Internship Report

Implementing Large Language Models for Personalized and Data-Driven Messaging

Name : Neralb Cika
University : Politecnico di Torino
Internship Organization : Volcanic Minds S.r.l
Academic Supervisor : Paolo Garza
Company Supervisor : Davide Dispenza
Internship duration : 300 hours
Date of submission : 26/06/2024
Github : https://github.com/volcanicminds/stage-ai-sales-agent.git

Introduction

Background Information

Volcanic Minds is a company specialized in tailor-made solutions and quality driven services. The areas of focus are :

Web development, App mobile, UX/UI design , CX experience, Cloud Computing, devOps architecture, API integration, CTO strategy.

They offer this kind of services:

- **Design Strategy :** Through comprehensive analysis of the product to be developed they ensure long-lasting results while maintaining perfect synergy with the clients.
- **Digital Factory :** They use the most robust and advanced technologies for *Frontend, Backend* and *Cloud*, satisfying all the specific requirements. Here all the ideas take shape and turn into concrete projects.
- **Project Leading :** They offer project leading services as the optimal solution for managing the project quickly and efficiently in every phase.

Purpose and Objective of the Internship

Looking at the new technologies that are being researched and developed in the world, we can say that Artificial Intelligence is a key driver of innovation across industries today. The company was keen on exploring this technology and all of its possible uses in the market.

During my internship, I focused on leveraging AI to generate personalized cold outreach messages on LinkedIn, targeting professionals and companies in the IT field and others to create potential partnerships. This involved gathering and analyzing data from various enrichment sites, primarily sourcing information from LinkedIn to tailor our messaging strategy.

By understanding both company and employee profiles, we aimed to create meaningful connections that could lead to strategic collaborations and business growth. Throughout the process, we refined our outreach approach to enhance efficiency and scalability, ensuring each interaction remained genuine and relevant to the recipient's needs

Methodologies

<u>Large Language Models</u>

1. Llama3

The Llama3 is an auto-regressive language model that uses an optimized transformer architecture, noted by its substantial 70 billion parameters and an impressive context window of 8192 tokens. This configuration allows the model to handle and generate text with depth and continuity, making it ideal for tasks requiring detailed understanding and contextual coherence. Leveraging cutting-edge transformer architecture,Llama3 excels in applications such as extensive document analysis, conversational agents and complex content creation, where maintaining context over long interactions is essential. We have taken in consideration also Llama3 with 8 billion parameter model.
Here is a summary of the technical details of Llama3:

- Uses standard decoder-only transformer
- The vocabulary is 128K tokens
- It is trained on sequences of 8K tokens.
- It applies grouped query attention (GQA)
- It is pretrained on over 15T tokens.
- It involves post-training that includes a combination of SFT, rejection sampling, PPO and DPO

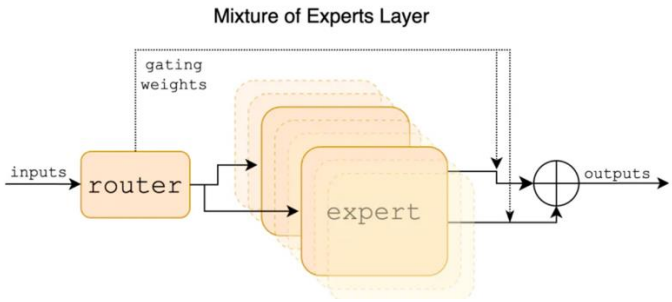
2. Gemma-7b-it

Gemma is a model released by Google DeepMind, a series of open language models inspired by the same research and technology used to create Gemini. The Gemma model includes 2B (trained on 2T tokens) and 7B (trained on 6T tokens) models including base and instruction-tuned checkpoints, trained on a context length of 8192 tokens.
Model Architecture : Gemma model architecture is based on transformer decoder with improvements including multi-query attention (used by the 2B model), multi-head attention (used by 7B model), RoPE embeddings, GeGLU activations, and normalizer location.
According to the technical report, Gemma 2B and 7B are trained on 2T and 6T tokens mainly consisting of web documents, mathematics, and code. Unlike Gemini, these models are not explicitly trained to support multilingual or multimodal capabilities. Vocabulary size is 256K tokens and uses a subset of the SentencePiece tokenize of Gemini, preserves whitespace in splits digits, and relies on byte-level encodings for unknown tokens.
This is the version in Italian language.

3. Mixtral-8x7b-3232768

Mixtral model is a decoder only Transformer model designed to leverage the strengths of multiple language models or algorithms. This can involve combining various models to enhance performance, accuracy and versatility across different tasks. Combining the strengths of different architectures, such as transformer-based models like GPT and BERT, the Mixtral model might excel in domain specialization, tailored for industries such as finance, healthcare, and customer service. By leveraging ensemble techniques, it improves predictions through integrated outputs from multiple models and efficiently handles diverse data sources. Some of its characteristics are:

- Mixture of experts model with 8 experts for MLP, with a total of 45 billion parameter model.
- Sliding Window Attention – Trained with 8k context length and fixed cache size, with a theoretical attention span of 128k tokens.
- GQA (Grouped Query Attention) – allowing faster inference and lower cache size.
- Byte-fallback BPE tokenizer – ensures that characters are never mapped to out-of-vocabulary tokens.



4. GPT 3.5 Turbo

ChatGPT is a model trained by OpenAI that has the capability to interact in a conversational way. This model is trained to follow instructions in a prompt to provide appropriate responses in the context of a dialogue.ChatGPT can help with answering questions, suggesting recipes, writing lyrics in a certain style, generating code, and much more.
ChatGPT is trained using Reinforcement Learning from Human Feedback (RLHF). Gpt-3.5-turbo is a ChatGPT API that is very powerful and very cheap, and is built specifically for conversational agents task. In fact OpenAI recommends this as their best model even for non chat use-cases.

Data Collection

We have used the Bardeen platform, a Google extension that automates workflow tasks. Our goal was to gather data about companies and employees that align with Volcanic Minds' interests and projects.We created data pipelines that collect information from various web sources and software, transform it into the right format, build appropriate databases, and store everything effectively. One of Bardeen's strengths is its integration with many other useful tools like Apollo and HubSpot.

Here's how we can describe the data pipelines:

Company Data Enrichment

1. Web scrapping with 'Ufficio Camerale' template.

In Italy, there is a website called Ufficio Camerale that has information about all registered companies. The data available includes **the number of employees, total earnings, foundation year, company email, and address**. We needed to input a list of company website URLs, and then we set up an automatic process to collect all the public information. We used Airtable, along with Bardeen, to store all of our data in an organized way. This system makes it easy to manage and access important company details efficiently..

2. API calls to Apollo.io

Apollo.io is a powerful tool designed to help businesses improve their sales efforts. The website gives users access to a huge database of over 220 million contacts and 30 million companies, making it easy to find and target potential customers.

In our case, we used Apollo API calls for enriching our database of companies. We were allowed only a limited amount of calls to make.

The data that we could secure about the companies are these:

- **Description :** This provides an overview of the company, highlighting its value propositions and key focus areas. It can also showcase past and current projects. With the description of the company, using also some classification tools from Bardeen, we were able to classify if the company was IT , Consultancy and other. We created a new column in the database with the name Type of Company. In our planning, we could tailor different messages based on the type of the company and the services that Volcanic Minds offered.
- **Keywords :** By incorporating these specific keywords into our communication, the language models was able to find common points and could generate messages that resonate more with the company's unique focus and needs.

All of the steps mentioned above are implemented using Bardeen playbooks.

Employee Data Enrichment

The tailored Linkedin messages would be directed in professionals working in different companies. This rich dataset is crucial for identifying commonalities and effectively targeting our messages. To achieve this, we have implemented an automated employee data enrichment workflow using Python and API pull requests. Specifically, we utilize Apollo.io's data, provided in JSON format. Through Python programming, we parse this data and transform it into various useful formats. Our API Pull request was constructed so that we requested data of people working in companies located in Italy and having titles:

- Sales manager, Founder, Co-founder, Co-fondatore, General Manager, CTO, CEO, Partnership manager.

The JSON response contains many fields, but we have extracted only the fields of interest, and they are:

- First Name, Last Name, Name, ID, Employment History, Title, Email, Linkedin URL.

With these fields we have the full information for the messages.

Employment History

This field is a dictionary by itself, and contains all the previous roles and positions the professional has had in the past, also indicating the year. From this field, we wanted to understand how many years of work experience does the professional have and the counting of times of each role using Python programming and data manipulation. In addition we wanted to analyze well the previous roles and understand if it would align well with our offerings and services.

After data preprocessing for both companies and employees, we have created two tables, one for each and we store them in Airtable. We will then serve to the large language models two dictionaries:

- **Employee_dict** which contains data about the employee
- **Company_dict** which contains data about the company.

Using the Employee_dict and Company_dict, the large language model generates personalized LinkedIn messages. By leveraging the detailed information about the professional’s work history, role, and company specifics, the model can generate messages that highlight common points, relevant experiences, and potential areas of interest This approach ensures that even when reaching out to a large number of professionals, each message remains personalized and relevant.

Prompt Engineering

This is maybe the most important phase of the work, since effective prompting should be made in-order to convey the right message. We have experimented a lot in this phase, made a lot of trial and errors with the prompts and observe how the large language models responds in various scenarios. We have used the same prompts for all the language models, and they have shown different results.
One famous technique that we have used is called **Few-Shot Prompting,** where basically you provide several different examples of responses for the LLM in-order to make a similar generation. We have provided around 4 examples of messages and they proved to be very useful.
Aspects that were very important in prompting were:

- Purpose and intent of the message.
- Which information to display in the output and which not.
- Which parts of the customers information to be focused on.
- Type of language to use, and ensuring that it did not use strange words or phrases.
- The tone of the language, professional and polite.

The prompts have been modified several times, and after deciding on which prompts to be the most effective, we have generated the messages.
In my part, I have generated more than 200 messages in total, but the number of messages that have been evaluated were around 120. Again, using Python we were able to systematically put the messages in tables also with information related to it in other columns like **Employee_dict** and **Company_dict**, so that in the evaluation part, the supervisors could observe how the quality of the message was related to the data provided to the LLM.

Evaluation

After generating the messages, now its evaluation part.
We decided to use 7 metrics as evaluators for our messages, based on a paper published about **Personalized Persuasion with LLM**, and those are:
Accuracy , Hallucination, Efficacy, Fluency, Coherence, Transparency, Safety, Human Alignment.

Here is a short description about the evaluators:
Accuracy: If the information provided in the message is correct and free from syntactic and semantic errors.
Efficacy: The effectiveness of the message in achieving its purpose.
Fluency: The smoothness and coherence of the language used in the message.
Transparency: The degree to which the message is clear, honest, and transparent about the intentions and expectations.
Safety: Ensuring that it does not contain any misleading content that could harm the reputation or interests of either company.
Human Alignment: How well the tone, style, and overall approach of the message align with the human touch.
Hallucination : Deviation from the input prompt or context, resulting in inconsistent results.

We highlight the fact that the evaluations ranged on a scale 1-3, and all of them were hand-made. The company supervisors attentively read all the message and evaluated them based on the metrics. Expert evaluations might be one of the highest form of evaluations since it's the most related to humans and the most accurate ones.

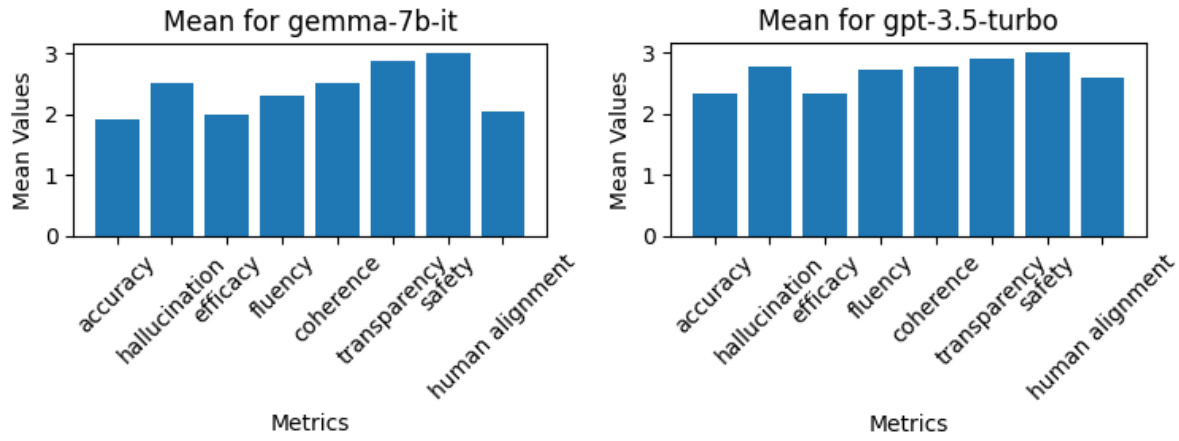
Another crucial aspect of our evaluation is that our company supervisors not only provided numerical ratings for the messages but also offered written feedback on what was good, what was bad, and how the messages could be improved. This feedback was included in a column within our data table, allowing for easy manipulation and analysis using Python.

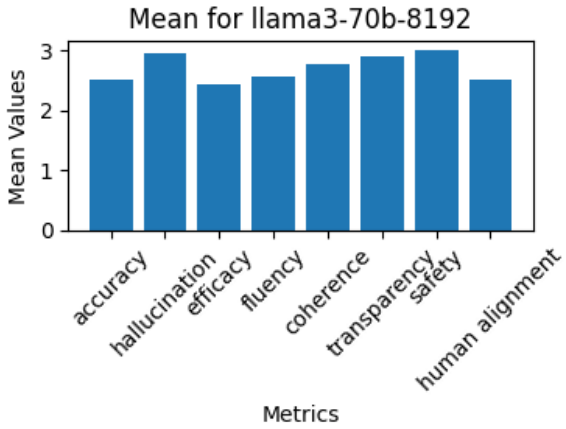
I implemented various programming techniques and utilized LLMs to analyze this feedback. These tools were very effective for general and broader topics. However, when it came to specific, nuanced remarks that require professional expertise, the LLMs were unable to fully capture the details. Nonetheless, LLMs can be highly useful for handling large volumes of feedback that would be impractical for a human to analyze manually

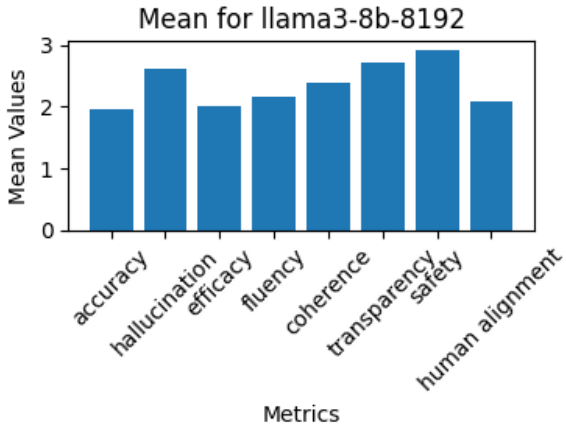
Conclusions

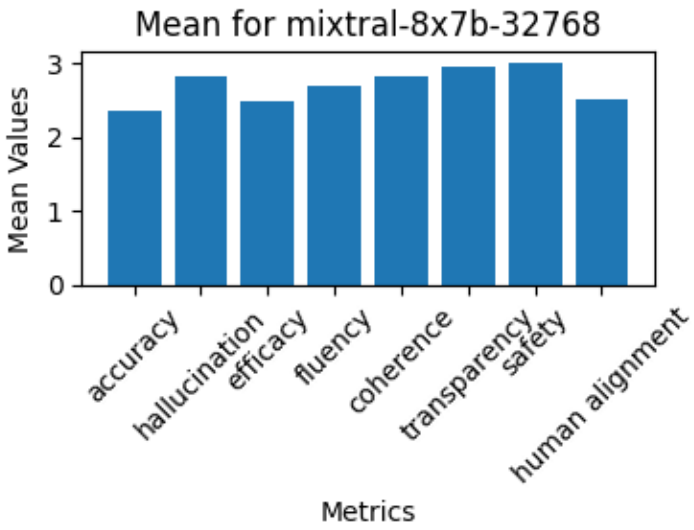
Based on the provided evaluations, we conducted statistical analyses to help the company understand the strengths and weaknesses of each LLM. We performed a straightforward analysis using the mean values and standard deviations for each model. Initially, we visualized the mean and standard deviation for each metric across all messages, offering a clear overview of the performance and consistency of each LLM.

Mean Values







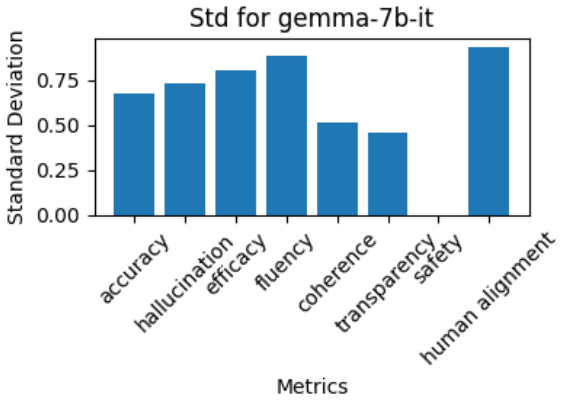


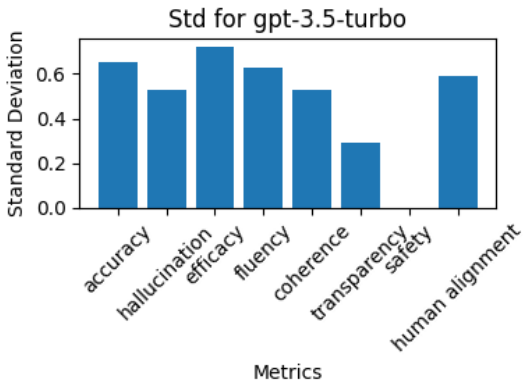
As the graphs show, Mixtral and Llama 70b have achieved the highest scores across several evaluators, with GPT-3.5 Turbo also performing well. It's important to note that not all evaluators are equally significant. After careful discussion, we concluded that Accuracy, Fluency, and Human Alignment are the most relevant metrics for this particular task. Therefore, we have prioritized these metrics when judging the models.

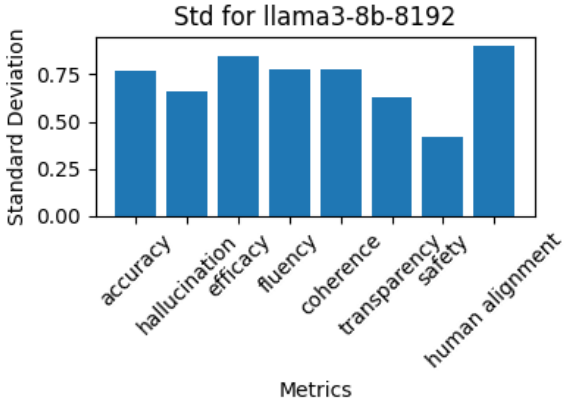
Standard Deviation

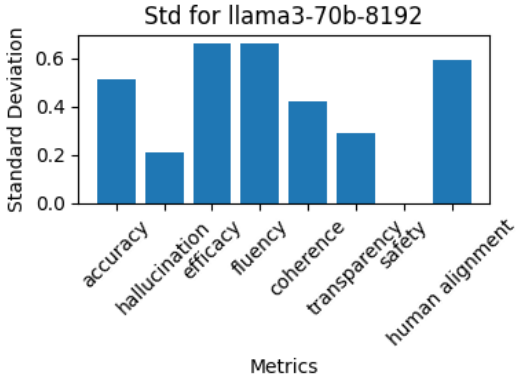
The purpose of this process was to understand the variability and consistency in the performance of each model across different metrics. By calculating the standard deviation, we aimed to identify which metrics showed more variability and which ones were more stable, providing insights into the strengths and weaknesses of each LLM in a quantifiable manner.

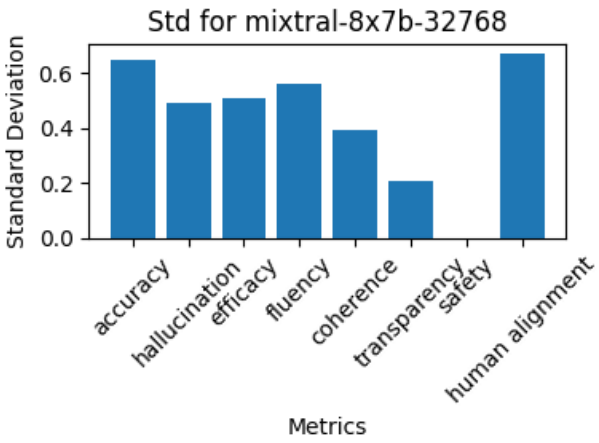
The results show that Llama3-70b has performed very well and might be the best model overall. However, it is also clear that each model behaves differently. This means we can choose the model that best fits the specific task we need to do.









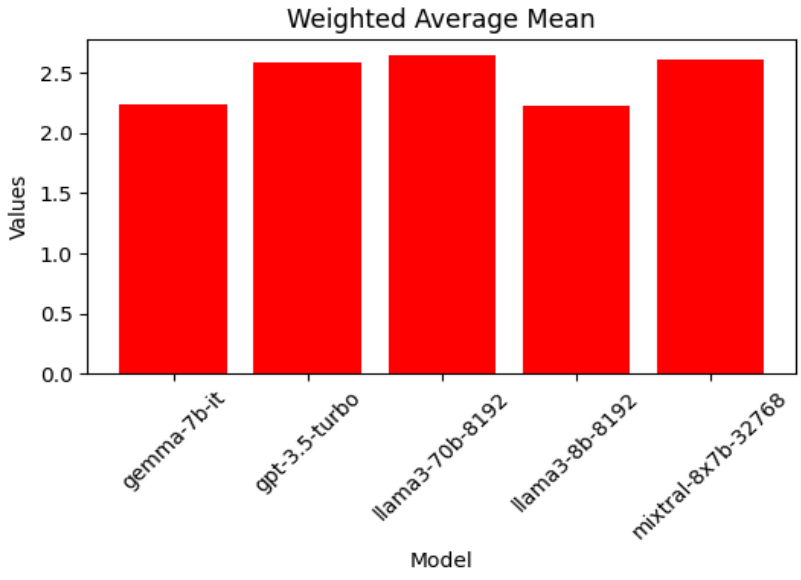


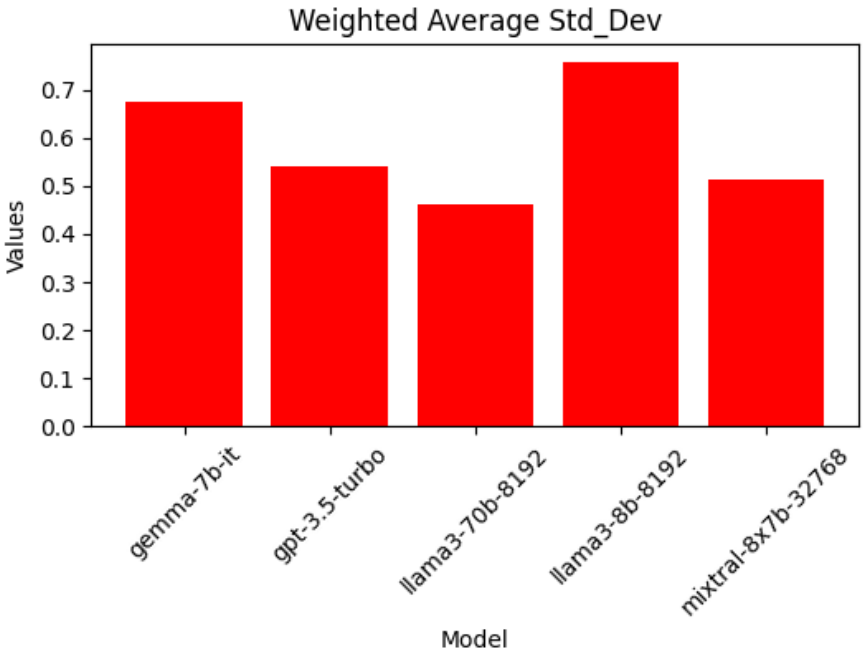
Furthermore, we believe that it would be reasonable to derive one value of performance for each model across the messages, so to have a more compact final value.
We have chosen a set of weights to distribute to the metrics, in order to compute a **Weighted Mean** and **Weighted Standard Deviation**. Basically, we make a weighted average on the values visualized before, setting a set of weights for the metrics that sum up to 1 that differ between them. We multiply the Mean and Standard Deviation by those values for each model and in the end we sum.

Our choices of values are these, but they can be altered based on the task:

- **Accuracy = 0.3**
- **Human Alignment = 0.2**
- **Fluency = 0.1**
- **Hallucination = 0.08**
- **Efficacy = 0.08**
- **Coherence = 0.08**
- **Safety = 0.08**

These are the results we have obtained:





We plan to continue our experiments and studies to gain a more comprehensive understanding of the LLMs. By conducting further testing and analysis, we aim to gather more detailed insights into the performance and behavior of each model. This will help us make more informed decisions about which model is best suited for various tasks.
Thank you for your attention.