Format:
<s>[INST] {***Instruction***}[/INST]
{***completion***}</s>
[INST] {***Instruction2***}[/INST]
{***completion2***}</s>

As we can see, Alpaca includes a section for the instruction where the task is specified, a section for the input, which is optional and can be used to provide context from which to draw, and finally the section for the response that the model should generate.

On the other hand, Mistral follows a different format, based on tokens that delineate the start and end of an instruction ([INST] and [\INST]) and the start and end of a conversation ([<s> and </s>]). Additionally, we notice how this format is also suited for a conversation, as it allows for the insertion of multiple instructions optionally.

As we mentioned, there is no valid reason to choose one format over another. However, the choice fell on the Mistral format, given the decision to use the eponymous model as the base model.

To prepare the dataset in the following format, we used the following function::

```
my_prompt = "<s>[INST] Sei un assistente utile ed affidabile. Rispondi in maniera adeguata alla domanda seguente:\n{} [/INST]\n\nRisposta: "

def formatting_prompt_func_with_answers(example):
    question = example["Domanda"]
    example["text"] = my_prompt.format(question) + example["Risposta"] + " </s>"
    return example
```

```
train = dataset['train'].map(formatting_prompt_func_with_answers)
eval = dataset['test'].map(formatting_prompt_func_with_answers)
```

This function simply iterates over the examples in our dataset and uses the "Domanda" and "Risposta" fields to construct unique samples of type 'text' that will adhere to the format we have prepared. We notice that a default system prompt has also been added for guided completion of the model to generate a response. Here is an example:

<s>[INST] Sei un assistente utile ed affidabile. Rispondi in maniera adeguata alla domanda seguente:

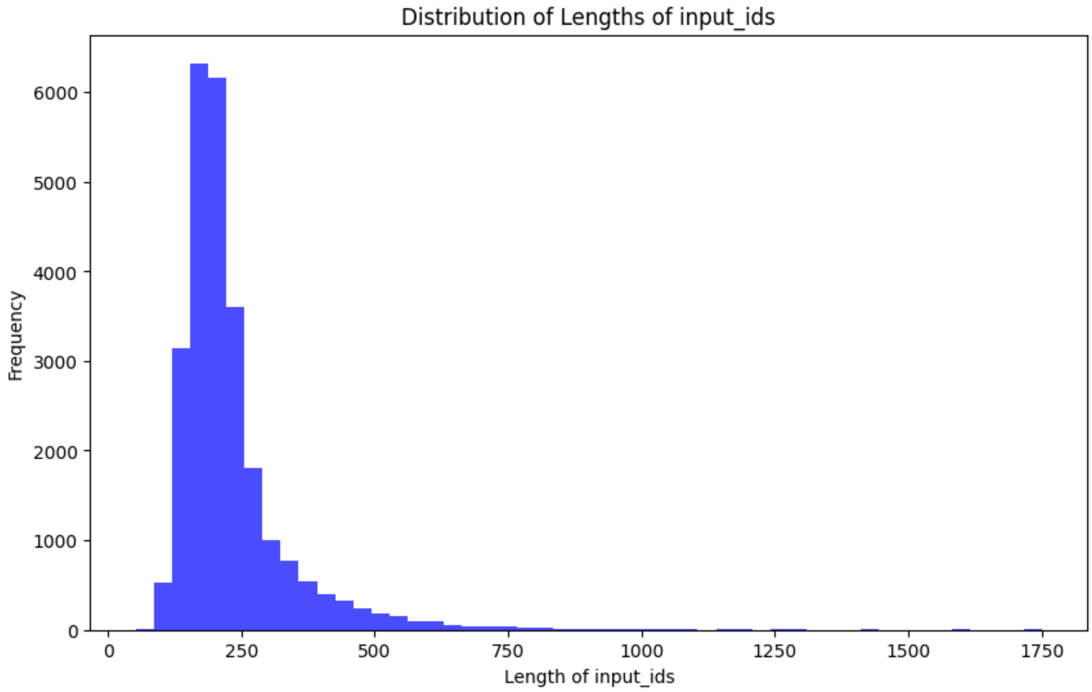
Come viene scoperta l'evasione contributiva? **[/INST]**

Risposta: L'INPS può scoprire l'evasione contributiva attraverso controlli, ispezioni sul posto di lavoro o attraverso segnalazioni. Una volta rilevate irregolarità, invierà una comunicazione al datore di lavoro che avrà un termine specifico, solitamente 30 giorni, per regolarizzare la situazione. **</s>**

In the data preparation process for fine-tuning a Large Language Model, another important step is tokenization of the data. A tokenizer is responsible for preparing the data to be more understandable for the model. The HuggingFace library provides tokenizers for each of its models.

During this process, two additional tasks must be performed: truncation of data according to the maximum token limit imposed, and adding padding for cases that do not reach the same value. When using the HuggingFace trainer, this work during model fine-tuning is transparent. We just need to specify the max_token_length to the trainer used, and this process will be handled under the hood.

To avoid defining this value arbitrarily, we relied on the use of the tokenizer to have a clearer view of the distribution of our samples based on tokens.:



It is evident that the majority of samples within our dataset do not exceed 512 tokens, which is why this length has been set as the maximum. This minimum token length will allow us to have faster and more efficient training.

5.3 Fine-Tuning Process with (Q)LoRA

Fine-tuning large pre-trained models presents significant computational challenges due to the need to adjust millions of parameters. This traditional approach demands substantial computational resources and time, posing a bottleneck for adapting models to specific tasks. Efficient resource utilization and cost-effectiveness are crucial considerations when selecting a fine-tuning strategy.

In addressing these challenges, LoRA offers an effective solution by decomposing the update matrix during fine-tuning. This technique optimizes resource usage and enhances efficiency.

This chapter delves into the most popular and effective variant of parameter-efficient methods: Low Rank Adaptation (LoRA), with a particular focus on QLoRA. QLoRA represents an even more efficient variant of LoRA, showcasing advancements in fine-tuning techniques for large pre-trained models.

5.3.1 LoRA and QloRA Introduction

In the domain of language models, fine-tuning an existing model to perform a specific task on particular data is a prevalent practice. This typically involves incorporating a task-specific head, if needed, and adjusting the neural network's weights through backpropagation during training.

It's crucial to distinguish this fine-tuning process from training from scratch. In the latter scenario, the model's weights are initialized randomly, whereas during fine-tuning, the weights are already optimized to some extent from the pre-training phase. The decision of which weights to optimize or update, and which ones to keep frozen, depends on the chosen technique.

Full fine-tuning entails optimizing or training all layers of the neural network. While this approach often yields superior results, it is also the most resource-intensive and time-consuming.

Fortunately, parameter-efficient approaches for fine-tuning have emerged as effective alternatives. While most of these methods have sacrificed some performance, Low Rank Adaptation (LoRA) has defied this trend by occasionally surpassing full fine-tuning. This success is attributed to LoRA's ability to prevent catastrophic forgetting, a phenomenon where the knowledge of the pretrained model is lost during fine-tuning.

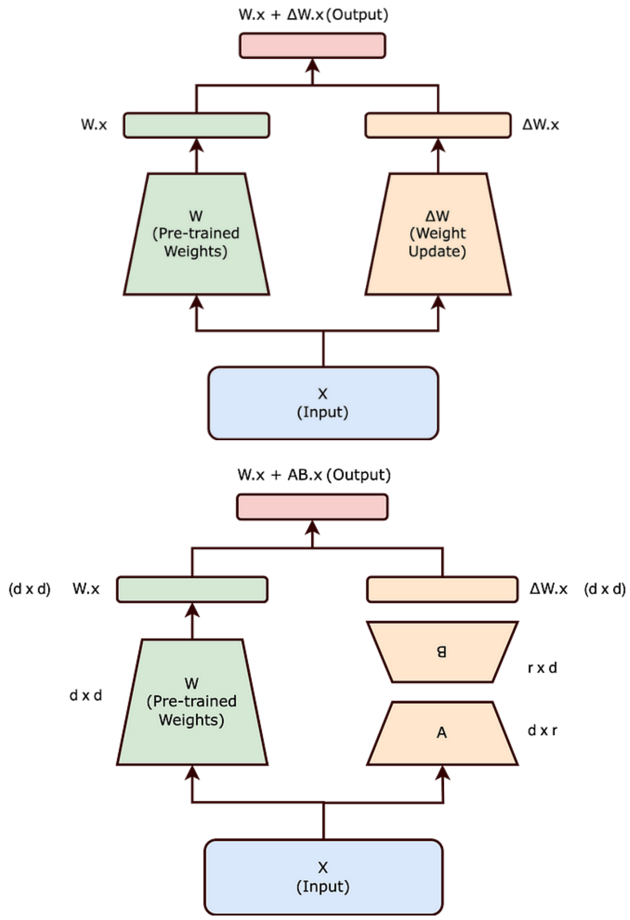
LoRA improves fine-tuning by fine-tuning two smaller matrices that approximate the larger weight matrix of the pre-trained language model, constituting the LoRA adapter. This fine-tuned adapter is then integrated into the pretrained model for inference.

QLoRA represents an even more memory-efficient iteration of LoRA. It achieves this by loading the pretrained model to GPU memory with quantized 4-bit weights, compared to 8-bit weights in LoRA, while maintaining similar effectiveness. The focus here will be on probing this method, comparing it with LoRA when necessary, and determining the optimal combination of QLoRA hyperparameters for achieving peak performance with minimal training time.

LoRA is conveniently implemented in the Hugging Face Parameter Efficient Fine-Tuning (PEFT) library, offering ease of use. QLoRA can be leveraged by combining bitsandbytes and PEFT. Moreover, the HuggingFace Transformer Reinforcement Learning (TRL) library provides a seamless trainer for supervised fine-tuning with integrated support for LoRA. These libraries collectively equip users with the necessary tools to fine-tune the chosen pretrained model effectively, enabling the generation of coherent and convincing product descriptions based on specified attributes.

5.3.2 How does LoRA works?

In the traditional fine-tuning process, adjustments to a pre-trained neural network's weights are made directly to its weight matrix (W) to tailor it to a new task. These modifications, denoted as ΔW, result in updated weights expressed as (W + ΔW). However, the LoRA approach introduces a novel strategy by decomposing ΔW, thereby mitigating the computational burden associated with fine-tuning large models.



By decomposing ΔW into matrices A and B, both of lower rank (r), the number of trainable parameters is significantly reduced. For instance, if W is a (d x d) matrix, conventional updating of W involves (d^2) parameters. In contrast, with matrices A and B sized (d x r) and (r x d) respectively, the total parameter count reduces to (2dr), a substantial reduction when (r << d).

The reduction in trainable parameters achieved through Low-Rank Adaptation (LoRA) offers several noteworthy benefits, particularly for fine-tuning large-scale neural networks:

1. Reduced Memory Footprint: LoRA diminishes memory requirements by minimizing the number of parameters to update, facilitating the management of large-scale models.

2. Faster Training and Adaptation: Simplifying computational demands accelerates the training and fine-tuning processes for large models, enhancing efficiency.

3. Feasibility for Smaller Hardware: LoRA's reduced parameter count enables fine-tuning of substantial models on less powerful hardware, such as modest GPUs or CPUs.

4. Scaling to Larger Models: LoRA enables the scaling of AI models without a proportional increase in computational resources, streamlining the management of growing model sizes.

In the realm of LoRA, the concept of rank plays a crucial role in determining the efficiency and effectiveness of the adaptation process. Notably, the paper highlights that the rank of matrices A and B can be exceptionally low, sometimes as low as one.

While the LoRA paper primarily focuses on experiments within Natural Language Processing (NLP), the underlying approach of low-rank adaptation holds broad applicability and could effectively enhance the training of various neural network architectures across different domains.

5.3.3 Make more efficient LoRA with QloRA

QLoRA, an extension of LoRA, endeavors to compress the weight parameters of pre-trained large language models (LLMs) into 4-bit precision. This compression significantly reduces the model's memory footprint, facilitating fine-tuning on a single GPU and rendering it viable for operation on less potent hardware.

QLoRA introduces several innovative techniques to minimize memory usage while preserving performance:

1. **4-bit NormalFloat (NF4)**: This data type, tailored for normally distributed data, empirically outperforms 4-bit Integers and 4-bit Floats. NF4 optimally quantizes data for neural networks, maintaining accuracy within the given bit constraint.

2. **Double Quantization**: Quantizing the quantization constants saves an average of about 0.37 bits per parameter, notably reducing memory overhead, especially for expansive LLMs.

3. **Paged Optimizers**: Leveraging NVIDIA unified memory, this technique manages memory spikes during training, preventing GPU memory exhaustion. Optimizer states are dynamically moved between CPU RAM and GPU memory as required.

In detail:

- **4-bit Normal Float Quantization (NF4)**: NF4 efficiently quantizes neural network weights into a 4-bit format, employing a method called "Quantile Quantization" to ensure optimal quantization for normally distributed data.

- **Double Quantization**: This strategy further optimizes memory usage by quantizing the quantization constants, effectively reducing the memory footprint.

- **Dequantization**: During training, quantized weights are converted back to their original data types (e.g., from 4-bit NormalFloat to 32-bit Float) for accurate computations, particularly during gradient calculation.

- **Paged Optimizers**: By managing memory usage during training of large LLMs, this approach addresses memory spikes by utilizing NVIDIA unified memory for automatic page-to-page transfers between CPU and GPU.

Overall, QLoRA presents a holistic approach to efficiently compressing and managing memory usage in large language models, facilitating their deployment on less powerful hardware without compromising performance.

6. Experiments, evaluation and comparation

The process of fine-tuning a model can drastically improve its performance and make it more effective for a specific task. In this regard, the model's hyperparameters play a crucial role during this process and determine the success of this important step.

In this chapter, we will delve into the model training phases, selecting and analysing different hyperparameter configurations, evaluating the model's responses, and seeking the most suitable situation for our purpose. Initially, we will look at the configuration of the environment used for the experiments and model training, ensuring that this process can be replicated smoothly. We will then analyse the various configurations used before the actual fine-tuning of the model, discovering which one best fit our case. Finally, we will evaluate the model and compare it with the currently available open-source models.

6.1 Environments and dependencies

Setup development environments is important to have a right base to develop and finetune our model. Thanks to the choice performed during our journey, we don't need to ask for an excessive computer power. In this paragraph, we are going to explain all the environment configuration in order to understand how can be easily replicate an experiment like this one.

In addition, a detailed introduction and explanation will be supplied about the library and dependencies used during the process.

6.1.1 Environment and Power Computing

Despite aiming to minimize resource usage and compute power for fine-tuning a 7B model, access to a basic GPU remains essential. To address this need, we leveraged platforms such as Colab and Kaggle, which offer GPU resources at no cost, albeit with certain limitations.

However, as the project progressed, the final fine-tuning required more robust computational power, leading us to rent a NVIDIA 4090 GPU.

To facilitate the fine-tuning process, we configured a docker image with the necessary packages, including:

- PyTorch 2.01
- CUDA 11.7
- cuDNN 8
- Full CUDA toolkit (including nvcc)

The cost of renting the GPU amounted to approximately $0.411 per hour, with the fine-tuning process completing within 2 hours and 30 minutes for three epoch. This translates to a total cost of approximately $1.233, ensuring cost-effectiveness in completing the task efficiently.

6.1.2 Hugging Face

The Hugging Face Hub stands as a cornerstone platform boasting over 350k models, 75k datasets, and 150k demo apps (Spaces), all of which are open source and publicly accessible. It serves as an online nexus where individuals can seamlessly collaborate and build machine learning solutions together. This expansive hub offers a fertile ground for exploration, experimentation, collaboration, and technological advancement in the realm of machine learning.

At the heart of Hugging Face's offerings lies a suite of immensely valuable libraries designed to streamline data preprocessing, fine-tuning, and beyond. Among these, the Transformers library reigns supreme, offering APIs that enable users to effortlessly download and utilize pretrained models, fine-tune them on custom datasets, and subsequently share their creations with the wider community via the model hub. Each Python module defining an architecture within Transformers is fully modular, allowing for quick modifications to facilitate rapid research experiments.

Transformers enjoys robust backing from the three most popular deep learning libraries—Jax, PyTorch, and TensorFlow—with seamless integration among them. This facilitates smooth transitions between training models with one framework and deploying them for inference with another, ensuring maximum flexibility and ease of use.

There are myriad compelling reasons to embrace the Transformers library:

1. **Easy-to-use state-of-the-art models**: Delivering high performance across a spectrum of tasks including natural language understanding & generation, computer vision, and audio tasks, Transformers boasts a low barrier to entry for both educators and practitioners. With just three classes to learn, users can leverage a unified API to harness all pretrained models.

2. **Lower compute costs, smaller carbon footprint**: By enabling researchers to share trained models instead of consistently retraining them, Transformers drives down compute time and production costs. With dozens of architectures and over 400,000 pretrained models spanning all modalities, practitioners can access a vast repository of resources to meet their needs.

3. **Choose the right framework for every stage of a model's lifecycle**: With the ability to train state-of-the-art models in just three lines of code, Transformers empowers users to seamlessly transition a single model between TF2.0/PyTorch/JAX frameworks as needed for training, evaluation, and production.

4. **Easily customize models to suit specific requirements**: Transformers provides examples for each architecture to reproduce results published by original authors, while

exposing model internals in a consistent manner. Moreover, model files can be used independently of the library for quick experimentation, affording users maximum flexibility and control.

TRL (Transformer Reinforcement Learning) complements the Transformers ecosystem as a full-stack library designed to facilitate training transformer language models with reinforcement learning. TRL encompasses tools for supervised fine-tuning (SFT), reward modeling (RM), and proximal policy optimization (PPO), all seamlessly integrated with Transformers. Supervised fine-tuning, a pivotal step in TRL, is made accessible through an easy-to-use API, enabling users to effortlessly create and train models on custom datasets with minimal code.

6.1.3 Unsloth

Unsloth emerges as a lightweight library tailored for accelerating fine-tuning of Large Language Models (LLMs), seamlessly integrated within the Hugging Face ecosystem encompassing Hub, Transformers, PEFT, and TRL. Developed collaboratively by the Unsloth team led by Daniel and Michael alongside the open-source community, this library caters to the need for swift LLM fine-tuning without compromising performance.

The versatility of Unsloth extends to most NVIDIA GPUs, ranging from GTX 1070s to H100s, offering compatibility across a broad spectrum of hardware configurations. At present, Unsloth supports the Llama (CodeLlama, Yi, etc.) and Mistral architectures, catering to diverse model requirements.

Unsloth operates by optimizing critical sections of the modeling code, implementing manual derivation of backpropagation steps, and rewriting all PyTorch modules into Triton kernels. This innovative approach not only reduces memory usage but also accelerates fine-tuning processes significantly. Importantly, Unsloth achieves these enhancements without sacrificing accuracy, as no approximations are made in the optimized codebase, ensuring a 0% accuracy degradation compared to normal QLoRA fine-tuning.

In essence, Unsloth stands as a powerful tool for practitioners seeking to expedite LLM fine-tuning while maintaining the highest standards of model accuracy and performance.

Let’s see some benchmark:

1 A100 40GB	Dataset	🤗	🤗 + Flash Attention 2	🦥 Unsloth	🦥 VRAM saved
Code Llama 34b	Slim Orca	1x	1.01x	**1.94x**	-22.7%
Llama-2 7b	Slim Orca	1x	0.96x	**1.87x**	-39.3%
Mistral 7b	Slim Orca	1x	1.17x	**1.88x**	-65.9%
Tiny Llama 1.1b	Alpaca	1x	1.55x	**2.74x**	-57.8%

6.1.4 WandB

Weights & Biases (WandB) is a machine learning development platform that allows users to track and visualize various aspects of their model training process in real-time.

In the context of machine learning, WandB is primarily used to:

- Track model performance metrics such as accuracy, loss, and other evaluation metrics during the training and evaluation phases.
- Visualize the model's learning process using graphs, charts, and histograms to gain insights into how the model is performing.
- Compare different models and their performance metrics to help choose the best-performing one.
- Collaborate with others by sharing experiments and results.

6.2 Hyperparameters Tuning

Hyperparameters configuration of the model must be chosen in order to optimize its functioning. The difference, compared to parameters, is that hyperparameters are not directly learned during the training process and cannot be optimized during that phase. They must be defined before training begins.

Exploring all hyperparameters to make the best choice is not possible in the contest of Large Language Model. An immense amount of resource would be used to achieve this result. Therefore, what we will do is choose initial configurations to explore and analyze their behaviour over epochs to avoid wasting resources and time.

Now, let's see in more detail what are the most used hyperparameters for training the model.

6.2.1 Hyperameter Tuning Method

Choosing the right combination of hyperparameters requires an understanding of the hyperparameters and the business use-case. However, technically, there are two ways to set them.

Manual hyperparameter tuning

Manual hyperparameter tuning involves experimenting with different sets of hyperparameters manually i.e. each trial with a set of hyperparameters will be performed by you. This technique will require a robust experiment tracker which could track a variety of variables from images, logs to system metrics.

We will use W&B Framework to manage this complex task and track our progress.

Tuning hyperparameters manually means more control over the process. However, manual tuning is a tedious process since there can be many trials and keeping track can prove costly and time-consuming. In addition, isn't a very practical approach when there are a lot of hyperparameters to consider.

Automated hyperparameter tuning

Automated hyperparameter tuning utilizes already existing algorithms to automate the process. The steps you follow are:

- First, specify a set of hyperparameters and limits to those hyperparameters' values (note: every algorithm requires this set to be a specific data structure, e.g. dictionaries are common while working with algorithms).

- Then the algorithm does the heavy lifting for you. It runs those trials and fetches you the best set of hyperparameters that will give optimal results.

Some common algorithms have been explored in order to better achieve this result like Random Search, Grid Search and other typical algorithms.

We will choose a manual approach since we can't try a lot of different combinations and in order to better analyze all the different configurations explored.

6.2.2 Standard Hyperparameters

Some hyperparameters are more relevant than other in some case and change them could take extremely different result.

So, let's analyse what are the most relevant hypermeters to explore:

Epoch

An "epoch" is a term used to describe one complete pass through the entire training dataset.

In practice, if we need to iterate over a dataset composed by 50.000 samples, one epoch means that the model has had the chance to learn from each samples of our dataset. That practically means that this value changes drastically the performance of the model and the time and resources to use for the training.

A low epoch number can take the model to underfit, which means it could perform poorly because it hasn't learned enough from the training data. In essence, it may not have had enough iterations to effectively learn and adjust its parameters (e.g., weights and biases). A high epoch number means a big risk of overfitting, where the model becomes too specialized in the training data and performs poorly on unseen data.

Learning Rate

The learning rate controls how quickly the model updates its parameters during training. A higher learning rate accelerates learning but may result in instability. A lower learning rate ensures stability but prolongs the training process. Optimal learning rates vary based on the task and model architecture.

Batch Size

Batch size determines how many data samples the model processes in a single iteration. Larger batch sizes can speed up training but require more memory. Smaller batch sizes can help the model thoroughly process each record. The choice of batch size should align with specific hardware capabilities and dataset size.

6.2.3 Lora Adapter Hyperparameters

Being our model trained using LoRA technique, some other hyperparameters need to be set and explored regarding the process of training the adapters. Let's see the detail:

r:

the rank of the A and B matrices

lora_alpha:

this is a pretty controversial parameter. A lot of people have a lot of ideas about it. You can consider it a scaling factor, and by default it should be equal to r, as far as I understand.

target_modules:

the portions of the model we want to optimize with LoRA. the BLOOM module has parameters named query_key_value which we want to optimize.

lora_dropout:

dropout is a technique which hides inputs to suppress the model from overfitting (called regularization). This is a probability of being hidden.

bias:

neural networks typically have two paramet per connection, a "weight" and a "bias". We're only training weights in this example.

task_type:

not super necessary, used in the superclass PeftConfig. Setting to CAUSAL_LM because the specific language model we're using is "causal".

6.4 Experiments & Analysis:

6.4.1 Hyperparameter Proposed Configuration

We explore four different configurations. Let’s see them in detail:

Name	**mistral7b_qlora_itaca_v2_conf0**	**mistral7b_qlora_itaca_v2_conf1**	**mistral7b_qlora_itaca_v2_conf2**	**mistral7b_qlora_itaca_v2_conf3**
per_device_train_batch_size	4	8	8	8
gradient_accumulation_steps	2	4	4	4
learning_rate	0.0002	0.0002	0.00002	0.00002
lr_scheduler_type	linear	linear	cosine	linear
optim	adamw_8bit	adamw_8bit	adamw_8bit	adamw_8bit

train/global_step	150	300	300	300
lora_rank	8	8	8	8
lora_modules	["q_proj", "v_proj"]	["q_proj", "k_proj", "v_proj", "o_proj","gate_proj", "up_proj", "down_proj"]	["q_proj", "k_proj", "v_proj", "o_proj","gate_proj", "up_proj", "down_proj"]	["q_proj", "k_proj", "v_proj", "o_proj","gate_proj", "up_proj", "down_proj"]
lora_dropout	0	0	0	0
qlora	4bit	4bit	4bit	4bit

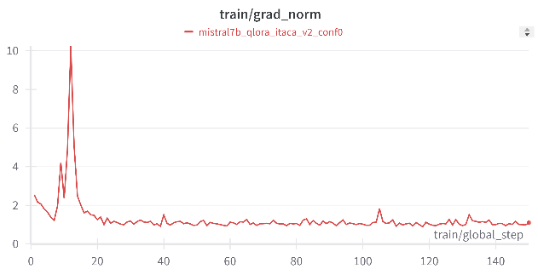
6.4.2 Configuration #0

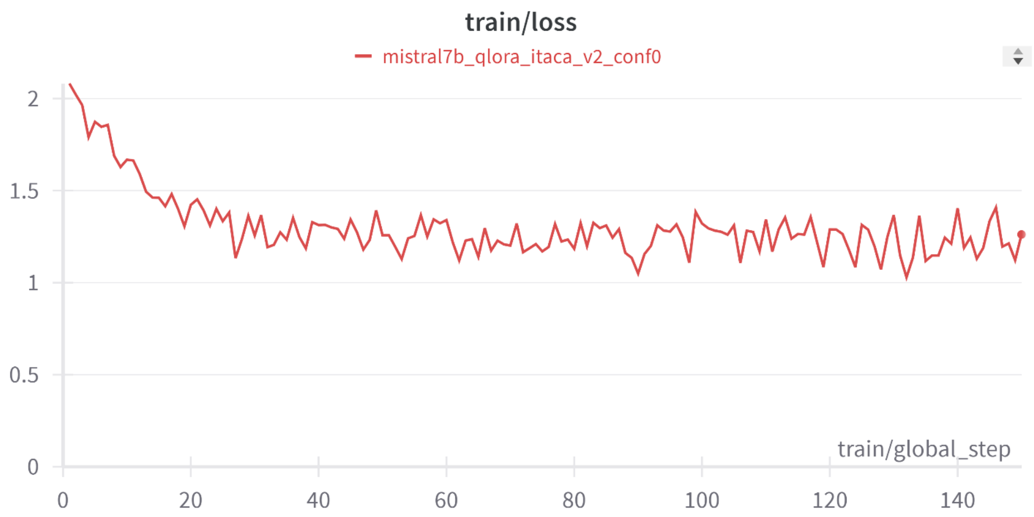
The first configuration analyzed is simply a check of the environment we are using to ensure that it can support the training of our model and that the model itself can learn from the dataset we possess.

For these reasons, it was chosen to begin analyzing the results using values that are almost standard and very common in the case of fine-tuning Large Language Models, but with a relatively small batch size.

It's worth noting that despite setting a size of 4, the gradient_accumulation allows us to update the weights only after two batches have been processed by the model, effectively simulating a larger batch size. For example, having chosen a value of 2 for the gradient accumulation in this case, the final batch size measure for the model during training will be twice the default batch size, hence equal to 8.

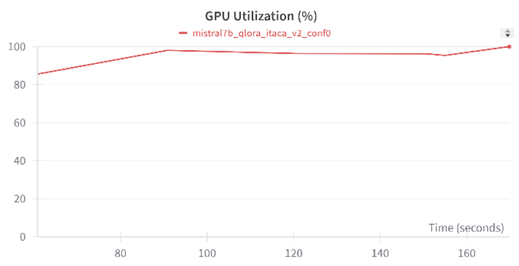
Let's see how the model behaved during a small portion of an epoch:

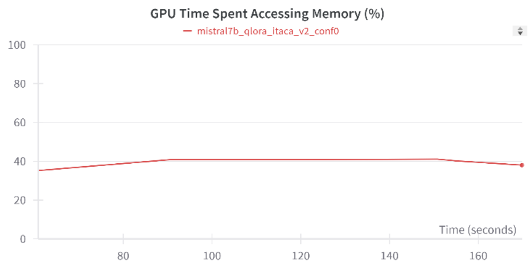


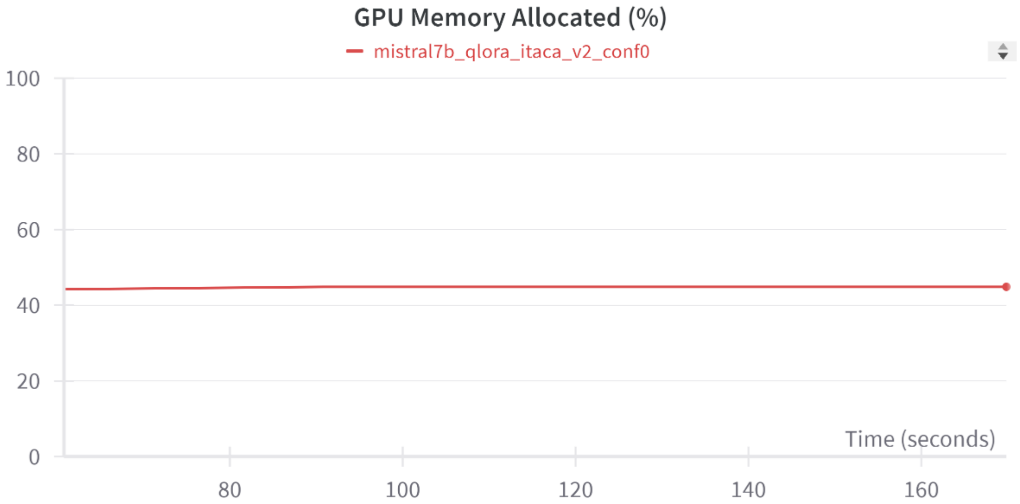


We immediately notice that the model has an immediate declining phase, followed by oscillations around the same average value. This could be a sign of insufficient power compared to what is necessary for learning. Essentially, we're suggesting that this could be a case of underfitting.

For this reason, in the next configuration, we will use all the linear modules of Lora, providing more parameters to train the model. Additionally, we will increase the reference epoch portion so that we can analyze it in more detail, with a lower risk of looking at too small a portion for analysis.



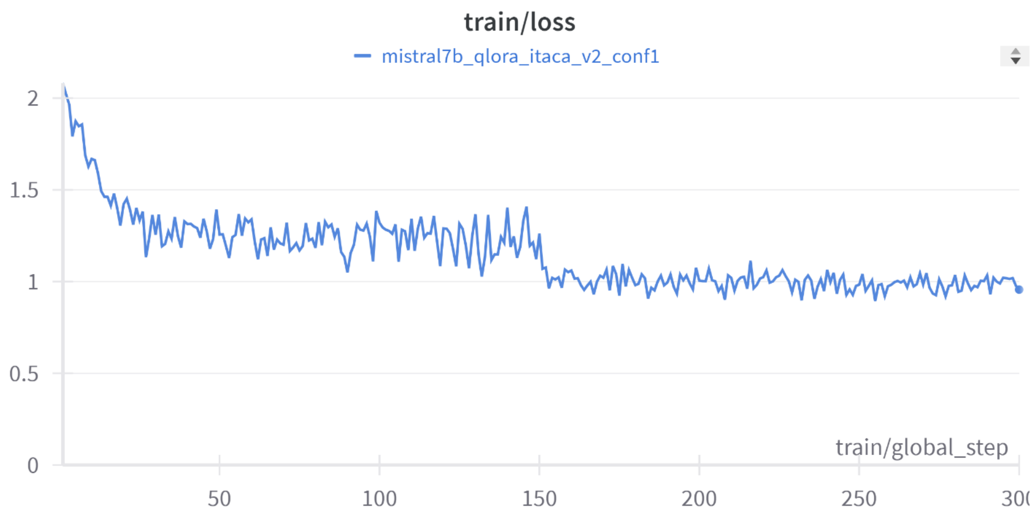




This trial has mainly helped us understand that we have enough computing power to use a larger batch size, improving training performance and reducing time. In fact, despite fully utilizing the GPU, only about 40% of its memory has been allocated.

Therefore, we will increase the batch size and the gradient accumulation accordingly.

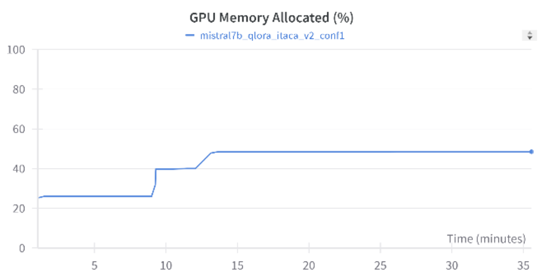
6.4.3 Configuration #1

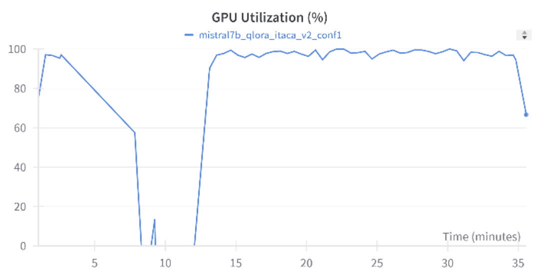


We immediately notice that increasing the portion of the analyzed epoch has provided a much clearer view, reducing the chances of analyzing something that would have changed over time.

The use of linear modules for LoRA seems to have had the desired effect, as the loss continues to decrease, albeit very slowly. This could clearly be due to using a relatively low learning rate. However, it's also true that over an epoch, this behavior should be beneficial, as it is much less likely to get stuck in local minima and oscillate between them.

What we can do next is to choose a different learning rate decay algorithm to analyze its descent:

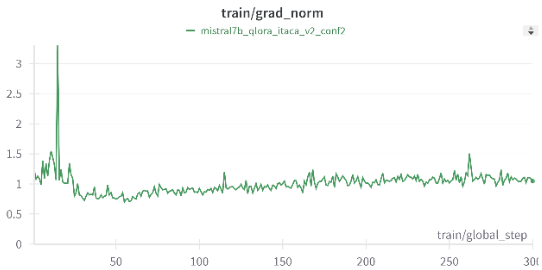


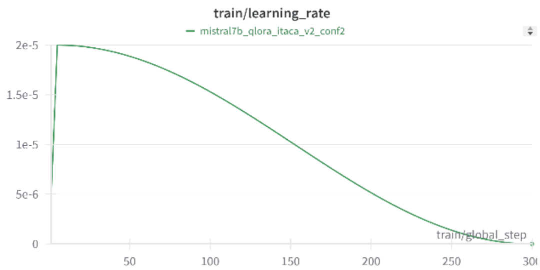


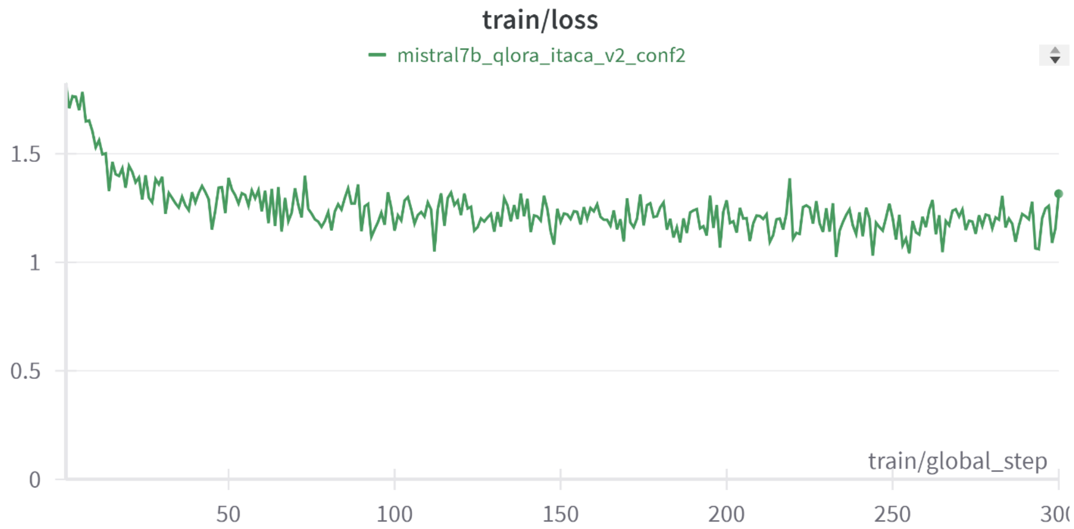
As we can see from these GPU utilization graphs, increasing the allocated resources does not seem to be computationally impossible to manage for the environment used. This has saved us a lot of time during training and resulted in almost better results.

6.4.4 Configuration #3

Let's try to evaluate the training progress using a different learning rate decay in this first configuration. For example, let's use the cosine scheduling decay:





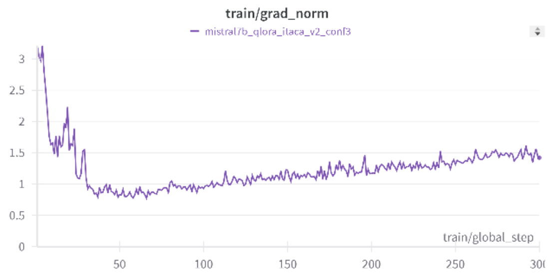


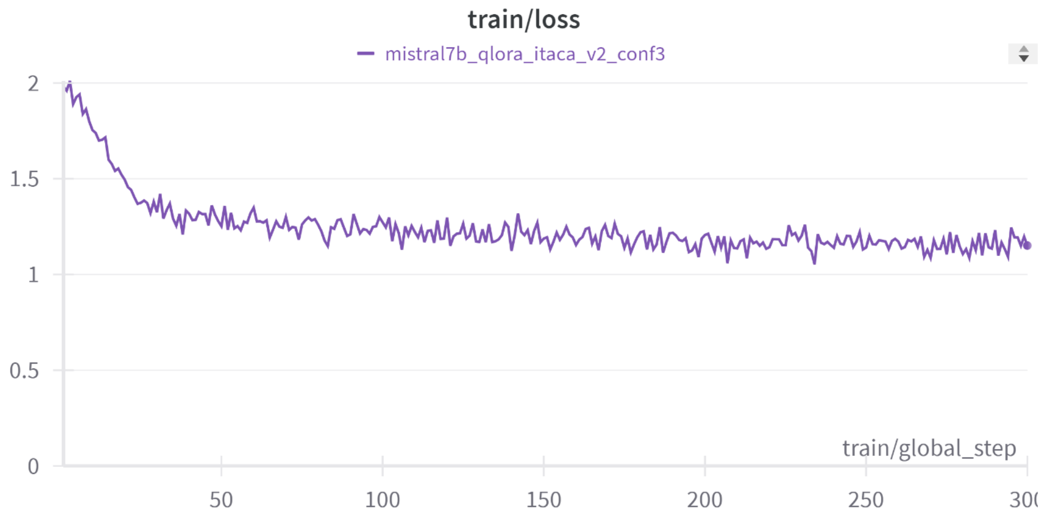
No improvement margins were noticed; indeed, it seems that this type of decay penalized the model's training. The training appears much more oscillatory and seems to stabilize after about 150 steps.

Let's revert to using linear decay, this time trying a lower learning rate. This will help us understand whether it can further stabilize our training or, conversely, yield no positive return.

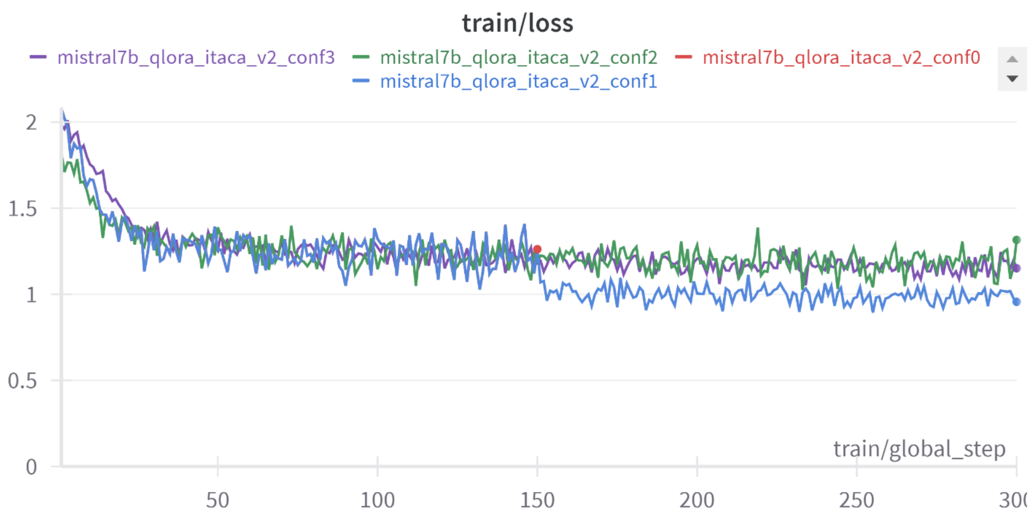
6.4.5 Configuration #4

Let's revert to our previous linear decay and try lowering the learning rate to see if there is any improvement during the model's learning.



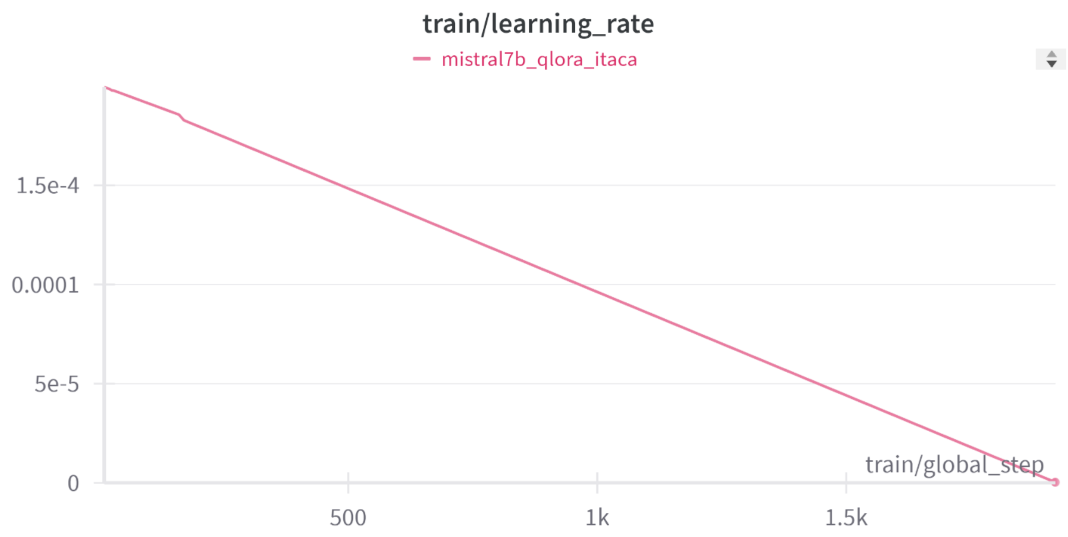


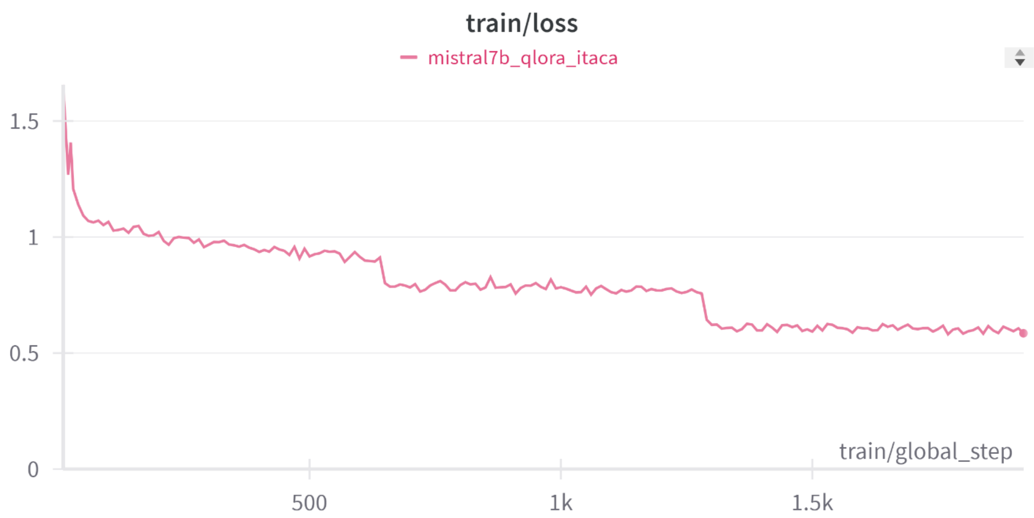
We certainly notice a generally more stable training compared to all the configurations we have witnessed. However, there don't seem to be any advantages compared to the configuration with linear decay and a higher learning rate, except for stability, which in itself results in a starting point of an order of magnitude lower, leading to much higher times and resources.



We can conclude that configuration #1 is the best in terms of the ratio of time/resources used to benefits. We are therefore ready to proceed with the final training of the model.

6.4.6 ITACA Finetuning





At the beginning of the training (from 0 to around 500 steps), there is a sharp decline in the training loss. This suggests that the model is quickly learning from the training data and making significant improvements in its predictions.

After approximately 500 steps, the rate of decline in the training loss slows down. This is a common phenomenon in machine learning, as the model begins to converge to a solution. The training loss continues to decrease but at a slower pace, indicating that the model is still learning and fine-tuning its parameters, but the rate of learning has decreased.

The training loss seems to stabilize between 0.5 and 1. This could be the point of convergence, where the model has learned as much as it can from the training data and any further training would not significantly reduce the loss.

6.5 Inference & Comparation

In this paragraph, we will perform inference on our quantized model, evaluate the responses, and compare them with the responses provided by other state-of-the-art open-source models, Mistral 7B Instruct and Llama2 7b Chat. For evaluation, we will use a primitive but effective system, namely human evaluation, by selecting 20 reference questions from 4 different domains: Society, Taxation, Collection, and OutOfScope Questions.

We will use two types of scores: Fact Check Score and ITA Score. The former will assign a score to the quality and accuracy of the response, while the latter will assess the language proficiency in Italian. Both are represented by a score ranging from 0 to 2, where:

Score	Meaning
0	Completely Incorrect;
1	Partially Correct/Incorrect;
2	Completely Correct.

To the end, we will sum up the scores and classify the models.

6.5.1 Society

Question	Qual è il capitale minimo richiesto per una SRL e quale invece per una SRLS?		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Il capitale minimo richiesto per una SRL è di 10.000 euro, mentre per una SRLS è di 1 euro.	2	2
MISTRAL7b Instruct	In Italia, per fondare una Società a Responsabilità Limitata (Srl), il capitale sociale minimo ammonta a €5.000,00. Per una Società a Responsabilità Limitata con Soci Unici (Srls), invece, il capitale sociale minimo è di €1,00. Tuttavia, quest'ultima forma giuridica ha la particolarità che l'intero patrimonio societario può essere sostenuto da un solo socio, pertanto non c'è obbligo di versamento iniziale del capitale sociale dal singolo soci come avviene nella tipica Srl.	1	2
Llama2 7b Chat	Il capitale minimo richiesto per una Società a Responsabilità Limitata (SRL) è di almeno 10.000 €, mentre per una Società a Responsabilità Limitata Semplificata (SRLS) è di almeno 3.000 €.	1	2

Question	Puoi spiegarmi quali sono le caratteristiche di una società a responsabilità limitata? Quali sono i vantaggi?		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Una società a responsabilità limitata (SRL) è una società di capitali che offre la responsabilità limitata ai soci, che rispondono solo del capitale investito. I vantaggi includono la responsabilità limitata, la separazione tra patrimonio personale e societario, la flessibilità nella struttura e gestione, e la possibilità di attrarre investimenti esterni.	2	2
MISTRAL7b Instruct	Una società a responsabilità limitata (SRL) ha i seguenti caratteristiche: 1. La proprietà è separata tra il socio e la società, garantendo protezione personale contro obbligazioni insostenibili della società. 2. Esiste un capitale sociale minimo da versare al momento dell'istituzione. 3. Gli utiliti distribuiti ai soci devono essere proporzionalmente alle quote detenute. 4. Il sociato può continuare ad esistere indipendentemente dai cambiamenti dei soci. I principali vantaggi di una SRL sono: 1. Limita la responsabilità degli azionisti verso l'esterno, protettendoli dalla possibile bancarotta della società. 2. Offre facilitazione per raccogliere capitali attraverso vendita di azioni. 3. Consente la continuità d'impresa indipendentemente dal cambio dei soci.	1	1

	4. Agevola la gestione e la trasferibilità delle quotas.		
Llama2 7b Chat	Una società aresponsabilità limitata (LTD) è un tipo di empresa che offre protezione legale per gli azionisti e shareholder, ma anche limitazioni nella responsabilità finanziaria dei proprietari. I principali vantaggi includono: • Protezione legale per gli azionisti e shareholder • Limitazione della responsabilità finanziaria dei proprietari • Potenzialmente più facile raccogliere capitali da investitori • Più facilmente gestione delle attività empresarie • Meno rischio di perdite personali dei proprietari	0	1

Question	**Quali sono gli step da seguire per aprire una nuova società?**		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Gli step da seguire per aprire una nuova società includono la scelta del tipo di società da costituire, l'apertura della partita IVA, l'iscrizione al Registro delle Imprese presso la Camera di Commercio, l'apertura della posizione fiscale e contributiva, la registrazione dell'indirizzo legale dell'azienda, l'iscrizione alla gestione previdenziale INPS e la richiesta di eventuali autorizzazioni o licenze.	2	2
MISTRAL7b Instruct	1. Scegliere un tipo di business e redigere un piano d'affari. 2. Selezionare un nome ed effettuare la registrazione legale del tuo business. 3. Obtenere le permessioni e licenze necessarie a seconda della natura dell'attività. 4. Preparare i documenti legali, come contratti, statuti, e formuli di assunzione. 5. Apertere un conto bancario per il tuo business. 6. Acquistare o affittare uno spazio commerciale o ufficio. 7. Reclutare dipendenti se necessario. 8. Comprare o acquistare qualsiasi attrezzatura o materiali richiesti per l'attività. 9. Promuovere il tuo business attraverso la rete sociale, siti web, pubblicità locale, ecc. 10. Iniziare ad operare in base al tuo piano d'affari e fornire ai clienti il tuo prodotto o servizio.	2	2
Llama2 7b Chat	Per aprere una nueva società, è necessario seguire i seguenti passaggi: 1. Definire il business idea e lo scopo sociale della società; 2. Scegliere un nome per la società e registrarlo presso il Registro delle Imprese; 3. Ottieni una sede legale per la società; 4. Crea un piano di gestione e un bilancio previsionale; 5. Fai affari con le autorità finanziarie e fiscali; 6. Recluta un staff competente e forma un consiglio d'amministrazione.	1	1

6.5.2 Tax collection

Question	Ho ricevuto una cartella esattoriale. Puoi spiegarmi come dovrei comportarmi?		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Ricevere una cartella esattoriale può essere un momento stressante e confuso. Tuttavia, è importante ricordare che la cartella esattoriale è una richiesta di pagamento che può essere sottoscritta anche da un avvocato. Puoi contestare la validità della pretesa entro 60 giorni dalla notifica. Se non contesta la pretesa entro il termine previsto, potresti essere soggetto a ulteriori azioni legali da parte dell'Agente della riscossione.	2	2
MISTRAL7b Instruct	Ricevere una cartella esattoriale significa che il tuo caso è stato iscritto in un procedimento giudiziario per la riscossione di un debito. Per comportarti in modo appropriato, ti consiglio: 1. Leggi attentamente la cartella e capisca i motivi del procedimento. 2. Contatta immediatamente l'ufficio esattore o il creditor per chiedere informazioni su come pagare il debito. 3. Se puoi, negozia un piano di pagamento se si applicano le condizioni. 4. Assicurati di tenere traccia dei pagamenti fatti e delle corrispettive conferme ricevute. 5. Considera consultare un avvocato se hai dubbi sulla validità del debito o sul procedimento.	2	2
Llama2 7b Chat	Certo! Se hai received a tax assessment notice, here's what you should do: 1. Review the notice carefully: Check if there are any errors or discrepancies in the amount owed or the information provided. If you find anything wrong, contact the assessor's office immediately. 2. Determine your options: You may have the option to appeal the assessment, either partially or fully. Research the deadlines and requirements for appealing in your area. 3. Pay the taxes due (if any): If you agree with the assessment, you can pay the taxes due by the specified date to avoid penalties and interest. 4. Ignore the notice: In some cases, ignoring the notice may not be a good idea, as it could lead to additional penalties and fines. However, if you believe the assessment is incorrect or unfair, you can choose to ignore the notice and pursue an appeal later on. 5. Consult a professional: If you're unsure about how to proceed, consider consulting a real estate attorney or appraiser who specializes in property tax disputes. They can help you navigate the process and represent you in court, if necessary.	1	0

Question	Cosa posso fare se ho dei debiti con l'Agenzia delle Entrate Riscossione?		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**

ITACA	Se hai dei debiti con l'Agenzia delle Entrate Riscossione, puoi procedere al pagamento online tramite il servizio 'Paga on-line'. In alternativa, puoi effettuare il pagamento presso uno sportello o un ufficio postale, oppure tramite home banking se sei un cliente di una banca.	2	2
MISTRAL7b Instruct	Se hai debiti con l'Agenzia delle Entrate Riscossione, è consigliabile contattarli per discutere di una possibile soluzione. Puoi chiedere informazioni sulla natura e sulle modalità del tuo debito, come la cifra da pagare e i termini. Potrebbe essere possibile negoziare un piano di ripagamento o trovare altre opzioni come il ritardo della data di scadenza o la divisione in rate regolari. Non dimenticare però che non si tratta di un problema facile da risolvere da solo, quindi potresti considerare di assumere l'assistenza di un professionista qualificato come un avvocato o un esperto fiscale.	2	2
Llama2 7b Chat	Se hai dei debiti con l'Agenzia delle Entrate Riscossione, puoi seguire questi passaggi per risolvere la situazione: 1. Contatta l'agenzia: Invía un'email o una richiesta di contatto all'agenzia per discutere della tua situazione e del pagamento dei tuoi debiti. 2. Pagare il debito: Puoi pagare il debito in rate mensili o in un'unica volta, a seconda delle possibilità finanziarie dell'agenzia. 3. Chiedere aiuto: Se non sei in grado di pagare il debito intero, puoi chiedere aiuto all'agenzia per obtainedue indulgenze fiscali o per ottenere un piano di pagamento modificato. 4. Rifiutare la raccolta: Se l'agenzia decide di raccogliere i tuoi debitori, puoi rifiutare questa opzione e continuare a pagare il debito in modo indipendente.	1	1

Question	**Che cos è un avviso bonario e come posso pagarlo?**		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Un avviso bonario è una comunicazione inviata dall'Agenzia delle Entrate che segnala un errore o un'omissione nella dichiarazione dei redditi del contribuente. Per pagare un avviso bonario, il contribuente ha la possibilità di effettuare il versamento utilizzando il modello F24 telematico.	2	2
MISTRAL7b Instruct	Un avviso benevolo, anche noto come avviso amichevole o avviso di buon ufficio, è una comunicazione che si fa volontariamente per aiutare qualcuno a risolvere un problema o evitare uno. Non richiede alcun compenso in cambio. Se vuoi offrire un avviso benevolo, non c'è nulla da pagare. Invece, se hai ricevuto un servizio o consulenza pagato e desideri esprimere gratitudine, puoi considerare fare una donazione al fornitore del servizio o ad una organizzazione caritativa correlata.	0	1
Llama2 7b Chat	Un avviso bollato (also known as a "bill") is a notice issued by a government agency or local authority requiring an individual to pay a debt, fine, or other amount. Here's how you can pay an outstanding bill: Check the deadline: Make sure you understand when the payment is due and whether there are any extensions available.	0	0

	Gather your documents: Collect all relevant documents related to the bill, including the original notification, proof of payment, and any correspondence with the authorities. Pay online: Many governments offer online payment options for bills. You can usually find this information on the official website of the agency that sent the bill. Visit a payment center: If you prefer to make a personal payment, visit a designated payment center near you. These centers typically accept cash, credit cards, and debit cards. Mail a check: If you cannot pay online or at a payment center, you can mail a check along with the required documentation to the address listed on the bill. Be sure to include your reference number and any other necessary details. Contact customer service: If you have questions or concerns about your bill, reach out to the relevant agency's customer service department for assistance. They may be able to provide additional payment options or help resolve any issues.		

6.5.3 Professions and work regimes

Question	**Puoi spiegarmi in cosa consiste il regime forfettario e quali sono le differenze con il regime ordinario?**		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Il regime forfettario è un regime fiscale agevolato per le partite IVA individuali che svolgono attività di impresa, arte o professione. Le differenze con il regime ordinario riguardano l'applicazione di un'imposta sostitutiva (del 15% o del 5% per i primi 5 anni per le start-up), l'esenzione da adempimenti IVA e ritenute di acconto, l'esclusione da Irap e l'applicazione di un'imposta sostitutiva anche per i compensi. Inoltre, il regime forfettario prevede una determinazione forfettaria del reddito imponibile e semplificazioni contabili.	2	2
MISTRAL7b Instruct	Il regime forfetterio, noto anche come "regime di forfeiture" o "confisca automatica", è una legge penale che consente alle autorità di confiscare beni o denaro ritenuti il frutto di attività illecite. Questo avviene senza la necessità di prova formale o di un processo giudiziario preesistente, ma solo sulla base di sospetti ragionevoli. In contrasto, nel regime ordinario, la confisca richiede una sentenza di condanna precedente. Le principali differenze tra i due regimi risiedono nella procedura utilizzata per ottenere la confisca: nel regime forfettario non è necessaria una condanna precedente, mentre nel regime ordinario si deve prima stabilire la colpevolanza del reo attraverso un processo giudiziario.	0	1
Llama2 7b Chat	Il regime forfettario è un tipo di regime finanziario che prevede la cessione, ad un soggetto terzo, di una parte del fatturato o dei profitti dell'azienda a cambio di un determinato importo. Le differenze principali rispetto al regime ordinario sono: • Decrescita della capacità economica: l'azienda perde una parte del suo fatturato o dei propri profitti, che possono essere utilizzati da altri soggetti;	0	2

	• Incremento della flessibilità finanziaria: l'azienda può ottenere liquidità immediate senza dover attendere la vendita delle proprie merci o servizi; • Riduzione del rischio commerciale: l'azienda non assume più il rischio commerciale di vendere i propri prodotti o servizi aventi un successo imprevedibile.		

Question	**Quali sono i vantaggi del regime forfettario?**		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	I vantaggi del regime forfettario includono una tassazione agevolata, con aliquota fissa al 15% (o al 5% per i primi cinque anni per le start-up), semplificazione degli adempimenti fiscali e contabili, esenzione dagli ISA (indici sintetici di affidabilità fiscale) e dalla tenuta delle scritture contabili.	2	2
MISTRAL7b Instruct	Il regime forfetterio, noto anche come sistema di diritto assicurativo o sistema di torti obbligatorii, offre alcuni vantaggi: 1. Garanzia di copertura: Questo tipo di sistema garantisce che le vittime riceveranno risarcimenti per danni indipendentemente dal fatto che il responsabile abbia o meno un patrimonio sufficiente per pagare i danni da solo. 2. Migliore protezione per chi non può assumersi rischio: I cittadani con bassa capacità economica possono essere meglio protetti in un regime forfettario poichè esso distribuisce equamente i costi dei danni tra tutti i partecipanti al sistema, piuttosto che lasciarli a carico della persona responsabile. 3. Reduzione delle dispute legali: Poichè la maggioranza dei casi è già prevista e coperta dal contratto di assicurazione, ci sono meno controversie legali e quindi tempi più corti per raggiungere una soluzione. 4. Costi controllati: Contrariamente alla torta libera, dove i costi potrebbero crescere infinitamente in relazione ai danni subiti, nel regime forfettario i costi sono predeterminati ed hanno un limite massimo. 5. Maggiore sicurezza economica: In un regime forfettario, gli individui possono godere di una certa misura di protezione contro eventuali perdite economiche dovute ad incidenti, senza doversi preoccupare troppo dell'eventuale mancanza di fondi per affrontare queste perdite.	0	2
Llama2 7b Chat	Il regime forfettario ha diversi vantaggi, tra cui: • Riduzione delle spese per l'amministrazione dei servizi; • Miglioramento della efficienza e della qualità dei servizi offerti; • Incentivo alla competizione tra gli operatori per garantire migliori servizi ai clienti.	1	2

Question	Che cosa è il coefficiente di redditività nel regime forfettario? Puoi fornirmi un esempio?		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Il coefficiente di redditività nel regime forfettario è una percentuale che viene applicata ai ricavi o ai compensi per determinare il reddito imponibile. Ad esempio, se un professionista ha un coefficiente del 78% e dei ricavi di 10.000 euro, il reddito imponibile sarà 7.800 euro (78% di 10.000 euro).	2	2
MISTRAL7b Instruct	In a forfaitary (or lump-sum) contract, the coefficient of productivity (also known as the load factor or capacity factor) is a measure of how effectively an asset is utilized to generate revenue over a given period. It represents the ratio of actual energy production or usage to the maximum possible during that same time frame. For instance, in the context of a solar power plant, if the plant has a total capacity of 1 MW and generates 800 MWh of electricity annually, its coefficient of productivity would be 0.8 (since 800 MWh / 1000 MW = 0.8). This means that the plant was able to produce 80% of its potential annual output based on its rated capacity.	1	0
Llama2 7b Chat	Il coefficiente di redditività nel regime forfettario (o "coefficient of yield" in inglese) misura la percentuale di ritorno su investimento che si può aspettare da una determinata risorsa finanziaria, come ad esempio un prestito o un'investment. In altre parole, rappresenta la proportionamento tra il valore atteso e il valore effettivo del capitale investito. Ad esempio, se si investe 100 dollari con un tasso d'interesse del 5%, si otterrà un return del 5% (ovvero 5 dollars) sul proprio investimento. Il coefficiente di redditività in questo caso sarebbe quindi 5%.	1	1

6.5.4 Fisco Generale

Question	Che cosa è un modello 730 e come si compila?		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Il modello 730 è un documento utilizzato per presentare la dichiarazione dei redditi. Si compila indicando i redditi percepiti, le spese sostenute e le detrazioni applicabili. È possibile compilare il modello 730 autonomamente o tramite un commercialista.	2	2
MISTRAL7b Instruct	Il Modello 730 non è una cosa ben definita in termini di tecnologia o industria specifica. Potrebbe riferirsi a diverse cose a seconda del contesto, ad esempio: 1. In termini di calcolatrice scientifica, il Texas Instruments SR-52 e SR-65 sono due calcolatrici programmabili con la serie designata "Model 73" (con il 730 che	0	2

	rappresenta il secondo modello). Si compilano usando il linguaggio di programmazione RPN (Reverse Polish Notation) scritto su tastiera speciale. 2. Nel settore automobilistico, la Mercedes-Benz Classe W114/W115 ha avuto un motore diesel OM 617 denominato M119 da 730 cm^3 (4,5 litri), ma non vi è alcuna informazione sulla compilazione specifica per questo motore. 3. In chimica, il metodo Hantzsch-Widmann è noto anche come "Schema 730", ma non ci sono passaggi specifici per compilarlo. Per ottenere risposte più precise, è necessario fornire maggiori dettagli sul contesto specifico della domanda.		
Llama2 7b Chat	Un modello 730 è una macchina da calcolo ad alta velocità sviluppata dalla IBM nel 1962. È stato il primo computer a utilizare l'architettura di von Neumann, che consiste in un'unica memoria condivisa per ospitare sia lo stack che i programmi. Il modello 730 era capace di eseguire fino a 500 000 operazioni al secondo e fu utilizzato principalmente within industria e nella ricerca scientifica. La sua compilation avveniva attraverso l'uso di linguaggi di programmazione specifici come Fortran e COBOL.	0	2

Question	**Ho sentito parlare degli ISA, puoi spiegarmi cosa sono?**		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Gli ISA, ovvero gli Indici Sintetici di Affidabilità, sono strumenti utilizzati dall'Agenzia delle Entrate per valutare l'affidabilità fiscale dei contribuenti. Questi indici considerano diversi aspetti come il volume delle vendite, i costi del personale, le spese per l'acquisto di beni e servizi, e altri fattori che possono influire sulla regolarità fiscale. Gli ISA aiutano a determinare l'affidabilità fiscale complessiva di un'impresa o di un professionista.	2	2
MISTRAL7b Instruct	ISAs, o Interactive Software Applications, sono programmi informatici interattivi che consentono all'utente di esplorare e manipolare dati in modo dinamico. Sono differenti dai software tradizionali poiché offrono una maggiore interazione con l'utente ed possono essere utilizzati per un ampio spettro di applicazioni, dalla statistica alla simulazione scientifica, dall'analisi finanziaria alle mappe interattive. In genere vengono scritti usando linguaggi speciali come R, Python o MATLAB.	0	1
Llama2 7b Chat	Certamente! ISA (Industrial Safety Association) è un'organizzazione no-profit che si occupa di promuovere la sicurezza e la salute nel settore industriale. L'ISA fornisce standard di sicurezza, tecnologie e servizi per aiutare le aziende a minimizzare i rischi e ad ottimizzare la loro performance di sicurezza.	0	1

Question	**Mi sono reso conto di avere un Isee troppo alto. Puoi spiegarmi quali sono i valori che contribuiscono ad aumentarlo?**

Models/Value	Response	Fact Check Score	ITA Score
ITACA	I valori che contribuiscono ad aumentare l'Isee familiare includono: il patrimonio mobiliare e immobiliare, i redditi percepiti da tutti i componenti del nucleo familiare, le quote di possesso dei beni e i redditi dei fabbricati. Queste informazioni sono fondamentali per determinare l'Isee e influenzano l'accesso a prestazioni sociali e fiscali.	2	2
MISTRAL7b Instruct	Sono state intese due frasi nella tua domanda: "Havere un ISEE troppo alto" implica probabilmente desiderare un valore basso, quindi in questo contesti, i fattori che possono incrementare il tuo ISEE (Indice di Studio Economico Europeo) include: 1. Alta rendita familiare 2. Possesso di beni immobili o altri assetti di grande valore 3. Numero elevato di dipendenti domestici 4. Costi elevati per la scuola privata dei figli minori 5. Altri costi speciali come cure mediche o alimontazione animale. Queste situazioni possono portare a un maggiore reddito dichiarato e quindi a un punteggio ISEE più alto, rendendo più difficile accedere a certi programmi di aiuti economici o sconti fiscali.	1	1
Llama2 7b Chat	Certainly! Here are some factors that can contribute to an excessive ISEE score: 1. Age of the student: The younger the student, the more time they have had to accumulate wealth and assets through inheritance or investments. 2. Investment income: If a family has significant investment income, this could increase their overall wealth and lead to a higher ISEE score. 3. Business ownership: Families who own businesses may have access to additional funds and resources that can boost their net worth. 4. Real estate holdings: Families with significant real estate holdings, such as rental properties or vacation homes, can also contribute to a higher ISEE score. 5. Retirement accounts: Families with large retirement account balances, such as 401(k)s or IRAs, can also impact their ISEE score. 6. Other assets: Any other assets owned by the family, such as artwork, collectibles, or valuable heirlooms, can also be included in the ISEE calculation.	0	0

6.5.5 Out Of Scope

Question	**Qual è la capitale del Giappone?**		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**

ITACA	Mi dispiace, purtroppo non sono stato addestrato per rispondere a questo tipo di domande. Tuttavia, se vuoi pormi una qualsiasi domanda nel settore legale, fiscale o commerciale, sarò felicissimo di aiutarti!	2*	2*
MISTRAL7b Instruct	Tokyo è la città più popolosa e economica di giorno in Giappone, ma l'autorità politica centrale si trova a Tokyo ma ufficialmente la capitale del Giappone rimane ancora Kyoto (Kyōto) secondo la Costituzione Nipponese adottata nel 1947. Tuttavia, il governo centrale e buona parte dell'amministrazione statale sono ora basate a Tokyo.	2	2
Llama2 7b Chat	Tokyo	2	2

Question	**Raccontami una barzelletta**		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Mi dispiace, purtroppo non sono stato addestrato per rispondere a questo tipo di domande. Tuttavia, se vuoi pormi una qualsiasi domanda nel settore legale, fiscale o commerciale, sarò felicissimo di aiutarti!	2*	2
MISTRAL7b Instruct	"Perchè stai sempre in cucina? - Per cuocare le tue parole, signoranna! (Why are you always in the kitchen? - To cook your words, madam!)"	1	1
Llama2 7b Chat	Sure! Here's a quick joke: Why don't scientists trust atoms? Because they make up everything!	1	0

Question	**Can you speak English Itaca?**		
Models/Value	**Response**	**Fact Check Score**	**ITA Score**
ITACA	Mi dispiace, purtroppo non parlo l'inglese. Tuttavia, se vuoi pormi una qualsiasi domanda nel settore legale, fiscale o commerciale, sarò felicissima di aiutarti!	2*	2*[9]
MISTRAL7b Instruct	Yes, I can speak and understand English fluently. How may I help you today?	2*	2*
Llama2 7b Chat	Yes, I can speak English.	2*	2*

[*] Samples evaluated as correct samples when out-of-scope question are inquires to ITACA or other models

6.5.6 Human evaluation

	Fact Check Score	ITA Score
ITACA	24	24
Mistral 7b Instruct	17	22
Llama2 7b Chat	11	17

The results show how ITACA has significantly outperformed the competitors in tasks related to the legal and fiscal domains. Additionally, the comprehension and use of the Italian language seem to have improved considerably in ITACA compared to counterparts that do not always respond in the same language as the question is posed. Sometimes, the response in Italian is provided, but it may be unclear or subject to errors and non-existent words.

7. ITACA Deployment

However, it’s important to understand that these models are merely components within a larger software landscape. After achieving adequate performance in a controlled environment, the next step is to integrate it into your broader system.

This process of exposing the AI product to users is what we call **deployment**. In this piece, we’ll explore various aspects of this process and discuss strategies and tools that can help us successfully navigate it. But before we go any further, let’s clarify some key points of the development workflow.

7.1 Deployment of LLM Models

The development phase provides us with a controlled, predictable, and isolated environment conducive to testing and experimentation. It's a space where models can be tested without the pressure of real-world requirements or the significant consequences of errors.

On the other hand, the production environment exposes models to the real world, actual user interactions, and expectations of high availability and reliability. The transition between these two separate steps entails a variety of other disciplines like software engineering and ML engineering, and careful refining of the model to improve robustness. Furthermore, it can also include optimizing the model to improve non-functional metrics like latency or integrating it with other systems to achieve seamless functionality.

Thus, it's wrong to think of the LLM as the crux of the application or think this marks the end of the development. Elements such as data dependencies, model complexity, reproducibility, testing, monitoring, and version modifications play a more significant role in maintaining a practical LLM-powered solution. As illustrated below, the LLM code is merely a fraction in the grand scheme.

Let's say we're trying to develop a intelligent QA assistant for our company's customer service department. The goal is to allow clients to access company information without the need for direct human interaction.

We worked hard on data processing and standardization, feature design, and testing a range of configurations. We built a customized large language model that receives queries as input and provides responses based on company-specific information. So, what's next?

7.2 Required Hardware and Architecture

Hardware and Infrastructure

Deploying an LLM requires robust hardware and infrastructure. Here are the key components you'll need:

1. **Powerful GPUs/TPUs**: LLMs demand significant computational power. High-end GPUs (Graphics Processing Units) or TPUs (Tensor Processing Units) are essential for training and inference.

2. **Cloud or On-Premises**: You can choose to deploy your LLM in the cloud or on-premises infrastructure. Cloud solutions like AWS, Azure, and GCP offer scalable options, while on-premises setups provide more control.

3. **Storage**: LLMs often require large storage capacities for storing model weights, training data, and results. Fast and reliable storage systems are crucial.

Software and Frameworks

1. **Deep Learning Frameworks**: Popular deep learning frameworks like TensorFlow and PyTorch are essential for building and deploying LLMs. These frameworks provide the tools and libraries required for model development.

2. **Hugging Face Transformers**: The Hugging Face Transformers library is a valuable resource for working with LLMs. It offers pre-trained models and easy-to-use APIs for fine-tuning and deployment.

3. **Docker Containers**: Docker containers help create isolated environments for running LLMs, making deployment more manageable and consistent.

7.3 Overview of Deployment Options

Before venturing into the deployment of custom LLMs, it's crucial to grasp the available options:

1. **Hugging Face Inference Endpoints**: Renowned for its extensive model repository, Hugging Face not only furnishes pre-trained models but also furnishes a user-friendly

deployment platform. Streamlining the deployment process, it stands out as an excellent choice for experimentation and small-scale deployment.

2. **Amazon SageMaker**: As an integral component of AWS, Amazon SageMaker furnishes a comprehensive machine learning platform. Tailored for larger-scale deployments, it extends additional capabilities for data preprocessing, training, and monitoring, rendering it an ideal choice for expansive projects.

3. **Azure Machine Learning**: Microsoft's Azure ML platform presents another robust avenue for deploying custom LLMs. It furnishes a cloud-based milieu for constructing, training, and deploying machine learning models, encompassing LLMs in its repertoire.

7.4 Hugging Face for LLM Deployment

Hugging Face stands as a central hub for all things related to Natural Language Processing (NLP) and language models, playing a pivotal role in both model sourcing and deployment facilitation.

Hugging Face gives us the change to deploy two types of model:

1. **Pre-Trained Models**: Hugging Face's Model Hub boasts an extensive array of pre-trained models. These models, including FALCON-40B, LLAMA-7B, LLAMA-40B, and more, serve as foundational building blocks for custom LLM development.

2. **Custom Fine-Tuning**: Researchers and developers can capitalize on Hugging Face's pre-trained models and fine-tune them using domain-specific data. This process enables the creation of custom LLMs tailored to unique NLP tasks.

Deploying the LLM as a hugging face Inference Endpoint:

Hugging Face Inference Endpoints offers an easy and secure way to deploy Machine Learning models for use in production. Inference Endpoints empower developers and data scientists alike to create AI applications without managing infrastructure: simplifying the deployment process to a few clicks, including handling large volumes of requests with autoscaling, reducing infrastructure costs with scale-to-zero, and offering advanced security.

Here are some of the most important features for LLM deployment:

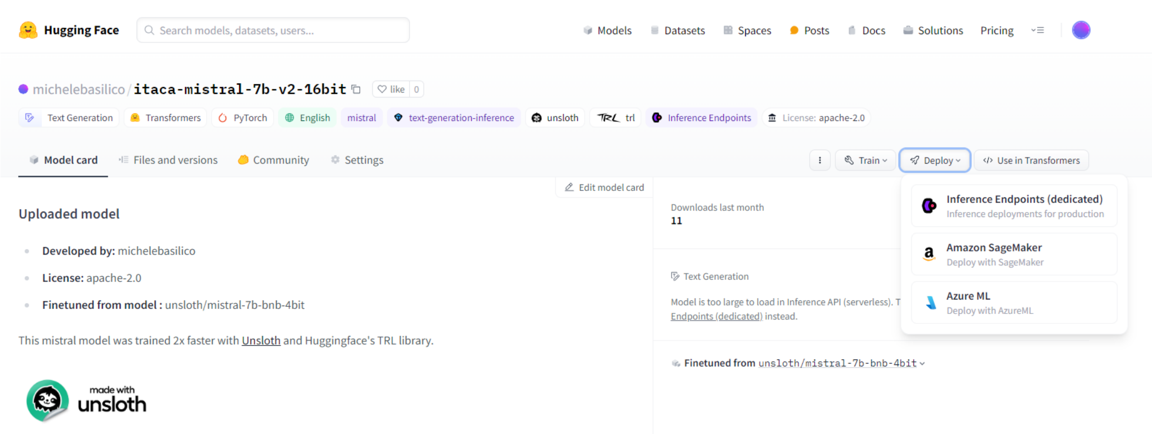
1. **Easy Deployment**: Deploy models as production-ready APIs with just a few clicks, eliminating the need to handle infrastructure or MLOps.

2. **Cost Efficiency**: Benefit from automatic scale to zero capability, reducing costs by scaling down the infrastructure when the endpoint is not in use, while paying based on the uptime of the endpoint, ensuring cost-effectiveness.

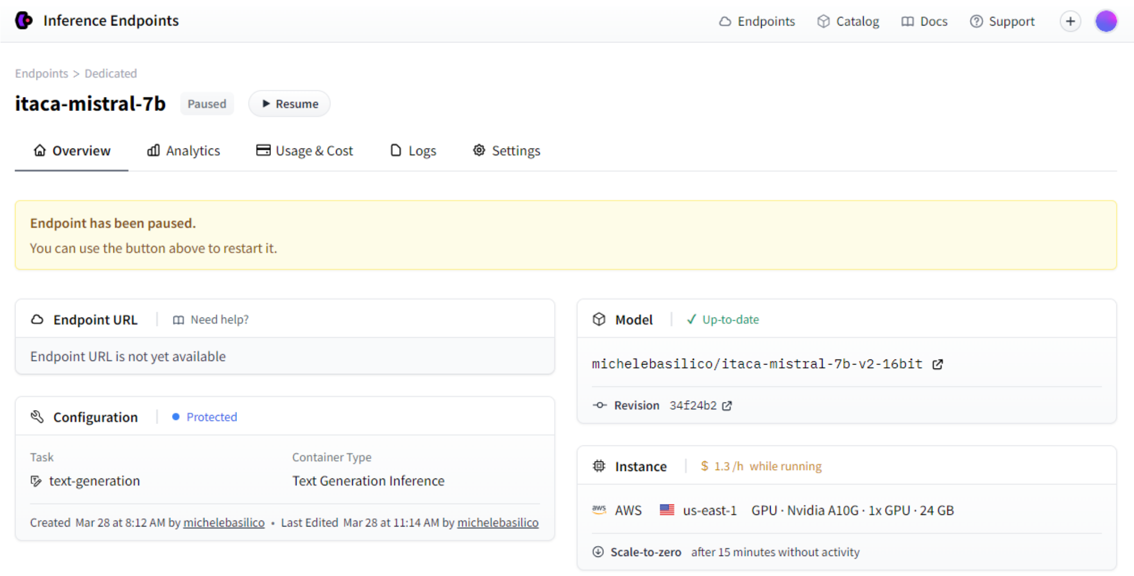
3. **Enterprise Security**: Deploy models in secure offline endpoints accessible only through direct VPC connections, backed by SOC2 Type 2 certification, and offering BAA and GDPR data processing agreements for enhanced data security and compliance.

4. **LLM Optimization:** Optimized for LLMs, enabling high throughput with Paged Attention and low latency through custom transformers code and Flash Attention power by Text Generation Inference

5. **Comprehensive Task Support:** Out of the box support for 🤗 Transformers, Sentence-Transformers, and Diffusers tasks and models, and easy customization to enable advanced tasks like speaker diarization or any Machine Learning task and library.



Inference Endpoints suggest an instance type based on the model size, which should be big enough to run the model. In our case, we are developing 16bit version of ITACA. The suggested architecture is composed by a Nvidia A100 * 1 with 24 GB RAM.



The Endpoint overview provides access to the Inference Widget, which can be used to manually send requests. This to quickly test the Endpoint with different inputs. The widget also generates a cURL command we can use. Following parameters are supported in the command:

- **temperature**: Controls randomness in the model. Lower values will make the model more deterministic and higher values will make the model more random. Default value is 1.0.
- **max_new_tokens**: The maximum number of tokens to generate. Default value is 20, max value is 512.
- **repetition_penalty**: Controls the likelihood of repetition. Default is null.
- **seed**: The seed to use for random generation. Default is null.
- **stop**: A list of tokens to stop the generation. The generation will stop when one of the tokens is generated.
- **top_k**: The number of highest probability vocabulary tokens to keep for top-k-filtering. Default value is null, which disables top-k-filtering.
- **top_p:** The cumulative probability of parameter highest probability vocabulary tokens to keep for nucleus sampling, default to null
- **do_sample**: Whether or not to use sampling; use greedy decoding otherwise. Default value is false.

- **best_of:** Generate best_of sequences and return the one if the highest token logprobs, default to null.

- **details:** Whether or not to return details about the generation. Default value is false.

- **return_full_text:** Whether or not to return the full text or only the generated part. Default value is false.

- **truncate:** Whether or not to truncate the input to the maximum length of the model. Default value is true.

- **typical_p:** The typical probability of a token. Default value is null.

- **watermark**: The watermark to use for the generation. Default value is false.

7.5 Gradio UI Experience

Gradio stands out as a powerful Python toolkit designed to expedite the development of interactive demos and web applications tailored for machine learning models, APIs, or any custom Python functions. Its robust features enable users to effortlessly share their creations through built-in sharing capabilities.

At the heart of Gradio lies the **gr.Interface** class, serving as the cornerstone for crafting engaging demonstrations. This class is meticulously engineered to encapsulate machine learning models within intuitive user interfaces. Upon instantiation, users must define three key parameters:

1. **fn**: This parameter denotes the function that the user interface will revolve around. It essentially serves as the core functionality behind the interface.

2. **inputs**: Gradio offers a diverse range of components for input, each corresponding to the input requirements of the function. The number of input components must align with the function's arguments.

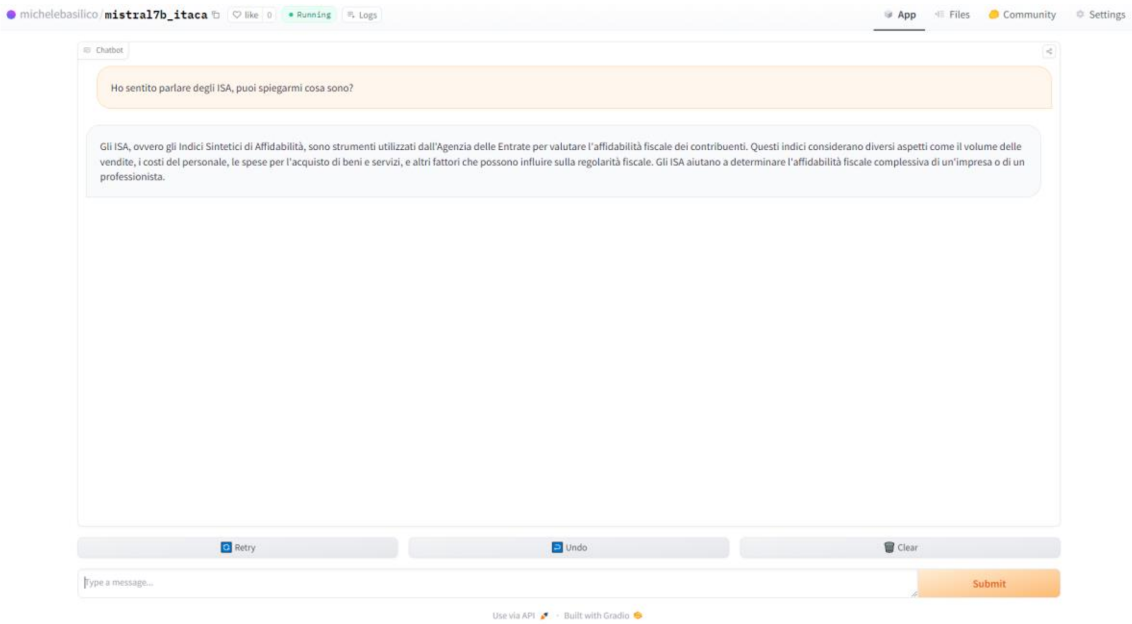
3. **outputs**: Similarly, Gradio provides an array of components for output, tailored to display the results generated by the function. The count of output components should match the number of return values from the function.

Furthermore, Gradio encompasses an additional high-level class known as **gr.ChatInterface**. Specifically crafted for the creation of Chatbot user interfaces, this class simplifies the process of building interactive Chatbot interfaces. Users simply supply a function, and Gradio seamlessly constructs a fully functional Chatbot UI, ready for deployment.

Let's look the code to deploy a chatbot like style on HuggingFace with Gradio:

```
import gradio as gr
from transformers import pipeline
import requests
import os

start_token = "<s>"
start_instruction_token = "[INST] "
end_instruction_token = " [/INST]"
system_prompt = "Sei un assistente utile ed affidabile. Rispondi in maniera adeguata
alla domanda seguente:\n"
start_completion = "\nRisposta: "

API_URL = "https://cyk11dj2ce5ybyjq.us-east-1.aws.endpoints.huggingface.cloud"
token = "Bearer " + os.getenv("ITACA_TOKEN")

headers = {
    "Accept": "application/json",
    "Authorization": token,
    "Content-Type": "application/json"
}


def query(payload):
    response = requests.post(API_URL, headers=headers, json=payload)
    return response.json()


def predict(message, history):
    new_message = start_token + start_instruction_token + system_prompt + message +
end_instruction_token + start_completion
    print(new_message)
    output = query({
        "inputs": message,
        "parameters": {
            "max_new_tokens": 1024,
            "return_full_text": False
        }
    })
    return output[0]["generated_text"]


iface = gr.ChatInterface(predict)
iface.launch()
```

With just this few lines of code, we can achieve a great result like this one:



8. Result

In the last part of this thesis work, we can dedicate a space to summarize the result achieved, explain possible limitations, and analyse possible improvements to the model and for the process.

8.1 A look back to the project

It has been a long journey. The ITACA project was born with a very challenging goal: to compete with very limited resources in a complex and dynamic field, such as artificial intelligence, by creating a modern chatbot in the Italian language for the legal, fiscal, and commercial sectors.

We aimed to achieve excellent results without resorting to demanding and costly resources, and without the fear of entrusting private data to third-party and untrusted companies.

We started our journey by introducing the world of Large Language Models in detail, showing the necessary path for its development and highlighting all the key steps that constitute it.

We delved into the long and complex process of creating not one, but two completely new datasets in the Italian language for the legal and fiscal fields, without being able to rely on experts in the field. We utilized some existing tools and developed others specifically for our purpose, such as LLMDSGenerator.

Subsequently, we introduced all the most modern Large Language Models at the state of the art and questioned which one could be most suitable to support our project.

We explored in detail the process of fine-tuning the model, describing the most modern techniques available, which allowed us to save time and resources throughout the process, from the LoRA strategy, which enabled us to train only part of the model, to the quantization of the model itself, which allowed us to work with low computing power and reduced times.

We then analyzed the results obtained, comparing them to those of the available modern open-source large language models, and demonstrating the excellent results achieved despite the limited resources available.

Finally, we leveraged modern tools on the market to deploy the model and make it freely usable, just like a chatbot, both through the use of APIs and by providing an app based on Gradio and available on a HuggingFace space.

The development of ITACA has not been easy; none of the steps examined have been. Things didn't always go in the right direction, but the final result has more than demonstrated how it is possible today to create a highly prepared model for a specific task, even starting from open-source solutions, with commercial licenses, and without the computational resources of a big company.

8.2 Limits and challenges

Despite the success of the project, it is important to learn and understand its defects and limitations in order to address them in the future. Highlighting the challenges faced can provide insights into potential solutions.

Despite the continuous evolution in the field of Large Language Models (LLMs) and the availability of more open-source models for commercial use, they may not always be optimally suited for all tasks. Language compatibility is a significant issue, particularly for languages other than English.

Currently, most available models have been pretrained on English language data. When applied to foreign languages, many of these models demonstrate difficulties or even complete inability to perform effectively. Even models trained on a minority of data in other languages often struggle to provide accurate responses, mixing words from different languages or inventing entirely new ones, leading to responses in languages different from the intended one.

This language challenge is evident in our work as well. If the base model was trained with minimal or no data in Italian, the fine-tuning process may not have yielded the desired effects because there was no prior knowledge to leverage.

This challenge necessitated the creation of a large dataset. While fine-tuning models often require fewer examples, such as around 1000 or slightly more, generating a dataset of this size would not only have allowed for greater attention to detail but also would have enabled manual creation. However, in our case, so few samples would not have been sufficient to adequately integrate the Italian language into the model, leading us to develop a synthetic dataset generator.

Another limitation faced was the lack of adequate computational resources for such a problem. While we demonstrated that a good model can be obtained with minimal or even no investments and resources, this may have limited or affected the final result to some extent. Quantization and fine-tuning applied exclusively to the model's adapters showcased remarkable efficiency despite the limited resources required. However, we cannot definitively assess the degradation of the obtained model compared to one fine-tuned using a 16-bit version or even a full fine-tuning process.

8.3 Possible Improvements

As discussed in the previous paragraph, despite the achieved results, the current evolution of the Artificial Intelligence field suggests that these accomplishments could soon be surpassed.

Firstly, one potential advancement could involve using a model trained entirely in Italian as the base model for the fine-tuning process. While currently, models trained on datasets entirely in English dominate the market, especially in the open-source domain, it's easy to foresee the need for models capable of effectively communicating in all languages. Training a model that already has a strong understanding of the Italian language could exponentially enhance ITACA's capabilities.

Secondly, further progress could be achieved by conducting a more meticulous search and tuning of parameters during the model fine-tuning process, especially with access to more powerful and efficient resources. This would enable more thorough exploration of different solutions. Moreover, leveraging more substantial resources could enhance model performance, allowing for the use of more powerful base models with more parameters to train, such as LLama70B, Mistral8x7B, and beyond. Additionally, increased resources could be invested in creating an even more excellent quality dataset, possibly by involving domain experts.

It's important to note that the LLMDSGenerator project was developed alongside ITACA and is still in its early stages. It may have some defects or issues to address before being definitively utilized. The LLMDSGenerator project will continue to be supported and developed, and one potential future step for ITACA could involve retraining it on a more reliable and robust dataset based on the new version of the generator.

Lastly, considering the utilization of new, more efficient training techniques could further improve the model. While this project was underway, several new techniques and technologies have emerged. Methods such as RAG or DPO could be employed for training a model with even greater performance.

Bibliography

Abubakar Abid, A. A. (2019, Jun 6). *Gradio: Hassle-Free Sharing and Testing of ML Models in the Wild.* Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.1906.02569

Ashish Vaswani, N. S. (2017). Attention Is All You Need. 15.

Biewald, L. (2020). *Experiment Tracking with Weights and Biases*. Retrieved from wandb: https://www.wandb.com/

Bonnet, A. (2023, 08 22). *Fine-tuning Models: Hyperparameter Optimization*. Retrieved from https://encord.com: https://encord.com/blog/fine-tuning-models-hyperparameter-optimization/

Capelle, T. (2024, Jan 15). *How to Fine-Tune an LLM Part 1: Preparing a Dataset for Instruction Tuning*. Retrieved from https://wandb.ai/: https://wandb.ai/capecape/alpaca_ft/reports/How-to-Fine-Tune-an-LLM-Part-1-Preparing-a-Dataset-for-Instruction-Tuning--Vmlldzo1NTcxNzE2

Capelle, T. (2024, Feb 20). *How to Fine-Tune an LLM Part 2: Instruction Tuning Llama 2*. Retrieved from https://wandb.ai/: https://wandb.ai/capecape/alpaca_ft/reports/How-to-Fine-Tune-an-LLM-Part-2-Instruction-Tuning-Llama-2--Vmlldzo1NjY0MjE1

Capelle, T. (2024, Feb 1). *How to Fine-tune an LLM Part 3: The HuggingFace Trainer*. Retrieved from https://wandb.ai/: https://wandb.ai/capecape/alpaca_ft/reports/How-to-Fine-tune-an-LLM-Part-3-The-HuggingFace-Trainer--Vmlldzo1OTEyNjMy

Dan Hendrycks, C. B. (2020, Sep 7). *Measuring Massive Multitask Language Understanding.* Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.2009.03300

Daniel Han, M. H. (2023). *unsloth*. Retrieved from https://unsloth.ai/: https://unsloth.ai/#unsloth%20PC%20logo

Das, S. (2024, Jan 25). *Fine Tune Large Language Model (LLM) on a Custom Dataset with QLoRA*. Retrieved from https://dassum.medium.com/: https://dassum.medium.com/fine-tune-large-language-model-llm-on-a-custom-dataset-with-qlora-fb60abdeba07

Das, S. (2024, 01 25). *https://dassum.medium.com/fine-tune-large-language-model-llm-on-a-custom-dataset-with-qlora-fb60abdeba07*. Retrieved from medium: https://dassum.medium.com/fine-tune-large-language-model-llm-on-a-custom-dataset-with-qlora-fb60abdeba07

Hugging. (2023). *dward Beeching and Clémentine Fourrier and Nathan Habib , Sheon Han, Nathan Lambert, Nazneen Rajani , Omar Sanseviero , Lewis Tunstall , Thomas Wolf*. Retrieved from Open LLM Leaderboard: https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

Hugo Touvron, L. M. (2023, Jul 18). *Llama 2: Open Foundation and Fine-Tuned Chat Models.* Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.2307.09288

Hugo Touvron, T. L.-A. (2023, Feb 27). *LLaMA: Open and Efficient Foundation Language Models.* Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.2302.13971

Keisuke Sakaguchi, R. L. (2019, Jul 24). *WinoGrande: An Adversarial Winograd Schema Challenge at Scale.* Retrieved from https://arxiv.org/: https://arxiv.org/abs/1907.10641

Leandro von Werra, Y. B. (2020). *TRL: Transformer Reinforcement Learning.* Retrieved from GitHub: https://github.com/huggingface/trl

Lewis Tunstall, E. B. (2023). *The Alignment Handbook.* Retrieved from GitHub: https://github.com/huggingface/alignment-handbook

Lintang Sutawika, H. S. (2024). *EleutherAI/lm-evaluation-harness: v0.4.2.* Retrieved from https://doi.org/10.5281/zenodo.10829972

LoRA: Low-Rank Adaptation of Large Language Models. (2021, Jun 17). Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.2106.09685

Maheshkar, S. (2024, Jan 9). *A Brief Introduction to LoRA.* Retrieved from https://wandb.ai/: https://wandb.ai/sauravmaheshkar/LoRA/reports/A-Brief-Introduction-to-LoRA--Vmlldzo2MDAyOTU3

Mistral 7B. (2023, Oct 10). Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.2310.06825

Neil Houlsby, A. G. (2019, Feb 2). *Parameter-Efficient Transfer Learning for NLP.* Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.1902.00751

Peter Clark, I. C. (2018, Mar 14). *Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge.* Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.1803.05457

Problems, T. V. (2021, Oct 27). *Training Verifiers to Solve Math Word Problems.* Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.2110.14168

QIGen: Generating Efficient Kernels for Quantized Inference on Large Language Models. (2023, Jul 7). Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.2307.03738

QLoRA: Efficient Finetuning of Quantized LLMs. (2023, May 23). Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.2305.14314

Ramanathan, B. (2022, Dec 21). *Processing Data for Large Language Models.* Retrieved from https://wandb.ai: https://wandb.ai/wandb_gen/llm-data-processing/reports/Processing-Data-for-Large-Language-Models--VmlldzozMDg4MTM2

Rohan Taori, I. G. (2023). *Stanford Alpaca: An Instruction-following LLaMA model.* Retrieved from GitHub: https://github.com/tatsu-lab/stanford_alpaca

Rowan Zellers, A. H. (2019, May 19). *HellaSwag: Can a Machine Really Finish Your Sentence?* Retrieved from https://arxiv.org: https://doi.org/10.48550/arXiv.1905.07830

Schmid, P. (2023, July 4). *Deploy LLMs with Hugging Face Inference Endpoints*. Retrieved from https://huggingface.co: https://huggingface.co/blog/inference-endpoints-llm

Sooriyarachchi, A. (2023, Aug 30). *Efficient Fine-Tuning with LoRA: A Guide to Optimal Parameter Selection for Large Language Models.* Retrieved from https://www.databricks.com/: https://www.databricks.com/blog/efficient-fine-tuning-lora-guide-llms

Stephanie Lin, J. H. (2021, Sep 8). *TruthfulQA: Measuring How Models Mimic Human Falsehoods.* Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.2109.07958

Thakur, A. (2023, 12 22). *How to Evaluate, Compare, and Optimize LLM Systems*. Retrieved from wandb.ai: https://wandb.ai/ayush-thakur/llm-eval-sweep/reports/How-to-Evaluate-Compare-and-Optimize-LLM-Systems--Vmlldzo0NzgyMTQz

Thakur, A. (2023, Dec 22). *How to Evaluate, Compare, and Optimize LLM Systems*. Retrieved from https://wandb.ai/: https://wandb.ai/ayush-thakur/llm-eval-sweep/reports/How-to-Evaluate-Compare-and-Optimize-LLM-Systems--Vmlldzo0NzgyMTQz

Thomas Wolf, L. D. (2019). HuggingFace's Transformers: State-of-the-art Natural Language Processing. *https://arxiv.org/abs/1910.03771v5*, 8.

Tu, V. (2023, Oct 12). *Fine-tuning Mistral 7B with W&B.* Retrieved from https://wandb.ai/: https://wandb.ai/vincenttu/finetuning_mistral7b/reports/Fine-tuning-Mistral-7B-with-W-B--Vmlldzo1NTc3MjMy

Yizhong Wang, Y. K. (2022, Dec 20). *Self-Instruct: Aligning Language Models with Self-Generated Instructions.* Retrieved from https://arxiv.org/: https://arxiv.org/abs/2212.10560

Zhuoyan Li, H. Z. (2023, Oct 11). *Synthetic Data Generation with Large Language Models for Text Classification: Potential and Limitations.* Retrieved from https://arxiv.org/: https://doi.org/10.48550/arXiv.2310.07849