Member-only story

Adding Empathy to Agentic AI

Fine-tuning AI Agents based on User Personas to improve their Empathy Quotient

12 min read · 5 days ago

Debmalya Biswas Follow

Listen Share More

1. Introduction

The discussion around ChatGPT (in general, Generative AI), has now evolved into Agentic AI. While ChatGPT is primarily a chatbot that can generate text responses, AI agents can execute complex tasks autonomously, e.g., make a sale, plan a trip, make a flight booking, book a contractor to do a house job, order a pizza. Fig. 1 below illustrates the evolution of agentic AI systems.

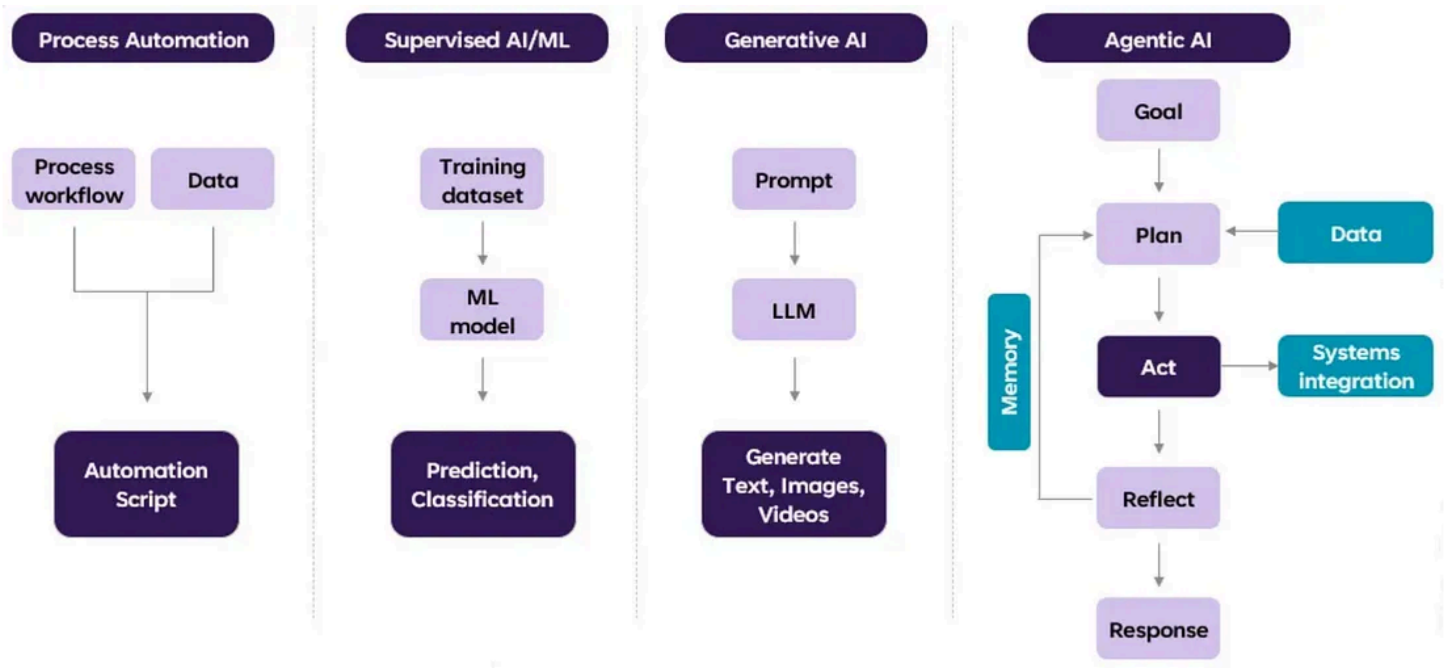

Fig. 1: Agentic AI evolution (Image by Author)

Bill Gates recently envisioned a future where we would have an AI agent that is able to process and respond to natural language and accomplish a number of different tasks. Gates used planning a trip as an example.

> Ordinarily, this would involve booking your hotel, flights, restaurants, etc. on your own. But an AI agent would be able to use its knowledge of your preferences to book and purchase those things on your behalf.

Fast-forward a few years, and we can foresee a lot more human-AI interaction in the not so distant future. Our homes, cars, and apps will likely be using AI assistants / agents to interact with us on a daily basis.

> *As AI permeates every corner of our lives, the quality of these interactions will contribute to our day-to-day mental health.*

We have learned the hard way that our interactions with social media can have significant impact on our mental health. AI agents / assistants are no different with the recent widely circulated news where a Google Gemini based chatbot basically told a student:

> Human … Please die.

So it makes sense to build these interactions with **empathy by design.** Empathy involves conveying that you can both understand and share the emotions of another person, sometimes also recognized through the metaphor of “emotion resonance.” Scientific studies have characterized the crucial ingredients of empathy as caring, kindness, and concern.

Empathy is of course one of the factors contributing towards improving the overall user **wellbeing**, which includes multiple facets, e.g., life satisfaction, purpose and accomplishment, mental and physical health, positive relationships, financial stability — illustrated in Fig. 2.

Fig. 2: User wellbeing core facets (Image by Author)

In a previous work, we have considered a reinforcement learning (RL) based approach to recommend an optimal wellbeing prompt / action to the user at the right time — based on the current state of the user. In general, from a technology point of view, empathy can be considered as one of the (key) enablers towards **personalizing** the human-AI interaction.

Sam Altman highlighted the personalization aspect in one of his recent tweets:

> In a few weeks, we plan to put out a new version of ChatGPT that allows people to have a personality … If you want your ChatGPT to respond in a very human-

> like way, or use a ton of emoji, or act like a friend, ChatGPT should do it.

The tweet of course became controversial for the below part — :)

> as part of our “treat adult users like adults” principle, we will allow even more, like erotica for verified adults.

but we will focus on the empathy / personalization aspect in this article. Analogous to fine-tuning of large language models (LLMs) to domain specific LLMs / small language models (SLMs),

> *we argue that fine-tuning of these (generic) AI agents will be needed with respect to user specific persona and context — to drive their responsible adoption.*

The key benefits of AI agent personalization include:

- Personalized interaction: The AI agent adapts its language, tone, and complexity based on user preferences and interaction history. This ensures that the conversation is more aligned with the user’s persona and communication style.
- Conversation context: The AI agent has a topical and situational awareness of the user persona and conversation topic, so that it can prioritize or highlight the relevant pieces of content — optimizing the interaction to achieve the user goal in a more engaging fashion.
- Proactive Assistance: The AI agent anticipates the needs of different users and offers proactive suggestions, resources, or reminders tailored to their specific tasks and corresponding urgency.

The rest of the article is structured as follows: we outline a reference architecture for agentic AI platforms in Section 2, extending the same to add an empathy / personalization layer to agents in Section 3 based on fine-tuning LLMs and RL techniques. Section 4 concludes the article and provides some directions for future work.

2. Agent AI Platform Reference Architecture

In this section, we highlight the key components of a reference agentic AI platform — illustrated in Fig. 3:

- Agent marketplace
- Orchestration layer
- Integration layer
- Shared memory layer
- Governance layer, including explainability, privacy, security, etc.

with the **Personalization** layer added in Section 3.

Fig. 3: Agentic AI platform reference architecture (Image by Author)

Given a user task, we prompt a LLM for the task decomposition — this is the overlap with Gen AI. Unfortunately, this also means that agentic AI systems today are limited by the **reasoning** capabilities of LLMs. For ex., the GPT4 task decomposition of the prompt

> Generate a tailored email campaign to achieve sales of USD 1 Million in 1 month, The applicable products

> and their performance metrics are available at [url]. Connect to CRM system [integration] for customer names, email addresses, and demographic details.

is detailed in Fig. 4: (Analyze products) — (Identify target audience) — (Create tailored email campaign).

Fig. 4: Agentic AI execution of a Marketing use-case (Image by Author)

The LLM then monitors the execution / environment and adapts autonomously as needed. In this case, the agent realised that it is not going to achieve its sales goal and autonomously added the tasks:
(Find alternative products) — (Utilize customer data to **personalize** the emails) — (Perform A/B testing).

Given the need to orchestrate multiple agents, there is a need for an **integration layer** supporting different agent interaction patterns, e.g., agent-to-agent API, agent API providing output for human consumption, human triggering an AI agent, AI agent-to-agent with human in the Loop. The integration patterns need to be supported by the underlying AgentOps platform. Andrew Ng recently talked about this aspect from a performance perspective:

> Today, a lot of LLM output is for human consumption. But in an agentic workflow, an LLM might be prompted repeatedly to reflect on and improve its output, use tools, plan and execute multiple steps, or implement multiple agents that collaborate. So, we might generate hundreds of thousands of tokens or more before showing any output to a user. This makes fast token generation very desirable and

makes slower generation a bottleneck to taking better advantage of existing models.

It is also important to mention that integration with enterprise systems (e.g., CRM in this case) will be needed for most use-cases. For instance, refer to the model context protocol (MCP) proposed by Anthropic recently to connect AI agents to external systems where enterprise data resides.

Given the long-running nature of such complex tasks, **memory management** is key for Agentic AI systems. Once the initial email campaign is launched, the agent needs to monitor the campaign for 1-month.

> *This entails both context sharing between tasks and maintaining execution context over long periods.*

The standard approach here is to save the embedding representation of agent information into a vector store database that can support maximum inner product search (MIPS). For fast retrieval, the approximate nearest neighbors (ANN) algorithm is used that returns approximately top k-nearest neighbors with an accuracy trade-off versus a huge speed gain.

Finally, the **governance layer**. We need to ensure that data shared by the user specific to a task, or user profile data that cuts across tasks; is only shared with the relevant agents (privacy, authentication and access control). Refer to my previous article on **Responsible AI Agents** for a discussion on the key dimensions needed to enable a well governed AI agent platform in terms of hallucination guardrails, data quality, privacy, reproducibility, explainability, etc.

3. Personalizing AI Agents to express Empathy

Users today expect a seamless and personalized experience with customized execution to meet their specific requirements. However, AI agent personalization remains challenging due to scale, performance, and privacy challenges.

User persona based personalization aims to overcome these challenges by segmenting the end-users of a service into a manageable set of user categories, which represent the demographics and preferences of majority of users.

In this article, we focus on **LLM agents**, which loosely translate to invoking (prompting) an LLM to perform natural language processing (NLP) tasks, e.g., processing topical data / documents, summarizing them, generating responses based on the retrieved data.

Given this, we can apply the below techniques to add the necessary empathy / personalization aspects:

- Prompt engineering
- Few-shot learning
- Context engineering: add more content expertise to the prompt
- LLM fine-tuning
- RL based optimization by incorporating continuous user feedback

> *Simple prompt engineering here includes assigning the LLM a role and providing specific instructions for the desired behavior.*

However, such prompting usually leads to a very wordy response with the underlying LLM

- first trying to understand the emotion,
- then articulating the triggers behind the emotion;
- ending in a (hallucinated) response further asking an open-ended question to invite more conversation.

Viewing the response holistically, it feels like it reached into a grab-bag of therapeutic techniques, pulled several out, and glued them together into a well-written fictional response -:)

A mix of few-shot learning and context engineering can be used to improve LLM performance here.

> *Few-shot learning consists of adding a few (e.g., 5–10) explicit examples to your prompt, to show the LLM "how to do things.". Context engineering further helps by providing additional content knowledge to the prompt, e.g., the kind of knowledge captured in therapy textbooks.*

In this article, we focus on primarily the last two techniques: LLM fine-tuning and RL based response optimization. The solution architecture to perform **user persona based fine-tuning of AI agents** is illustrated in Fig. 5.

Fig. 5: User persona based fine-tuning of AI agents (Image by Author)

The fine-tuning process consists of first **parameterizing** (aggregated) user data and conversation history and storing it as memory in the LLM via adapters, followed by fine-tuning the LLM for personalized response generation. The agent — user persona **router** helps in performing user segmentation (scoring) and routing the tasks / prompts to the most relevant agent persona.

For example, refer to the papers below for details of persona based LLM fine-tuning in educational, medical contexts, professional networks, respectively.

- EduChat: considers pre-training models on an educational corpus to establish a foundational knowledge base, and subsequently fine-tuning them on personalized tasks, e.g., essay assessment.
- LLM based Medical Assistant Personalization combines parameter-efficient fine-tuning (PEFT) with a memory retrieval module to generate personalized medical responses.
- LinkedIn retrieves feed recommendations from a pool of hundreds of millions of candidates by fine-tuning Llama 3 as dual encoder to generate high quality embeddings for both users (members) and content (items). This includes quantizing numerical features in the prompt to facilitate greater alignment between the retrieval and ranking layers.

3.1 User Data Embeddings

In this section, we focus on generating the agent — user interaction embeddings, which is a pre-requisite for both fine-tuning and/or real-time retrieval-augmented-generation (RAG) prompt context augmentation.

> *Fine-tuning AI agents on raw user data is often too complex, even if it is at the (aggregated) persona level.*

This is primarily due to the following reasons:

- Agent interaction data usually spans multiple journeys with sparse data points, various interaction types (multimodal), and potential noise or inconsistencies with incomplete queries — responses.
- Moreover, effective personalization often requires a deep understanding of the latent intent / sentiment behind user actions, which can pose difficulties for generic (pre-trained) LLMs — LLM agents.
- Finally, fine-tuning is computationally intensive. Agent-user interaction data can be lengthy. Processing and modeling such long sequences (e.g., multi-years' worth of interaction history) with LLMs can be practically infeasible.

A good solution reference to overcome the above issues is Google's work on User-LLMs. According to the authors,

> *USER-LLM distills compressed* **representations** *from diverse and noisy user interactions, effectively capturing the essence of a user's behavioral patterns and preferences across various interaction modalities.*

This approach empowers LLMs with a deeper understanding of users' latent intent (inc. sentiment) and historical patterns (e.g., temporal evolution of user queries — responses) enabling LLMs to tailor responses and generate personalized outcomes.

3.2 Reinforcement Learning based Personalization

In this section, we show how LLM generated responses can be personalized based on a reinforcement learning (RL) enabled recommendation engine (RE).

RL is a powerful technique that is able to achieve complex goals by maximizing a reward function in real-time. The reward function works similar to incentivizing a child with candy and spankings, such that the algorithm is penalized when it takes a wrong decision and rewarded when it takes a right one — this is reinforcement.

High-level, the RL based LLM response / action RE works as follows:

1. The (current) user emotion and agent interaction history are combined to quantify the user sentiment curve and discount any sudden changes in user sentiment;
2. leading to the aggregate reward value corresponding to the last LLM response provided to the user.
3. This reward value is then provided as feedback to the RL agent — to choose the next optimal LLM generated response / action to be provided to the user.

More concretely, we can formulate the integration of an RL enabled RE with an LLM based chat app as follows — illustrated in Fig. 6:

Fig. 6: Formulation of RL based Agent personalization (Image by Author)

Action (*a*): An action *a* in this case corresponds to a LLM generated response delivered to the user in response to a user task / prompt — as part of an ongoing agent interaction.

Agent (*A*): is the one performing actions. In this case, the agent is the chat app delivering LLM responses to the users, where an action is selected based on its policy (described below).

Environment: refers to the world with which the agent interacts, and which responds to the agent's actions. In our case, the environment corresponds to the user *U* interacting with the chat app. *U* responds to *A*'s actions, by providing different types of feedback, both explicit (in the form of a chat response) and implicit (e.g., change in user sentiment).

Policy (π): is the strategy that the agent employs to select the next based action (NBA). Given a user profile U_p, (current) sentiment U_s, and query / task U_φ; the policy function computes the product of the response scores returned by the NLP and RE respectively, selecting the response with the highest score as the NBA:

- The NLP engine (NE) parses the task / prompt and outputs a ranked list of responses.
- The recommendation engine (RE) provides a score for each response based on the reward function, and taking into account the use-case context, user profile, preferences, sentiment, and conversation history. The policy function can be formalized as follows:

Reward (*r*): refers the feedback by which we measure the success or failure of an agent's recommended action (response). The feedback can e.g. refer to the amount of time that a user spends reading a recommended article, or the change in user sentiment upon receiving a response. We consider a 2-step reward function computation where the feedback f_a received with respect to a recommended action is first mapped to a sentiment score, which is then mapped to a reward

$r(a, f_a) = s(f_a)$

where *r* and *s* refer to the reward and sentiment functions, respectively.

4. Conclusion

Agentic AI personalization has the potential to significantly accelerate agentic AI adoption by improving user satisfaction rates. In this article, we considered personalization by adding an empathy layer to human-AI interactions based on user personas.

We proposed a reference architecture for an agentic AI platform, and provided the details to implement an empathy / personalization layer for the platform with (a) agent-user router to perform user segmentation by mapping prompts to the most relevant agent persona — emotion, (b) leveraging agent-user interaction embeddings.

We also proposed an RL based optimization module that is able to incorporate user feedback (in terms of the quality and timeliness of agent recommendations / responses) to improve future interactions. We believe that the above aspects are critical to enabling an engaging human-AI interaction, in a responsible fashion keep user sensitivity in mind.

Agentic Ai · Empathy Design · Personalization · Large Language Models · Generative Ai Solution

Written by Debmalya Biswas

4.6K followers · 175 following

AI/ML, Privacy and Open Source | x-Nokia, SAP, Oracle | 50+ Patents https://www.linkedin.com/in/debmalya-biswas-3975261/

Responses (6)

Bgerby

What are your thoughts?

MENTOR HERO AI
2 days ago

empathy by design.

Actually, to be honest about the potential of the technology, it could be designed to emulate empathy. It can and will never be empathetic. I think it's far more human centered to be designing tech that is computing the contextual affect of humans... more

6 1 reply Reply

Hodman Murad
2 days ago

The challenge in designing empathetic AI lies not in its ability to mirror user emotions, but in recognizing when adaptation might undermine the user's deeper needs. A system that always gives you what you want could miss opportunities to offer what... more

11 1 reply Reply

Elfreda
7 hours ago

This is exactly what we've been talking about! The future of interaction is not about making users adapt to machines, but about having machines understand users.

5 1 reply Reply

See all responses

More from Debmalya Biswas and AI Advances

In Data Science Collective by Debmalya Biswas

Agentic AI MCP Tools Governance

Discovery and governance guidelines for AI Agents in the Enterprise

✦ Sep 16 284 7

In AI Advances by Nikhil Anand

I wasted months running slow LLMs before learning this

Why your LLM is running at just 10% of its potential speed

Oct 10 826 11

In AI Advances by Fareed Khan

How to Build an Advanced Agentic E-commerce WhatsApp Bot with Hyperstack AI Studio

Meta Webhooks, RAG-Powered Q&A, and Secure, Stateful Logic

Sep 26 596 7

In Data Science Collective by Debmalya Biswas

Guardrails for AI Agents

Use-case specific validation tests and guardrails generation for Agentic AI

Aug 4 430 10

See all from Debmalya Biswas

See all from AI Advances

Recommended from Medium

In Level Up Coding by Fareed Khan

Building an Agentic Deep-Thinking RAG Pipeline to Solve Complex Queries

Planning, Retrieval, Reflection, Critique, Synthesis and more

4d ago 801 4

Dr Nicolas Figay

Knowledge Graphs and Ontologies: Beyond the Dictionary Fallacy

Most knowledge graph practitioners treat ontologies as sophisticated dictionaries—structured vocabularies and entity hierarchies...

6d ago 87 2

Aruna Pattam

Agentic AI: Part 7— Governance in Agentic AI

In Part 6 of this series, we explored how observability transforms trust from an abstract belief into measurable assurance allowing...

6d ago 56 1

In CodeToDeploy by TechToFit - Master Your Life with Tech

I Tried Google's New AI Agents. It's a Gold Rush.

I spend my days deep in the world of AI, but every so often, something drops that makes me stop everything. This is one of those times...

Oct 10 164 3

In Artificial Intelligence in Plain English by Alpha Iterations

Build Agentic RAG using LangGraph

A practical guide to build Agentic RAG with complete project code

5d ago 14

In Agentic AI & GenAI Revolution by Yi Zhou

OpenAI Cofounder Warned of an AI Agent Crisis—Agentic Engineering Is the Way Forward.

Andrej Karpathy warns AI agents aren’t ready. Agentic Engineering is the next discipline to govern AI through human-AI partnership.

3d ago 12 4

See more recommendations