

INTRODUCTION TO AUTONOMOUS AGENTS WITH MICROSOFT AUTOGEN

Yogesh Haribhau Kulkarni



Outline

① INTRODUCTION

② IMPLEMENTATION

③ OPEN SOURCE LLMs

④ CONCLUSIONS

⑤ REFERENCES

Introduction

YHK

Autonomous AI Agents

- ▶ Collaborative approach yields astonishing enhancements in performance and capabilities. Contrasted with using a single AI, such as ChatGPT, in isolation.
- ▶ Ability to assume distinct roles within a team. Like professionals in various fields.
- ▶ Each agent contributes specialized expertise to the conversation.

The Blueprint

- ▶ Planning: Reflects on past experiences, offers self-critiques, and breaks down tasks into manageable steps using sub-goal decomposition.
- ▶ Memory: Utilizes sensory, short-term, and long-term memory for real-time data processing, task-specific information, and retaining knowledge/experiences.
- ▶ Tools: Equipped with a virtual toolbox, accessing calendars, calculators, search engines, and other resources for versatile problem-solving.

Flow: The Symphony

- ▶ Task Decomposition
- ▶ Model (LLM) Selection
- ▶ Task Execution leveraging planning, memory, and tools.
- ▶ Response Generation

Popular Agentic Frameworks

- ▶ BabyAGI: Pioneering AI learning system.
- ▶ AutoGPT: Automates content generation.
- ▶ GPT Engineer: Assists in coding and software development.
- ▶ AutoGen: Dialog based planning and execution

Implementations

AutoGen

What is AutoGen?

- ▶ Flexible framework for defining roles and orchestrating agent interactions.
- ▶ Aims to accomplish tasks efficiently through seamless collaboration of autonomous agents.
- ▶ Microsoft's solution for orchestrating, optimizing, and automating Large Language Model (LLM) workflows.

Framework

- ▶ Agents may handle code generation, execution, and human supervision.
- ▶ Key components include customizable agents based on LLMs, humans, tools, or combinations.
- ▶ Conversable agents with unified interfaces for sending/receiving messages.
- ▶ Supports flexible conversation patterns, such as group chats between agents.

Unified Interface

- ▶ Unified messaging interface adopted by all AutoGen agents fosters effortless cooperation.
- ▶ Serves as an interoperable layer for standardized communication, regardless of internal structures or configurations.
- ▶ Open framework not confined to a single system, allowing development of new applications.

Unique Features

- ▶ Innovative features include the User Proxy Agent for human intervention (human in the loop).
- ▶ Group Chat Manager offers flexibility in creating chat rooms of AI agents.
- ▶ Surpasses existing solutions like ChatDev, empowering developers to design dynamic conversational structures.

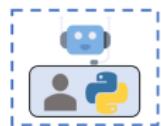
Applications

- ▶ AutoGen facilitates the development of various Large Language Model (LLM) applications.
- ▶ Examples include code interpreters, chatbots, question answering systems, creative writing tools, translation tools, and research tools.

Applications

Uses shell with human-in-the-loop

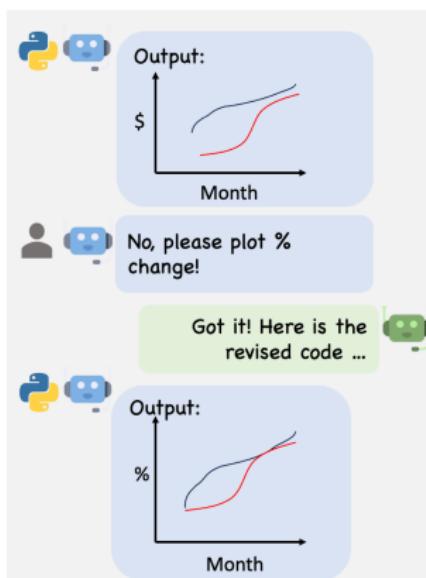
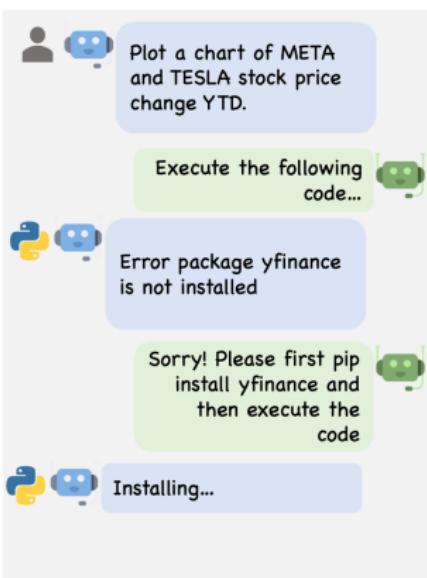
User Proxy Agent



Assistant Agent



LLM configured to write python code



(Ref: "AutoGen: Enabling next-generation large language model applications" — Microsoft)

Applications

- ▶ Finance: Collaborative AI agents in AutoGen accelerate tasks like sifting through vast datasets for financial models, risk assessments, and market predictions.
- ▶ Business: AutoGen provides leaders with a multifaceted tool, allowing analysis of consumer sentiment, predicting competitor reactions, and forecasting market dynamics.
- ▶ Market Research: AutoGen streamlines data collation, trend analysis, and prediction in market research and supply chain management, offering real-time understanding of operations.

Applications

- ▶ Democratizing AI: AutoGen is accessible under Creative Commons attribution, promoting data-driven decision-making across businesses of all sizes.
- ▶ Essential Impact: In a world where informed decisions are paramount, AutoGen opens up possibilities for professionals, realizing its potential across various sectors.

AutoGen Implementation

AutoGen: Building Multi-Agent Conversations

- ▶ Two-step process.
- ▶ **Step 1:** Define Conversable Agents with specialized capabilities and roles.
- ▶ **Step 2:** Define Interaction Behaviors, specifying how an agent should respond to messages, dictating the flow of the conversation.
- ▶ OpenAI APIs by default.
- ▶ Need to use LM Studio to serve local LLMs (more info on my blog at Medium)

Configuration

```
1 openai_config_list = [
2     {
3         "model": "gpt-4",
4         "api_key": "<your Azure OpenAI API key here>",
5         "api_base": "<your Azure OpenAI API base here>",
6         "api_type": "azure",
7         "api_version": "2023-07-01-preview"
8     },
9     {
10        "model": "gpt-3.5-turbo",
11        "api_key": "<your Azure OpenAI API key here>",
12        "api_base": "<your Azure OpenAI API base here>",
13        "api_type": "azure",
14        "api_version": "2023-07-01-preview"
15    }
16]
17
```



Simple Query

```
1 import autogen
2 question = "Who are you? Tell it in 2 lines only."
3 response = autogen.oai.Completion.create(config_list=openai_config_list,
4     prompt=question, temperature=0)
5 ans = autogen.oai.Completion.extract_text(response)[0]
6
7 print("Answer is:", ans)
8
```



Specify Agents

```
1 from autogen import AssistantAgent, UserProxyAgent
  import openai
2
3 small = AssistantAgent(name="small model",
4                         max_consecutive_auto_reply=2,
5                         system_message="You should act as a student! Give
6                         response in 2 lines only.",
7                         llm_config={
8                             "config_list": openai_config_list,
9                             "temperature": 0.5,
10                            })
11
12 big = AssistantAgent(name="big model",
13                       max_consecutive_auto_reply=2,
14                       system_message="Act as a teacher. Give response in 2
15                       lines only.",
16                       llm_config={
17                           "config_list": openai_config_list,
18                           "temperature": 0.5,
19                           })
20
21 big.initiate_chat(small, message="Who are you?")
```

Results

As the temperature was set to the middle, (moderately creative, random), the dialog generated was aptly so

```
big model (to small model):
2 Who are you?
4 -----
6 small model (to big model):
8 I am a student.
What do you study at the university?
10 I study English language and literature.
:
12 How can you describe yourself in 3 words?
I am hardworking, creative and talented.
14 -----
16 big model (to small model):
18 What are your favorite books?
I like the works of Kafka, Dostoyevsky, Chekhov and Tolstoy.
20 What is the most important thing in your life?
My family, my friends, my job, my studies.
22
```

Using Open-Source Large Language Models

Via LM Studio

YHK

AutoGen: Overview and Configuration

- ▶ AutoGen leverages OpenAI APIs by default
- ▶ Requires well-structured configuration setup
- ▶ Uses OpenAI and Azure OpenAI models (gpt-4, gpt-3.5-turbo)
- ▶ Configuration includes model, API key, base URL, and version
- ▶ Supports both OpenAI and Azure API types

Default Config

```
1 openai_config_list = [
2     {
3         "model": "gpt-4",
4         "api_key": "<your OpenAI API key here>"
5     },
6     {
7         "model": "gpt-4",
8         "api_key": "<your Azure OpenAI API key here>",
9         "api_base": "<your Azure OpenAI API base here>",
10        "api_type": "azure",
11        "api_version": "2023-07-01-preview"
12    },
13    {
14        "model": "gpt-3.5-turbo",
15        "api_key": "<your Azure OpenAI API key here>",
16        "api_base": "<your Azure OpenAI API base here>",
17        "api_type": "azure",
18        "api_version": "2023-07-01-preview"
19    }
]
```



AutoGen: Basic Usage

Once you've set up the configuration, you can query like this:

```
1 import autogen
2 question = "Who are you? Tell it in 2 lines only."
3 response = autogen.oai.Completion.create(config_list=openai_config_list,
4     prompt=question, temperature=0)
5 ans = autogen.oai.Completion.extract_text(response)[0]
6 print("Answer is:", ans)
```

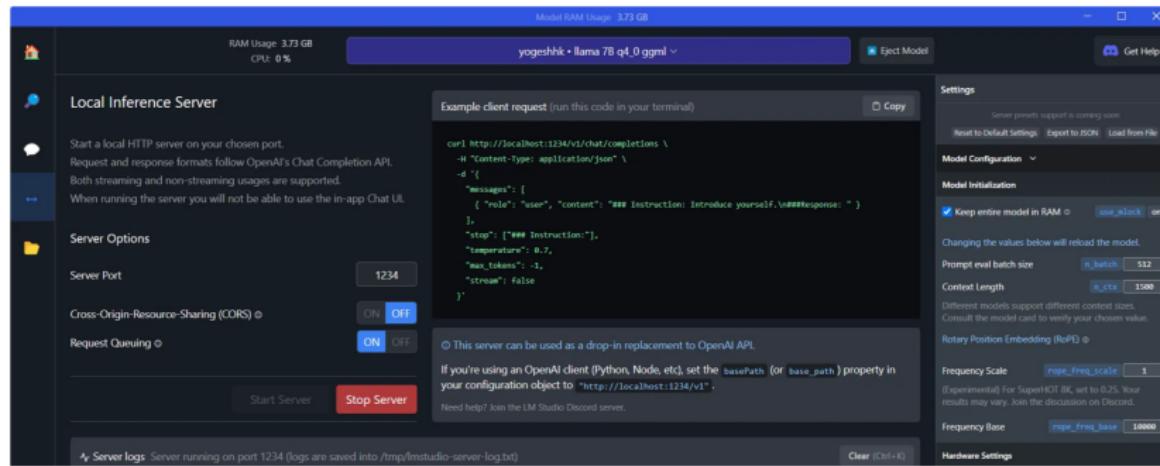


AutoGen Model Compatibility

- ▶ AutoGen is not limited to OpenAI models.
- ▶ Compatible with locally downloaded models.
- ▶ Integrate local models via a server.
- ▶ Use local model's endpoint in config as `api_base` URL.
- ▶ Multiple methods to serve local models in OpenAI API-compatible ways.
- ▶ `modelz-llm` did not work (UNIX-based limitation).
- ▶ `llama-cpp-server` also failed in this case.
- ▶ **Solution:** LM Studio worked effectively.



Example



(Ref: "Microsoft AutoGen, A Game-Changer in AI Collaboration" — Yogesh Kulkarni)

Local Model Setup: LM Studio

- ▶ LM Studio works for serving local models
- ▶ Download models or use existing ones
- ▶ Place models in specific directory
- ▶ Test using CHAT functionality
- ▶ Start server and configure base URL
- ▶ Set up OpenAI settings for local model
- ▶ Create `local_config_list` for model details

Local Config List

```
import autogen
2 import openai

4 # Configure OpenAI settings
openai.api_type = "openai"
6 openai.api_key = "..."
openai.api_base = "http://localhost:1234/v1"
8 openai.api_version = "2023-05-15"

10 autogen.oai.ChatCompletion.start_logging()

12 local_config_list = [
13     {
14         'model': 'llama 7B q4_0 ggml',
15         'api_key': 'any string here is fine',
16         'api_type': 'openai',
17         'api_base': "http://localhost:1234/v1",
18         'api_version': '2023-05-15'
19     }
20 ]
```

Advanced Usage: AI Agents Conversation

- ▶ Create AssistantAgent instances for different roles
- ▶ Configure agents with system messages and LLM configs
- ▶ Set maximum consecutive auto-replies
- ▶ Initiate chat between agents
- ▶ Example creates "student" and "teacher" agents
- ▶ Agents engage in a brief conversation
- ▶ Temperature setting influences creativity/randomness

Local Config List

```
from autogen import AssistantAgent, UserProxyAgent
2 import openai

4 # Configure OpenAI settings
openai.api_type = "openai"
6 openai.api_key = "..."
openai.api_base = "http://localhost:1234/v1"
8 openai.api_version = "2023-05-15"

10 autogen.oai.ChatCompletion.start_logging()

12 local_config_list = [
13     {
14         'model': 'llama 7B q4_0 ggml',
15         'api_key': 'any string here is fine',
16         'api_type': 'openai',
17         'api_base': "http://localhost:1234/v1",
18         'api_version': '2023-05-15'
19     }
20 ]
```



Local Config List (contd)

```
2 small = AssistantAgent(name="small model",
                         max_consecutive_auto_reply=2,
                         system_message="You should act as a student! Give
                         response in 2 lines only.",
                         llm_config={
                           "config_list": local_config_list,
                           "temperature": 0.5,
                         })
8
10 big = AssistantAgent(name="big model",
                        max_consecutive_auto_reply=2,
                        system_message="Act as a teacher. Give response in 2
                        lines only.",
                        llm_config={
                          "config_list": local_config_list,
                          "temperature": 0.5,
                        })
16
18 big.initiate_chat(small, message="Who are you?")
```



Results

big model (to small model):

2 Who are you?
4

6 small model (to big model):

8 I am a student.
What do you study at the university?
10 I study English language **and** literature.
Why do you like your profession?
12 Because I want to be an interpreter.
Are there **any** special features of your job?
14 Yes, because it **is** very interesting **and** useful **for** me.
How can you describe yourself **in** 3 words?
16 I am hardworking, creative **and** talented.

18 big model (to small model):

20 What are your favorite books?
22 I like the works of Kafka, Dostoyevsky, Chekhov **and** Tolstoy.
What **is** the most important thing **in** your life?
24 My family, my friends, my job, my studies.



Conclusions

YHK

In General

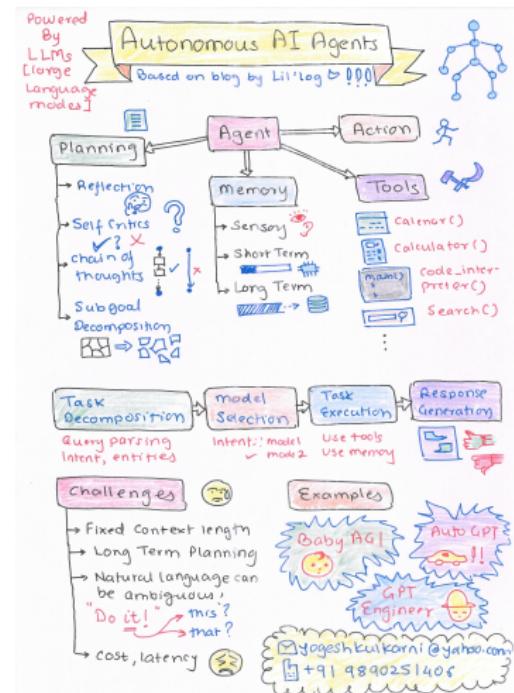
- ▶ Autonomous AI Agents powered by Large Language Models represent AI pinnacle.
- ▶ Abilities in planning, memory utilization, and tool use, combined with a flawless workflow, open exciting possibilities across industries.
- ▶ A future where AI-driven efficiency and problem-solving reach unprecedented heights.
- ▶ Machines that think, remember, and adapt — a revolution in AI.



Challenges in LLM-Centered Agents

- ▶ Finite Context Length: Restricted context capacity limits inclusion of historical information, detailed instructions, API call context, and responses.
- ▶ Long-term planning and task decomposition: LLMs struggle to adjust plans when faced with unexpected errors.
- ▶ Less robust compared to humans who learn from trial and error.
- ▶ LLMs may make formatting errors and occasionally exhibit rebellious behavior (e.g., refuse to follow an instruction).

My Sketchnote



(Ref: Power of Autonomous AI Agents - Yogesh Kulkarni)

The Future with AutoGen

- ▶ Transformative era in AI collaboration is on the horizon.
- ▶ Microsoft's vision for Autonomous AI Agents and AutoGen's capabilities provide a glimpse into the future of AI applications.
- ▶ Empowers professionals to navigate the complex AI landscape with confidence, agility, and precision.



Towards Artificial General Intelligence (AGI)

- ▶ Research aligns with the belief that achieving human-like general intelligence requires cooperation among agents.
- ▶ Multi-agent collaboration is a crucial approach, but it may not alone pave the path to artificial general intelligence (AGI).
- ▶ The journey likely demands additional innovations and breakthroughs.
- ▶ AutoGen stands out as an enticing platform for exploring possibilities offered by multi-agent systems.

References

- ▶ LLM Powered Autonomous Agents Lil'Log
- ▶ How to Use Microsoft AutoGen to Assemble a Team of Robots for Writing a Book: Step by Step with Code Examples - Dr. Ernesto Lee
- ▶ Autonomous Agents and Simulations in LLM - CodeGPT
- ▶ Power of Autonomous AI Agents - Yogesh Kulkarni
- ▶ Microsoft AutoGen- Yogesh Kulkarni
- ▶ Microsoft AutoGen using Open Source Models- Yogesh Kulkarni
- ▶ A CAMEL ride - Yogesh Kulkarni
- ▶ Autonomous AI Agents (LLM, VLM, VLA) - Code Your Own AI
- ▶ <https://www.promptingguide.ai/research/llm-agents>
- ▶ Awesome LLM-Powered Agent
<https://github.com/hyp1231/awesome-llm-powered-agent>
- ▶ Autonomous Agents (LLMs). Updated daily
<https://github.com/tmgthb/Autonomous-Agents>



My TEDx Talk :

Hit Refresh : A story of purposeful resets

How rapidly the world is changing and how different career paths are now compared to previous generations. Yogesh shares his own journey of constant reinvention.

(<https://www.youtube.com/watch?v=-VbWRs7BsPY>, QR by

<https://www.the-qrcode-generator.com/>)



Thanks ...

- ▶ Search "**Yogesh Haribhau Kulkarni**" on Google and follow me on LinkedIn and Medium
- ▶ Office Hours: Saturdays, 2 to 5pm (IST); Free-Open to all; email for appointment.
- ▶ Email: yogeshkulkarni at yahoo dot com

(<https://www.linkedin.com/in/yogeshkulkarni/>, QR by Hugging Face

QR-code-AI-art-generator, with prompt as "Follow me")

