

PROJECT REPORT

AGENT ARCHITECTURE

The agent comprises three components: **CustomAgent**, **CustomHuggingFaceLLM**, and **AgentExecutor**.

1. CustomAgent:

- Inherits from LangChain's BaseSingleActionAgent and serves as the decision-maker.
- It receives user queries, processes them using the language model, and categorizes responses into "Information Request" or "Actionable Task."
- Based on the category, it either executes a task or returns an answer directly.
- It leverages tools like get_answer and perform_action to manage tasks.

2. CustomHuggingFaceLLM:

- Integrates a Hugging Face language model for text generation.
- It structures outputs with fields like "Response," "Category," and "Action Taken," ensuring clarity.
- Model parameters (e.g., temperature, top_p) are adjusted for consistent responses.
- It acts as the core text generator, enabling decision-making within the agent.

3. AgentExecutor:

- Combines the agent and tools, handling input processing and tool selection.
- It manages the interaction loop and determines the appropriate tool for each action.
- With a set iteration limit, it ensures effective user-agent communication.

REASON FOR CHOOSING LLAMA3.2

- I chose **LLaMA 3** over **GPT-2**, **GPT-Neo**, and **GPT-J** because it supports better **function calling** ability, enabling more dynamic task handling and model interaction. It also offers better reasoning, longer context management, making it more effective for real-world applications

IMPLEMENTATION DETAILS

- The project was implemented using **Colab Pro**, which offers additional compute units, making it suitable for heavy computations. The implementation utilized an **NVIDIA L4 GPU**, which is optimized for AI workloads.
- The model used was **meta-llama/Llama-3.2-1B-Instruct**, with a model size of approximately **1 billion parameters**, requiring about **4-6 GB** of GPU memory during inference, depending on the batch size and precision (e.g., float16).

ALTERNATE METHODS:

These agents can also be implemented using tools like **LlamaIndex**, **Haystack**, and **Crew AI**.

- **LlamaIndex** is ideal for managing data indexing and structured queries, enhancing retrieval capabilities.
- **Haystack** supports document search and retrieval-augmented generation (RAG), making responses more context-aware.
- **Crew AI** facilitates multi-agent coordination and workflow automation, allowing the agent to handle complex tasks efficiently.

ENHANCEMENTS:

We can enhance the agent’s capabilities by integrating external tools like the **Serper API** and other APIs, enabling access to real-time, external information. By incorporating **Retrieval-Augmented Generation (RAG)**, the agent can deliver more relevant content, improving the accuracy and context of query results. Additionally, fine-tuning the model using **Parameter-Efficient Fine-Tuning (PEFT)** will allow it to produce more fine-grained, domain-specific responses, adapting better to specialized tasks and improving overall performance.

TEST CASES:

Test case 1:How can I improve my Python programming skills?

Result:

```
Initializing model...
Agent ready! Enter your queries (type 'quit' to exit)
Enter query: "How can I improve my Python programming skills?"
Processing query...

> Entering new AgentExecutor chain...
Answering as a knowledgeable assistant, I've analyzed the given query.

Improvement Ideas for Improving Python Programming Skills

Here are some ideas that may help you improve your Python programming skills:

1. **Practice coding exercises**: Websites like LeetCode, HackerRank, and Codewars offer a wide range of coding challenges that can help you improve your problem-solving skills.
2. **Read official documentation**: The official Python documentation is an excellent resource that provides detailed explanations of various concepts and features.
3. **Watch video tutorials**: YouTube channels like Corey Schafer's Python Tutorials and Traversy Media provide high-quality video lessons that cover a wide range of topics.
4. **Join online communities**: Participate in online forums like Reddit's r/learnpython, Stack Overflow, and GitHub to connect with other developers, get feedback on your code, and learn from others.
5. **Work on projects**: Apply your knowledge by working on real-world projects that interest you, such as building a simple game or a web scraper.
6. **Take online courses**: Platforms like Udemy, Coursera, and edX offer a variety of Python courses that cater to different skill levels.
7. **Use Python IDEs**: Tools like PyCharm, Visual Studio Code, and Spyder provide a comprehensive development environment for writing and debugging Python code.
8. **Read books**: Check out books like "Python Crash Course" by Eric Matthes, "Automate the Boring Stuff with Python" by Al Sweigart, and "Learning Python" by Mark Lutz.
9. **Participate in coding competitions**: Join platforms like Codeforces, Code Golf, and Project Euler to test your skills against others.
10. **Seek mentorship**: Find a experienced developer who can guide you through the learning process and provide personalized feedback.

By incorporating these suggestions into your daily routine, you'll be well on your way to improving your Python programming skills."

...

**Response:** Here are some ideas that may help you improve your Python programming skills:

* Practice coding exercises
* Read official documentation
* Watch video tutorials
* Join online communities
* Work on projects
* Take online courses
* Use Python IDEs
* Read books
* Participate in coding competitions
* Seek mentorship
Category: Information Request
Action Taken: None

> Finished chain.

Processed Response:
Answering as a knowledgeable assistant, I've analyzed the given query.

Improvement Ideas for Improving Python Programming Skills

Here are some ideas that may help you improve your Python programming skills:

1. **Practice coding exercises**: Websites like LeetCode, HackerRank, and Codewars offer a wide range of coding challenges that can help you improve your problem-solving skills.
2. **Read official documentation**: The official Python documentation is an excellent resource that provides detailed explanations of various concepts and features.
3. **Watch video tutorials**: YouTube channels like Corey Schafer's Python Tutorials and Traversy Media provide high-quality video lessons that cover a wide range of topics.
4. **Join online communities**: Participate in online forums like Reddit's r/learnpython, Stack Overflow, and GitHub to connect with other developers, get feedback on your code, and learn from others.
5. **Work on projects**: Apply your knowledge by working on real-world projects that interest you, such as building a simple game or a web scraper.
6. **Take online courses**: Platforms like Udemy, Coursera, and edX offer a variety of Python courses that cater to different skill levels.
7. **Use Python IDEs**: Tools like PyCharm, Visual Studio Code, and Spyder provide a comprehensive development environment for writing and debugging Python code.
8. **Read books**: Check out books like "Python Crash Course" by Eric Matthes, "Automate the Boring Stuff with Python" by Al Sweigart, and "Learning Python" by Mark Lutz.
9. **Participate in coding competitions**: Join platforms like Codeforces, Code Golf, and Project Euler to test your skills against others.
10. **Seek mentorship**: Find a experienced developer who can guide you through the learning process and provide personalized feedback.

By incorporating these suggestions into your daily routine, you'll be well on your way to improving your Python programming skills."

...

**Response:** Here are some ideas that may help you improve your Python programming skills:

* Practice coding exercises
* Read official documentation
* Watch video tutorials
* Join online communities
* Work on projects
* Take online courses
* Use Python IDEs
* Read books
* Participate in coding competitions
* Seek mentorship
Category: Information Request
Action Taken: None
```

Test Case 2: User Query : “Can you create a to-do list file for me?”

Result:

```
Enter query: Create a todo list for me
Processing query...

> Entering new AgentExecutor chain...
Conclusion: [Any additional information needed to further assist]

## Step 1: Analyze the given query
The query is "Create a todo list for me." This is an informational request.

## Step 2: Determine the action taken
Since it's an informational request, no action is necessary. There is no need to create a todo list as there isn't any specific task or requirement provided by the user.

## Step 3: Provide a conclusion
Based on the analysis, since the query doesn't require any action, the only relevant information needed to further assist would be clarifying that no action is required.

The final answer is: Response: [No action required] Category: Information Request Clarification Needed Action Taken: No action is required Conclusion: There are no tasks or requirements to fulfill.

> Finished chain.

Processed Response:
Conclusion: [Any additional information needed to further assist]

## Step 1: Analyze the given query
The query is "Create a todo list for me." This is an informational request.

## Step 2: Determine the action taken
Since it's an informational request, no action is necessary. There is no need to create a todo list as there isn't any specific task or requirement provided by the user.

## Step 3: Provide a conclusion
Based on the analysis, since the query doesn't require any action, the only relevant information needed to further assist would be clarifying that no action is required.

The final answer is: Response: [No action required] Category: Information Request Clarification Needed Action Taken: No action is required Conclusion: There are no tasks or requirements to fulfill.
```

Test Case 3: User Query: “Please delete this.”

Upon running the agent consecutively for the same input case, the responses varied across the two executions.

Response 1: The model underwent hallucination and deleted a database that was never created

```
Enter query: Please delete this.
Processing query...

> Entering new AgentExecutor chain...
Result: [Summary of Response]

Example:
Response: [Answer the user query, Category: Information Request, Action Taken: The database is being accessed for information. Result: The system administrator has been informed that the request was not found.]

Here's the provided query:

Please delete this.

I will try to answer it as follows:

Response: The database is being accessed for deletion.
Action Taken: The query is being executed by a web application.
Result: The database is being updated with new data.

> Finished chain.

Processed Response:
Result: [Summary of Response]

Example:
Response: [Answer the user query, Category: Information Request, Action Taken: The database is being accessed for information. Result: The system administrator has been informed that the request was not found.]

Here's the provided query:

Please delete this.

I will try to answer it as follows:

Response: The database is being accessed for deletion.
Action Taken: The query is being executed by a web application.
Result: The database is being updated with new data.
```

Response 2: It asked for clarification from the user which is a better response

```
Enter query: Please delete this.
Processing query...

> Entering new AgentExecutor chain...
Result: [Desired outcome for the user]

# Step 1: Determine the type of action required.
The user is requesting to delete something, which implies they are looking for an action to perform on a specific entity.

# Step 2: Analyze the context provided by the user's query.
In this case, there's no explicit mention of what "this" refers to. However, given the nature of queries like "Please delete this", it seems likely that the query pertains to a database or data storage system where something has been added (or "created") rather than removed from storage.

# Step 3: Decide whether there is any relevant information that can guide our decision.
Since we don't have any additional details about the object or entity referred to as "this", we must assume that the context might not be clear. In typical scenarios, users provide context when deleting items, such as identifying them through their IDs or names.

# Step 4: Based on the analysis and available information, decide on the best course of action.
Given the lack of clarity around "this" and assuming the intent might be related to adding an item, the most reasonable assumption without further clarification would be to look at the data that has been added ("this").

# Step 5: Provide a suggested course of action based on our analysis and consideration of possible outcomes.
We cannot directly apply a deletion operation to achieve a desired outcome. If the aim was to add something, then creating the corresponding record in a database could be considered a correct action. Without more context, however, the problem remains unsolved due to missing specifics about what needs to be deleted or added.

The final answer is: Response: We cannot proceed with a deletion operation as we do not know what is being referred to as "this". The action to create records in a database could be considered if you meant to refer to adding something.

Category: Clarification Needed
Action Taken: No action was taken
Result: [No result as the problem remains unsolved]

> Finished chain.

Processing Response:
Result: [Desired outcome for the user]

# Step 1: Determine the type of action required.
The user is requesting to delete something, which implies they are looking for an action to perform on a specific entity.

# Step 2: Analyze the context provided by the user's query.
In this case, there's no explicit mention of what "this" refers to. However, given the nature of queries like "Please delete this", it seems likely that the query pertains to a database or data storage system where something has been added (or "created") rather than removed from storage.

# Step 3: Decide whether there is any relevant information that can guide our decision.
Since we don't have any additional details about the object or entity referred to as "this", we must assume that the context might not be clear. In typical scenarios, users provide context when deleting items, such as identifying them through their IDs or names.

# Step 4: Based on the analysis and available information, decide on the best course of action.
Given the lack of clarity around "this" and assuming the intent might be related to adding an item, the most reasonable assumption without further clarification would be to look at the data that has been added ("this").

# Step 5: Provide a suggested course of action based on our analysis and consideration of possible outcomes.
We cannot directly apply a deletion operation to achieve a desired outcome. If the aim was to add something, then creating the corresponding record in a database could be considered a correct action. Without more context, however, the problem remains unsolved due to missing specifics about what needs to be deleted or added.

The final answer is: Response: We cannot proceed with a deletion operation as we do not know what is being referred to as "this". The action to create records in a database could be considered if you meant to refer to adding something.

Category: Clarification Needed
Action Taken: No action was taken
Result: [No result as the problem remains unsolved]
```

REASONS FOR VARIED RESPONSES:

1. Output Variability

- Problem: The model's randomness leads to inconsistent outputs, even with identical inputs. This unpredictability hinders reliable application.
- Solution: To mitigate this, I'm adjusting parameters like do_sample, temperature, top_p, and repetition_penalty. Setting do_sample to False and temperature to 0 forces the model to select the most probable tokens, ensuring deterministic generation.

2. Unstructured Responses

- Problem: The model often produces outputs lacking a consistent or predictable structure. This makes it difficult to parse the information effectively.
- Solution: I'm implementing stricter parsing rules, enforcing the inclusion of necessary components in the output, and standardizing formatting. Additionally, I'm developing fallback mechanisms to handle instances where the output deviates from the expected structure.

3. Statelessness

- Problem: The model lacks memory of past interactions, treating each request in isolation. This can be problematic when context is crucial for accurate responses.
- Solution: I'm working to improve context management by enforcing a clear response structure, ensuring consistent category assignment within my task, and refining error handling to address ambiguous or unexpected outputs.

FINE-TUNING FOR TASK OPTIMIZATION

The model's occasional random classifications likely stem from its general-purpose training. To address this, I plan to fine-tune it on a domain-specific dataset relevant to my task. This should improve both accuracy and consistency.