Agentic Recommender Systems: Foundations, Perspectives, and Lessons from Large Scale Deployments

Reza Yousefi Maragheh[1] Yashar Deldjoo[2] Ben Coleman[3] Jason Cho[4] Chi Wang[5]

[1]Walmart Global Tech, [2]Polytechnic University of Bari, [3]Google DeepMind, [4]PretaskAI [5]AG2AI

SIGIR 2026

Thought Process Evolution

Explain briefly Not Long!

TLDR

Multi-stage modeling framework for **highlight extraction**, **Theme-Aware Keyword Extraction model** (LLM-TAKE) utilizing Large Language Models (LLMs), focused on scalability and integrated evaluation

Objective of LLM-TAKE framework

- Summarizing products through highlights that are context and theme aware, and are diverse
- **Helps customers** in their decision journey through **swiftly understanding product** characteristics
- Overcome LLM-specific **challenges**:
  - **Scalability**
  - **Evaluating the results**
  - **Hallucinations** in LLMs

Highlights

Thought Process Evolution

What's the deal with this item?

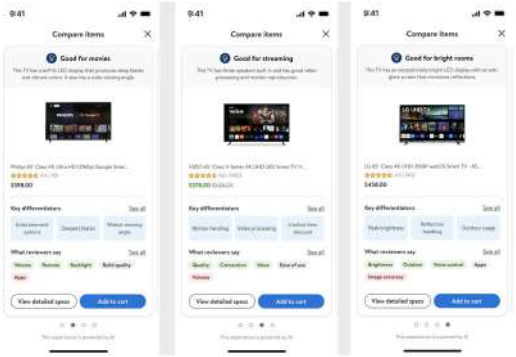
Hypothesis | **Planned Solution** | **Why this is harder?**

WE KNOW
Customers are looking for guidance and comparison between products they are interested in while browsing

IF WE
Show customers the most relevant use case of an item while they are comparing

THEN
We will make it easier for customers to make a decision

RESULTING IN
In an increase in the quality of ATC and higher conversion to purchases



But this one is harder. Why?
This falls upon largescale AI reasoning

Question:
If I give you item features, can You give me the usecases?
You should reason your way to generation and validation of usecases

Thought Process Evolution

Myriad of Problems

The Relevance Problem → **The Phrasing Problem** → **The Non-Informativeness Problem**

Good for Professional Photography

Good for Fashion Accessory!

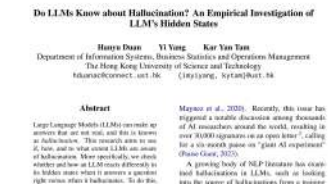
Good for Communication!

The question: How to deal with these edge cases in scale?

Thought Process Evolution

The Problem of Hallucination

- **Trustworthiness** of generated text by Language Models is an important element for large-scale implementations.
- Large Language Models have shown to **hallucinate** (Ji et al., 2023; Rawte et al., 2023)
- They generate responses that are **not consistent with instructions** (Duan et al., 2024).
- Thus, **evaluating the outcome of these models is crucial.**



Do LLMs Know about Hallucination? An Empirical Investigation of LLM's Hidden States

Hanyu Duan Yi Yang Kar Yan Tam
Department of Information Systems, Business Statistics and Operations Management
The Hong Kong University of Science and Technology
hduan@connect.ust.hk {imyiyang, kytam}@ust.hk

Abstract

Large Language Models (LLMs) can make up answers that are not real, and this is known as hallucination. This research aims to see if, how, and to what extent LLMs are aware of hallucination. More specifically, we check whether and how an LLM reacts differently in its hidden states when it answers a question right versus when it hallucinates. To do this, Mayaux et al., 2020). Recently, this issue has triggered a notable discussion among thousands of AI researchers around the world, resulting in over 30,000 signatures on an open letter[1], calling for a six-month pause on "giant AI experiments" (Future Of Life Institute, 2023).

A growing body of NLP literature has examined hallucinations in LLMs, such as looking into the source of hallucinations from a training

Classic Approaches

- **n − gram based** metrics
- **model-based** such as
  - Rouge Lin (2004),
  - BLEU Papineni et al. (2002),
  - BERTScore Zhang et al. (2019)
  - BARTScore Yuan et al. (2021)
- But have proven to show **weak correlation with human evaluations** Wang et al. (2023).

BERTSCORE: EVALUATING TEXT GENERATION WITH BERT

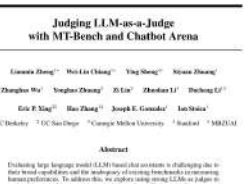
Tianyi Zhang*[1], Varsha Kishore*[1], Felix Wu*[1], Kilian Q. Weinberger[1], and Yoav Artzi[2]
[1]Department of Computer Science and [2]Cornell Tech, Cornell University
{vk352, fw245, kilian}@cornell.edu {yoav}@cs.cornell.edu
[2]ASAPP Inc.
tzhang@asapp.com

ABSTRACT

We propose BERTSCORE, an automatic evaluation metric for text generation. Analogously to common metrics, BERTSCORE computes a similarity score for each token in the candidate sentence with each token in the reference sentence. However, instead of exact matches, we compute token similarity using contextual embeddings. We evaluate using the outputs of 363 machine translation and image captioning systems. BERTSCORE correlates better with human judgments and

Neo-Classic Approaches

- Rapid evolution of LLMs, some researchers and practitioners have considered using LLMs for the evaluation purpose
  - Wang et al. (2023);
  - Zheng et al. (2024);
  - Maragheh et al. (2023);
  - Zhou et al. (2024).
- In this framework, due to high capability of **LLMs in mimicking human language capabilities**, they are used as a judge (Zheng et al., 2024)



Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena

Lianmin Zheng*[1] Wei-Lin Chiang*[1] Ying Sheng*[2] Siyuan Zhuang[1]
Zhanghao Wu[1] Yonghao Zhuang[1] Zi Lin[3] Zhuohan Li[1] Dacheng Li[1]
Eric P. Xing[4][5] Hao Zhang[3] Joseph E. Gonzalez[1] Ion Stoica[1]
[1] UC Berkeley [2] UC San Diego [3] Carnegie Mellon University [4] Stanford [5] MBZUAI

Abstract

Evaluating large language model (LLM) based chat assistants is challenging due to their broad capabilities and the inadequacy of existing benchmarks in measuring human preferences. To address this, we explore using strong LLMs as judges to

Thought Process Evolution

Images Credit: **Walmart.com**

Recommendation Explanation

For reasoning tasks,

- LLM as judge don't quite work really.
- Meaning it does not work.
- Yes, it does not work on complex tasks.

Is the usecase **"professional Photography"** relevant to this item given its features? **What is a professional camera? Really?**

Chat Completion

Metrics	Understandability		Groundedness	
	ρ	κ	ρ	κ
Vanilla	0.336	0.304	0.455	0.455
CoT	0.270	0.215	0.485	0.485
Self-Refine	0.167	0.144	0.412	0.412

Recommendation Explanation

Metrics	Relevance		Phraseness			
	ρ	κ	ρ	κ	ρ_{avg}	κ_{avg}
Vanilla	0.102	0.058	0.018	0.011	0.060	0.035
CoT	0.084	0.062	0.040	0.029	0.062	0.045
Self-Refine	0.075	0.046	0.026	0.012	0.050	0.029

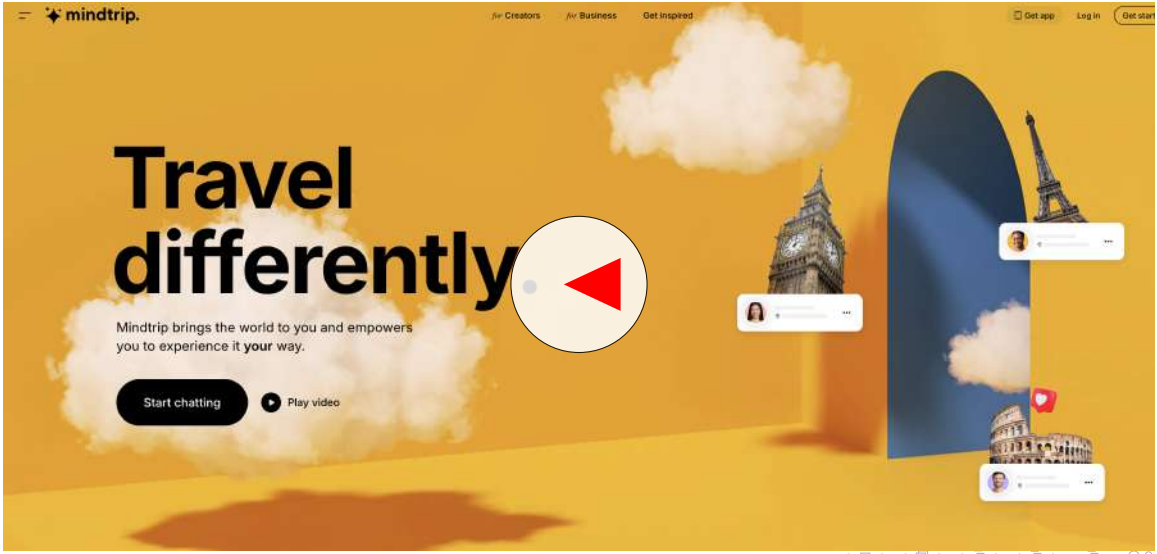
Kappa, rho <0.1 is not good at all!
Single Prompts Don't work!

Startups!

Startups

Image Credit: **dreamstime.com**

MindTrip


mindtrip.
for Creators
for Business
Get inspired
Get app
Log in
Get start
Travel differently
Mindtrip brings the world to you and empowers you to experience it your way.
Start chatting
Play video

MindTrip

Image Credit: **mindtrip.ai**

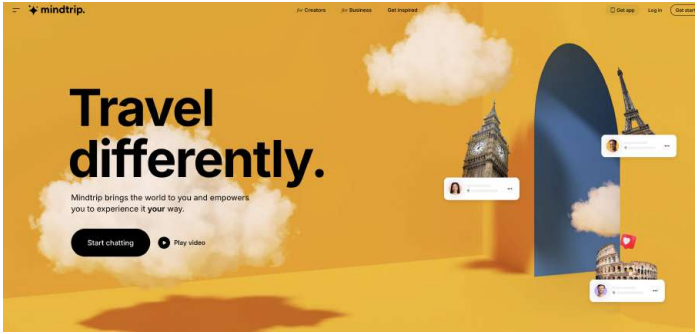


Figure: AI travel agent that recommends/optimizes itineraries, hotels, activities-Series A: Oct 2024

Milestones

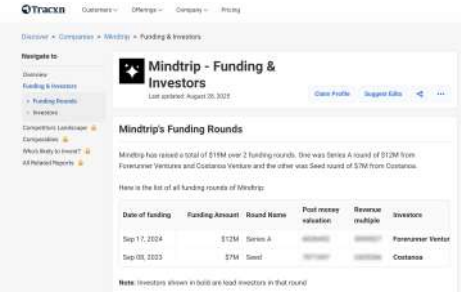
Credit: eightception.com

Key Growth Milestones

- 2023:
  - September: Seed round: $7M raised led by Costanoa Ventures.
  - November: Recognized among "Hot 25 Travel Startups for 2025" by PhocusWire.
- 2024
  - May: Official consumer product launch, showcasing conversational AI for trip planning. Launch of "Mindtrip for Business" for DMOs, media, and travel advisors.
  - Mid-year: Introduced "Start Anywhere" feature (create itineraries from a photo, blog, or video).
  - September: Series A, $12M raised, bringing total funding to $19M.
- 2025: Released a mobile app to enhance the on-trip experience.

MindTrip

Credit: **tracxn.com** and **prnewswire.com**



Moss said the funding with Amex Ventures couldn't come at a better time.

"Amex shares our vision for advancing the travel industry," said Moss. "With more travel planning and on-trip capabilities in the pipeline, we're excited for this next chapter and the collaborations ahead."

Kevin Tsang, managing director of Amex Ventures, said the company believes Mindtrip's approach can help transform the industry.

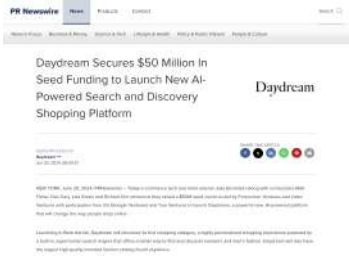
DayDream

Image Credit: **daydream.ing**

Figure: Chat-based fashion shopping agent for discovery and recommendations across thousands of brands. Seed: June 2024

DayDream

Credit: **prnewswire.com**



ALTA

Image Credit: **altadaily.com**

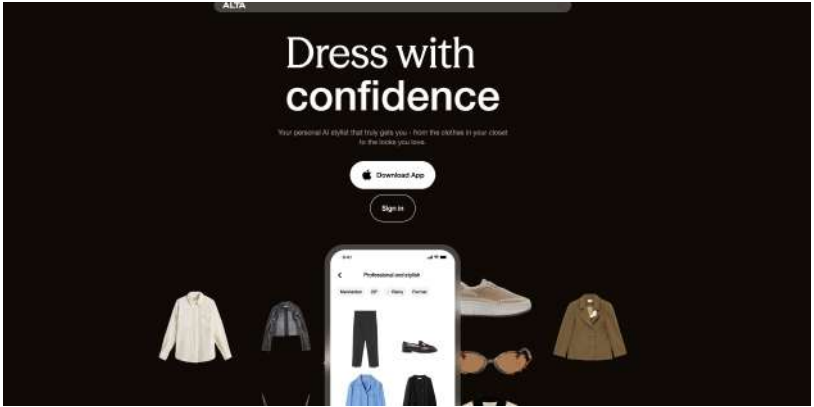


Figure: Agentic AI personal stylist that plans outfits and recommends products. Seed: June 2025

ALTA

Image Credit: **menlovc.com**

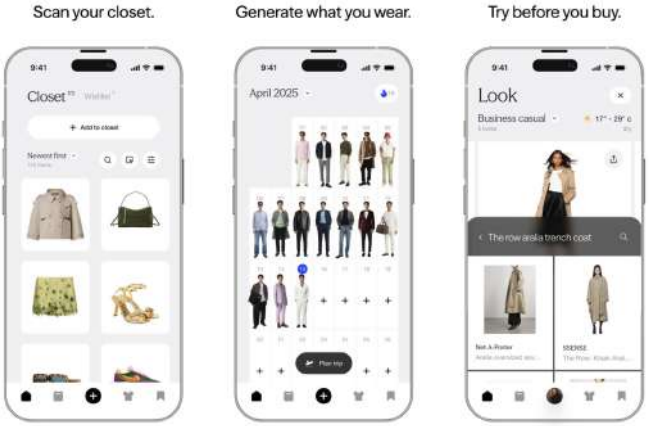
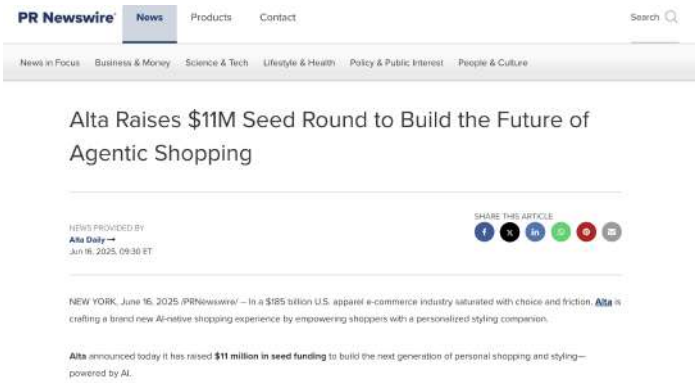


Figure: Alta App experience.

ALTA

Credit: **prnewswire.com**



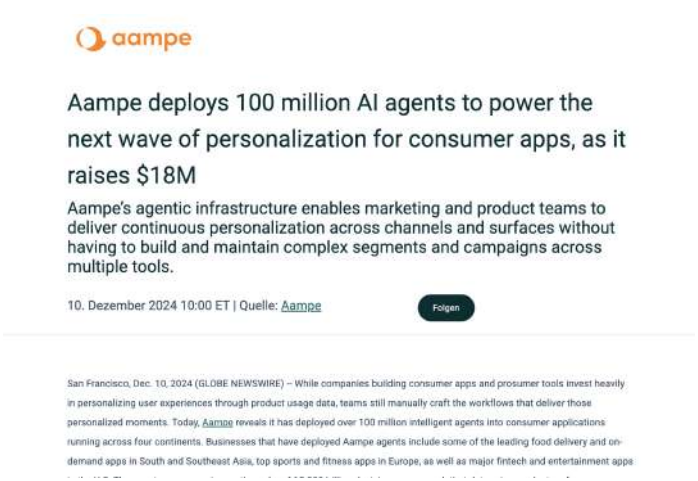
Aampe

Image Credit: **aampe.com**

Figure: Aampe's Agentic AI learns what works for each customer. Then it instantly adapts your messaging and delivers at optimal times to drive better engagement, growth and unlock valuable insights. Series A-Dec 2024

Aampe

Credit: **globenewswire.com**–All: 25.5M



aampe

Aampe deploys 100 million AI agents to power the next wave of personalization for consumer apps, as it raises $18M

Aampe's agentic infrastructure enables marketing and product teams to deliver continuous personalization across channels and surfaces without having to build and maintain complex segments and campaigns across multiple tools.

10. Dezember 2024 10:00 ET | Quelle: Aampe

Folgen

San Francisco, Dec. 10, 2024 (GLOBE NEWSWIRE) -- While companies building consumer apps and prosumer tools invest heavily in personalizing user experiences through product usage data, teams still manually craft the workflows that deliver those personalized moments. Today, Aampe reveals it has deployed over 100 million intelligent agents into consumer applications running across four continents. Businesses that have deployed Aampe agents include some of the leading food delivery and on-demand apps in South and Southeast Asia, top sports and fitness apps in Europe, as well as major fintech and entertainment apps

June 2026: MoEngage Aqcuired Aampe

Credit: yourstory.com

Mergers And Acquisitions ❯

MoEngage acquires San Francisco-based AI infra startup Aampe

The acquisition of Aampe marks a strategic move by MoEngage to expand its AI-powered marketing platform, helping brands deliver highly personalised customer experiences at scale.

YS Team YS • 16794 Stories

Constructor

Image Credit: **constructor.com**

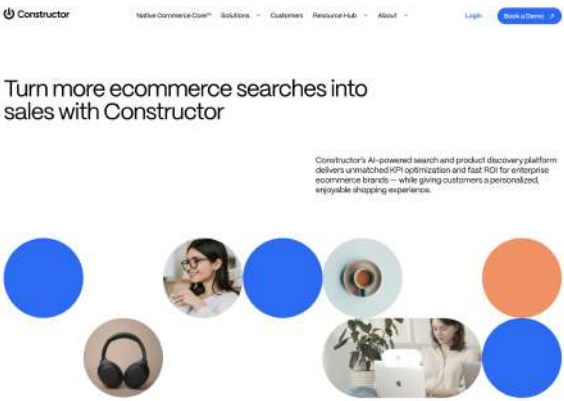
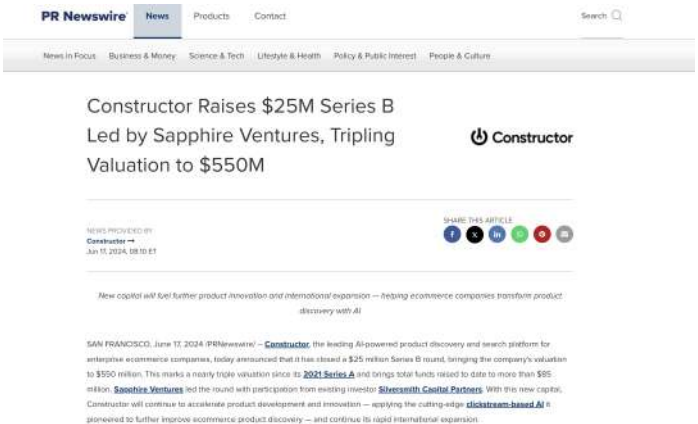


Figure: AI Shopping Agent + search & recommendations for enterprise ecommerce. Series B: June 2024

Constructor

Credit: **prnewswire.com**-all funds: 85M



PR Newswire News Products Contact Search

News in Focus Business & Money Science & Tech Lifestyle & Health Policy & Public Interest People & Culture

Constructor Raises $25M Series B Led by Sapphire Ventures, Tripling Valuation to $550M

Constructor

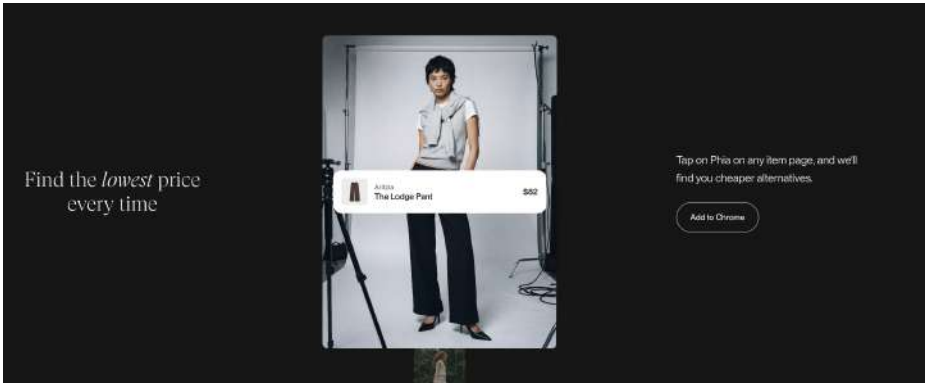
NEWS PROVIDED BY
Constructor
Jun 17, 2024, 08:10 ET

SHARE THIS ARTICLE

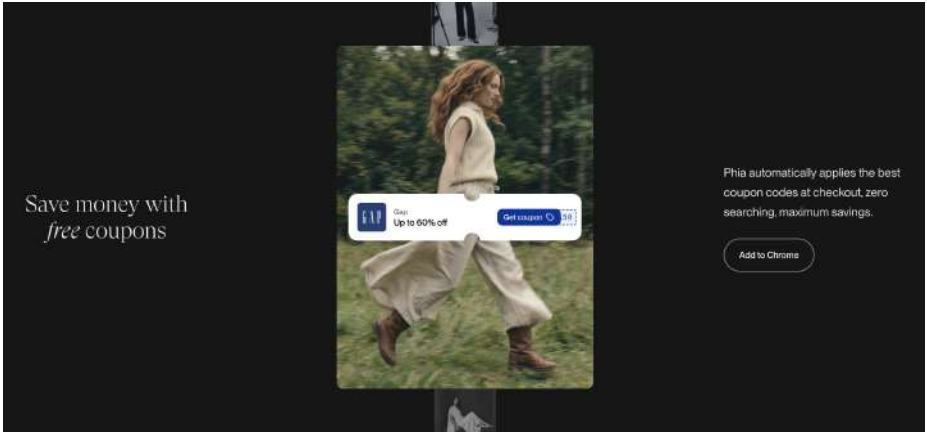
New capital will fuel further product innovation and international expansion — helping ecommerce companies transform product discovery with AI

SAN FRANCISCO, June 17, 2024 /PRNewswire/ – **Constructor**, the leading AI-powered product discovery and search platform for enterprise ecommerce companies, today announced that it has closed a $25 million Series B round, bringing the company's valuation to $550 million. This marks a nearly triple valuation since its **2021 Series A** and brings total funds raised to date to more than $85 million. **Sapphire Ventures** led the round with participation from existing investor **Silversmith Capital Partners**. With this new capital, Constructor will continue to accelerate product development and innovation — applying the cutting-edge **clickstream-based AI** it pioneered to further improve ecommerce product discovery — and continue its rapid international expansion.

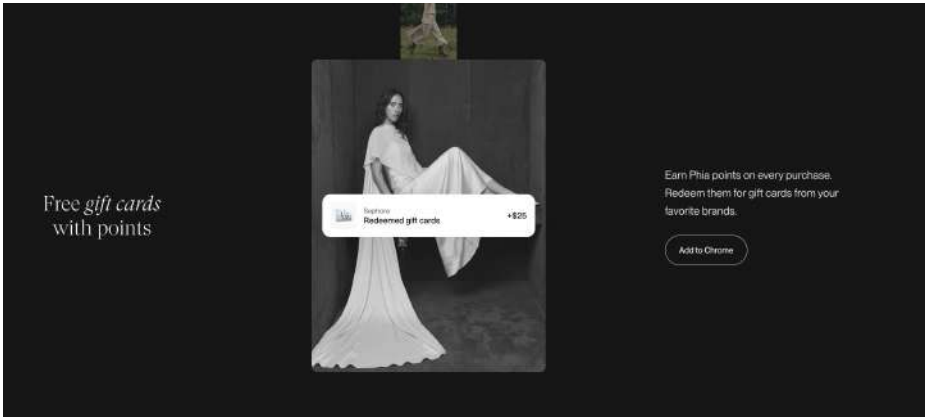
Phia-what it does?



Phia-what it does?



Phia-what it does?

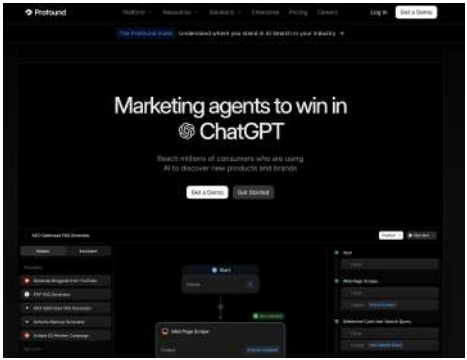


Phia funding, Credit**globenewswire.com**

- Launched April 2025,
- raised an $8M seed led by Kleiner Perkins in September 2025,
- closed an oversubscribed $35.5M Series A in January 2026
- total funding to $43.5M at a $185M valuation.
- Its shopping agent compares prices across retail and resale, calculates resale value, tracks price fluctuations, and applies discounts across 350M+ products from 9,600 brands — passing 1.5 million users with 11x revenue growth in its first year.

Profound

- Answer-Engine / Generative-Engine Optimization (GEO) platform
- the "SEO" layer for AI recommendations. Tracks when, where, and how a brand is cited across ChatGPT, Claude, Perplexity, Gemini, Copilot, and AI Overviews, then auto-generates content agents to influence it.



Profound's Funding: Credit: **fortune.com**

AI · TECH INVESTMENTS

Exclusive: As AI threatens search, Profound raises $96 million to help brands stay visible

By Lily Mae Lazarus
Reporter, News

February 24, 2026, 8:00 AM ET

The unicorn startup has fewer than 120 employees but is competing for the same AI talent that can work at OpenAI or Anthropic.
COURTESY OF PROFOUND

Figure: Series C: Feb 2026, $96M at $1B valuation total: $155M

The Plumbing: Who Pays When the Agent Buys?

Credit: **prnewswire.com**, **businesswire.com**, **tracxn.com**, All: $47.5M

Company	Layer	What it does	Funding
Basis Theory	Payment data	A processor-neutral vault for card data. Merchants keep control and can give an AI agent a safe token to pay with. Runs the Agentic Commerce Consortium.	**$33M Series B** (∼$50M total)
Skyfire	Identity	*Know Your Agent* (KYA / KYAPay): signed tokens that give an agent a verified identity and wallet, so it is not blocked as a bot at login or checkout.	**$9.5M** over 2 rounds
nekuda	Checkout	Agentic payments SDK built on two primitives: *Secure Agent Wallet* (delegated credentials) and *Agentic Mandates* (what the agent may buy, under which conditions, with spend limits).	**$5M Seed**

If an agent recommends, someone must let it pay.

Why so much $$$?

What is an agent anyway?

Image Credit: Quite Like You! Andy Shauf, youtube.com

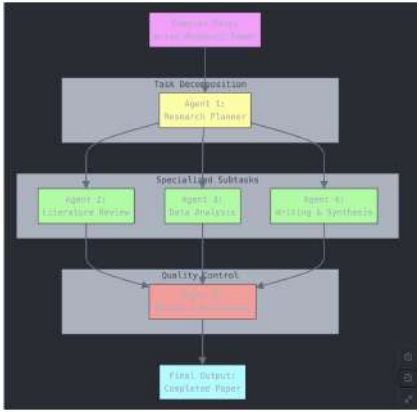
"I'll call "Society of Mind" this scheme in which each mind is made of many smaller processes. These we'll call agents. Each mental agent by itself can only do some simple thing that needs no mind or thought at all. Yet when we join these agents in societies-in certain very special ways-this leads to true intelligence".

From: **The Society of Mind, Marvin Minsky**

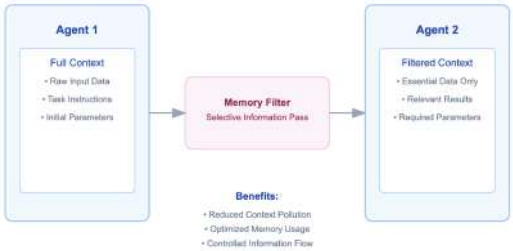
Why Multi-Agents?

Task Chunking

- **Reducing Reasoning Load**:Complex tasks often involve multiple interdependent steps that can overwhelm a single agent's decision-making capacity. Breaking these down into smaller, manageable chunks.
- **Failure Isolation**: Failures can be isolated to specific subtasks rather than requiring a complete restart of the entire process.
- **Specialized Agents**: Different agents can be optimized for different types of subtasks.



- **Reducing Context Pollution**: Controlled information flow between agents prevents context pollution and reduces the likelihood of conflicting or irrelevant information affecting decision-making processes.
- **Reducing Computational Overhead**: Selective memory passing allows for efficient resource utilization by only sharing essential information.



- **Beyond Language Processing**: Agents can be designed to interface with external tools like databases, APIs, or specialized software, expanding their capabilities beyond pure language processing.
- **Less expensive compute**: Task-specific tools can handle computationally intensive operations (like complex mathematical calculations or data processing) more efficiently than language models alone.

Why Multi-Agents?

We practice what we present:

- Past three slides (an only past three of them) are researched, written, drawn, evaluated, corrected by an agentic pipeline!

Gamma

Formalism

Formal Definition: LLM Agent

Definition

An **LLM Agent** is an intelligent system whose core decision-making and interaction capabilities are powered by a large language models.

$$A_{\text{LLM}} = \Big(\mathcal{M}, \mathcal{I}, \mathcal{O}, \mathcal{F}, \Omega\Big)$$

Formal Definition: LLM Agent

- $\mathcal{M}$: base model(s) for *understanding, reasoning, generation* (e.g., instruction-tuned transformer; optionally a mixture-of-experts).
- $\mathcal{I}$: *input space* observable by the agent (user text, images, traces, features, tool responses).
- $\mathcal{O}$: *output space* producible by the agent (natural language, structured JSON, function/tool calls, actions).
- $\mathcal{F} = \{f_k\}_{k=1}^{K}$: *external tools/APIs* the agent may invoke (retrieval, DB lookup, calculators, vision encoders, planners).
- Ω: *state & memory* enabling persistence across turns/sessions (short-term/working, episodic, semantic, procedural).

Multi-Agent System (MAS): Formal Definition

Definition

A **Multi-Agent System** is an ordered triple

$$\mathrm{MAS} \;=\; \Big(\mathcal{A},\, \mathcal{E},\, \Pi\Big),$$

where:

- $\mathcal{A} = \{A_1, \ldots, A_n\}$: finite set of agents; each A_i may be an LLM agent or another modular service.
- $\mathcal{E}$: shared *environment* exposing percepts/resources (APIs, UIs, simulators). Formally, a partially observable state space from which each agent receives observations and executes actions.
- $\Pi = (\mathbf{C}, \Gamma)$: *interaction protocol* constraining who may talk to whom and how.

Execution view: a run of MAS is a sequence of environment states and inter-agent messages that respect Π.

Interaction Protocol $\Pi = (\mathbf{C}, \Gamma)$

Communication matrix.

$$\mathbf{C} \in \{0,1\}^{n \times n}, \quad \mathbf{C}_{ij} = 1 \iff \text{messages of type } \gamma \in \Gamma \text{ are permitted from } A_i \to A_j.$$

- **Preset (static) routing:** $\mathbf{C}$ fixed at design time.
- **Autonomous (dynamic) routing:** channels toggled by guard functions $g_{ij} : \text{state} \to \{0,1\}$, e.g. $\mathbf{C}_{ij}(t) = g_{ij}(\mathcal{E}_t, \Omega_t^i, \Omega_t^j)$.

Message schemata Γ.

- *Performatives* (semantics): `INFORM`, `REQUEST`, `ACCEPT`, `REJECT`, ...
- *Serialization rules* (syntax): JSON schema, field names/types for machine readability.
- *Timing constraints*: ordering, deadlines, rate limits, retry/timeout policies.

Each $\gamma \in \Gamma$ defines both **syntax** and **semantics** so a receiver can parse and act on messages deterministically.

Minimal Example (Rec $\leftrightarrow$ Eval)

Agents: $\mathcal{A} = \{A_{\text{rec}}, A_{\text{eval}}\}$.

Environment:

$$\mathcal{E} = \big(\text{UserProfileDB},\ \text{ProductCatalogue},\ \text{BusinessRulesKB}\big).$$

Protocol:

$$\Pi = (\mathbf{C}, \Gamma), \quad \mathbf{C} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \Gamma = \{\texttt{candidate\_list},\ \texttt{compliance\_report}\}.$$

Message flow.

- $A_{\text{rec}} \to A_{\text{eval}}$: `candidate_list` (ranked JSON array $\{(s_1, \text{score}_1), \ldots, (s_L, \text{score}_L)\}$).
- $A_{\text{eval}} \to A_{\text{rec}}$: `compliance_report` (violations w.r.t. brand/policy $\in$ BusinessRulesKB).

Off-diagonal ones in $\mathbf{C}$ permit bidirectional messaging; zeros on the diagonal disallow self-messaging.

Autonomous Communication & Termination Conditions

Autonomous moderation of links.

$$\mathbf{C}_{\text{eval,rec}}(t) = \begin{cases} 1, & \text{NOT Halt}(t) \\ 0, & \text{Halt}(t) \end{cases} \quad \text{with} \quad \text{Halt}(t) \iff \big(\text{violations} = \varnothing\big) \vee (t - t_0 > \text{timeout}).$$

Protocol-level guards in Γ:

- *Stop condition*: end dialogue when all items pass compliance.
- *Retry/backoff*: bounded retries on transient failures.
- *Escalation*: on persistent violation, route to A_{policy} or human-in-the-loop.

Takeaway. Π can be preset yet *autonomously reconfigured* according to system state which balances accuracy with efficiency.

Formal Definition: Agent Memory Ω

Definition

The **memory** of an LLM agent is a finite set of stored items

$$\Omega \;=\; \Omega_{\text{short}} \;\cup\; \Omega_{\text{long}},$$

together with an *update* function $\mathcal{U}$ and a *retrieval* function $\mathcal{Q}$ that maintain and query it.

- Ω_{short}: *working memory*. The current context window. Readily accessible, bounded by token length, lost at session end.
- Ω_{long}: *persistent memory*, labelled by type:
  - EPI (*episodic*): what happened, when, under what circumstances.
  - SEM (*semantic*): timeless facts about the user, catalogue, or policy.
  - PROC (*procedural*): learned skills, scripts, tool routines.

Why it matters: an LLM is stateless by default. Without Ω, personalization is impossible.

Memory Item

Definition

A **memory item** is a triple $m = (k, v, \varsigma)$, where

$$k \in \mathbb{R}^{d_k}, \qquad v \in \mathbb{R}^{d_v}, \qquad \varsigma = (t,\ \ell,\ u).$$

- k: the **key**, an embedding or hashed identifier. The retrieval handle.
- v: the **value** payload. Text, dense vector, symbolic triple, or a pointer.
- ς: **metadata**. Timestamp t, type label $\ell \in \{\mathrm{EPI}, \mathrm{SEM}, \mathrm{PROC}\}$, update counter u.

Example. User states a guest has a gluten allergy:

$$k = \text{embed}(\text{“gluten allergy”}), \quad v = \text{“gluten”}, \quad \varsigma = (t{=}17{:}45,\ \ell{=}\mathrm{EPI},\ u{=}1).$$

The label EPI signals a one-off party constraint, not a durable taste.

Memory Operations: Update and Retrieval

Update. After context $\mathcal{C}_t$, the agent writes to memory:

$$\mathcal{U}: \ (\Omega_t, \ \mathcal{C}_t) \ \longmapsto \ \Omega_{t+1}.$$

Relevance. Given a task query τ, score each item:

$$S: \ \mathcal{T} \times \Omega \longrightarrow \mathbb{R}_{\geq 0}.$$

Retrieval. Return the K highest scoring items:

$$\mathcal{Q}(\tau, \Omega) \ = \ \widehat{\mathcal{C}} \ = \ \text{Top-K}_{m \in \Omega} \ S(\tau, m).$$

Naive choice: $S(\tau, m) = \cos\big(\text{embed}(\tau), \ k\big)$. For $\tau =$"find a gluten-free chocolate cake", this surfaces the allergy item.

Storage Modalities

- **Raw logs.** $\Omega^{\text{raw}} = [m_1, \ldots, m_T]$, retrieval is $\text{Tail}_L(\Omega^{\text{raw}})$. Trivial $\mathcal{U}$, but the prompt soon exceeds the window.
- **Summarisation.** $\mathcal{U}$ calls a summariser $\mathcal{R}_{\text{sum}} : \mathcal{C}_t \to \widetilde{\mathcal{C}}_t$. Saves tokens; recall fidelity is only as good as the summary.
- **Vector stores.** $k = e(v)$, retrieval by nearest neighbour. Scales to months of history, but blurs chronology.
- **Knowledge graphs.** $v = (s, r, o) \in \mathcal{E}^3$, queried exactly. Precise constraints and provenance, at the cost of curation.
- **Parametric.** $\mathcal{M}_{t+1} = \mathcal{M}_t \oplus \Delta_t$. Memory folded into the weights. Latency and safety limit real-time use.

Regulated Context Window

The agent blends the three long-term types under a token budget B:

$$\widehat{\mathcal{C}} = \widehat{\mathcal{C}}_{\text{PROC}} \cup \widehat{\mathcal{C}}_{\text{SEM}} \cup \widehat{\mathcal{C}}_{\text{EPI}}, \qquad \left|\widehat{\mathcal{C}}\right| \leq B.$$

The regulator is a knapsack problem.

$$\max_{\widehat{\mathcal{C}} \subseteq \Omega} \sum_{m \in \widehat{\mathcal{C}}} S(\tau, m) \qquad \text{s.t.} \qquad \sum_{m \in \widehat{\mathcal{C}}} |m| \ \leq \ B.$$

In Human Language. Retrieve best memory tokens w.r.t. the current query given that you have a budget constraint.

Let's Recap

Industrial Agentic RecSys-Usecases

Industrial Agentic RecSys-Usecases

Scenarios we will walk through

- **Interactive conversational Recommendation**
  - sub-agents coordinate (planner, retrieval, constraints), then build things like "Mickey-Mouse party" bundles.
- **User-simulation-Reco Eval**:
  - synthetic users and logging/summarization $\rightarrow$ stress-test policies pre-deployment.
- **Contextual & multi-modal Recommendation**:
  - vision+text (upload room photo and get a Boho set). Build complementarity & aesthetic coherence.
- **Recommendation Explanation**:
  - brand-consistent narratives that expose latent ranking logic ("inspired by your Disney purchases").

Interactive Recommendation

Interactive Recommendation

Agentic conversational planning for context-aware bundles

- Multi-turn conversation ⇒ refine constraints & intent
- Sub-agents for retrieval, validation, ranking, layout
- Memory-aware (STM/SEM/EPI/PROC) & tool-using

Interactive Recommendation

Task at a Glance

Definition (informal)

Interactive Recommendation engages in a **multi-turn** dialogue to identify, refine, and present suitable items, adaptively incorporating user feedback, contextual constraints, and memory/tool signals.

Why interactive?

- Real-world user intents are **open-ended** and **multi-step**
- Preferences emerge through **clarification** and **trade-offs**

Running example

"Mickey-Mouse themed birthday": decorations, gluten-free cake, favors; one conversation, many sub-goals.

Let $\mathcal{D}$ be conversation turns and $\mathcal{S}$ the item universe (SKUs). d_i and a_i are user and agent's utterances. A session at step t has transcript

$$\mathcal{C}_{1:t} = (d_1, a_1, \ldots, d_{t-1}, a_{t-1}, d_t), \quad d_i, a_i \in \mathcal{D}.$$

Define the interactive recommendation mapping

$$\Phi : \big(\mathcal{C}_{1:t},\, \mathcal{E}\big) \to \langle s_{(1)}, \ldots, s_{(L)} \rangle, \quad s_{(j)} \in \mathcal{S},$$

where, $\mathcal{E}$ is the shared space among agents. Φ outputs a ranked list of length L.

Objective with implicit constraints

$$\langle s_{(1)}, \ldots, s_{(L)} \rangle = \arg \max_{\langle s_1, \ldots, s_L \rangle} \sum_{j=1}^{L} \mathrm{Rel}\big(s_j \mid \mathcal{C}_{1:t}\big)$$

subject to constraints (theme, dietary, budget) encoded in $\mathcal{C}_{1:t}$ and available tools/environment $\mathcal{E}$.

- $\mathrm{Rel}(\cdot)$ may incorporate user history (SEM/EPI) and session cues (STM)
- Feasibility checked via tools (e.g., inventory, price, layout fit)

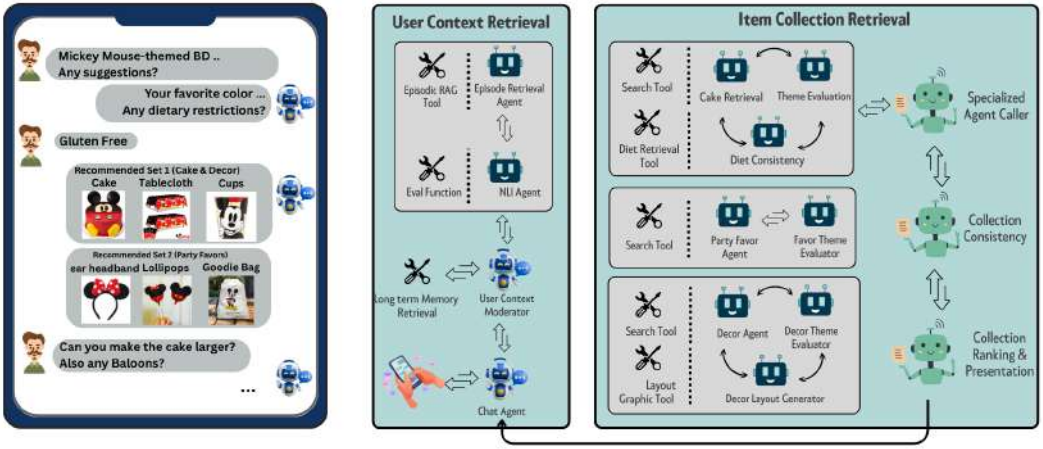
Objective

Dynamically adapt the Retrival/Ranking function Φ across turns to minimise user effort while satisfying latent sub-goals (e.g., *gluten-free chocolate cake*, *palette-consistent decor*).

- Spawn specialised sub-agents on demand (category retrievers, validators)
- Maintain **coherence** across turns via short/long-term memory
- Present **holistic** bundles (ranked collection + layout preview)

Interactive Recommendation

The Visual



Agent set $\mathcal{A}$

A_{chat} (chat), A_{epi} (episodic retrieval $\mathcal{Q}$), A_{nli} (episode vetting), A_{SAC} (specialised-agent caller),
A_{cake}, A_{decor}, A_{favor} (category retrieval), $A_{\text{col\_check}}$ (collection consistency), A_{rank} (ranking/presentation).

Environment & Protocol

$\mathcal{E} = \big(\text{ProductCatalogue}, \text{UserProfileDB}, \text{VectorDB}\big)$;
$\Pi = (\mathbf{C}, \Gamma)$ with $\mathbf{C}_{ij}{=}1$ iff A_j is spawned by A_i or $A_i{=}A_{\text{chat}}$;
$\Gamma{=}\{\texttt{query}, \texttt{episode\_list}, \texttt{tool\_call}, \texttt{item\_set}, \texttt{ranked\_list}\}$.

Interactive Recommendation

Message Flow & Orchestration

$$A_{\text{chat}} \xrightarrow{\texttt{query}} A_{\text{epi}} \xrightarrow{\texttt{episode\_list}} A_{\text{nli}} \xrightarrow{\texttt{validated\_episodes}} A_{\text{SAC}} \xrightarrow{\texttt{spawn}} \{A_{\text{cake}}, A_{\text{decor}}, A_{\text{favor}}\}$$
$$\xrightarrow{\texttt{item\_set}} A_{\text{col\_check}} \xrightarrow{\texttt{validated\_set}} A_{\text{rank}} \xrightarrow{\texttt{ranked\_list}} A_{\text{chat}}.$$

- A_{SAC} spawns micro-MAS blocks per category
- A_{nli} and $A_{\text{col\_check}}$ act as *gates* (reduce hallucinations, enforce theme/diet)
- A_{rank} personalises final presentation; `LayoutTool` renders a board

Combination of workflow and autonomy: A_{SAC} determines which specialized agents to call.

Stores

- **STM** (working): A_{chat} keeps last L turns of $\mathcal{C}_{1:t}$
- **EPI**: A_{epi} manages episode store via $\mathcal{U}/\mathcal{Q}$
- **SEM**: stable traits (colors, dietary rules, budget)
  - This may have its update pipeline.
- **PROC**: reusable prompts/templates, DB schema for autonomous tool use and DB read.
  - What are the internal tools and when should the agent call them?

Retrieve Retrieval Function $\mathcal{Q}$ for sub-goal τ of cake retrieval:

$$\begin{aligned}\widehat{\mathcal{C}} &= \mathcal{Q}(\Omega_t, \tau) \\ &= \{\text{Flavor\_pref: chocolate,} \\ &\quad \text{query: “Mickey-Mouse Cake”}\}.\end{aligned}$$

Feeds `SearchCakeAPI` with gluten-free, chocolate filter.

Update Update function for memory $\mathcal{U}$:
User: “gluten-free restriction.”

- Episode Distillation:

$$\widetilde{\mathcal{C}}_t = \text{“}\{\text{allergy: gluten}\}\text{”}$$

- Episodic Memory Update:

$$\Omega_{t+1}^{\text{EPI}} = \Omega_t^{\text{EPI}} \cup \{\text{allergy: gluten}\}.$$

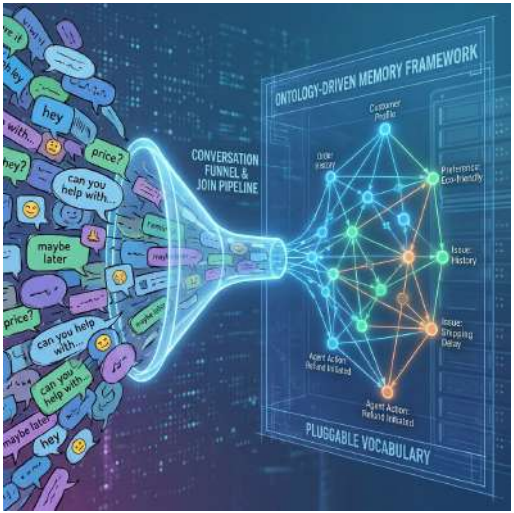
Interactive Recommendation

Tooling

- Tools $\mathcal{F}$: `SpecializedSearchAPI`, (graphic board of decor set).
- Sparse $\mathbf{C}$ limits communication.
- Observability: per-agent logs for `query`/`tool_call`/`ranked_list`
- Timeouts & fallbacks:
  - Control on hit rate,
  - Failure back up

- **Modularity**: category blocks are micro-MAS units, reusable across themes
- **Memory-aware personalisation**: injects user- and session-specific constraints , SEM/EPI yield relevant recommendation.
- **Error containment**: NLI & collection checks act as fact gates
- **Autonomy**: specialised-agent caller spawns workers on demand.
- **Richer User Experience**: interactive, using tools like `LayoutTool` offers better experience than a static recommendation.

Ontology Driven Memory



Why Vector Memory Is Not Enough

Two Sessions, One Forgotten Preference

Session 1 (three weeks ago)

User: I want Mickey-Mouse wallpaper for the kids' room.

Agent: Here are three Mickey-Mouse peel-and-stick options.

User: The second one is perfect.

Stored as: `embed(‘‘Mickey-Mouse wallpaper’’)`

Session 2 (today)

User: Help me put together a bed setup for that room.

Agent: Sure. Any color or style in mind?

User: *(has to say it all over again)*

Query: `embed(‘‘bed setup’’)`

Default starts for Memory?

- **Long-context stuffing**: history grows with every session. The model re-derives the same conclusions each turn, and latency grows with it.
- **Vector search / RAG**: optimised for surface similarity. It cannot decide which of two contradictory statements is current, and concepts do not travel across domains.
  - A stated interest in *Mickey-Mouse wallpaper* will not surface *Mickey-Mouse pillows* when the user later asks about a bed setup. The two live in different embedding neighbourhoods.

What is missing: a structured representation of the *customer*, separate from the conversations that produced it.

Definition

An **ontology** is a controlled vocabulary $\mathcal{V} = (\mathcal{N}, \mathcal{R}, \rho)$ declaring what may be known about a domain and how it behaves once stored.

- **Nodes** $\mathcal{N}$: the concepts the system is allowed to store, addressed by path, e.g. `preferences/color`. A closed statement of what can *ever* be remembered.
- **Edges** $\mathcal{R}$: relations between nodes, e.g. `has_field` from `preferences` into `preferences/color`.
- **Combination rules** $\rho : \mathcal{R} \to \{\text{APPEND}, \text{MERGE}, \text{REPLACE}\}$: attached to the edge, they decide what happens when a new fact meets an old one.

The **funnel** narrows one chat turn into **Atomic Memory Units** (AMUs), the smallest addressable piece of memory. The node list acts as the target schema.

User: *"I usually go for earth tones."*

```
{ "node_path" : "preferences/color",
  "value"     : "earth tones",
  "confidence": "high",
  "evidence"  : "I usually go for earth tones",
  "source"    : { "customer_id"     : "customer_001",
                  "conversation_id" : "conv-1" } }
```

The funnel only *extracts*. It does not decide how this fact relates to anything stored earlier.

The **join** folds each AMU into what is already stored. The rule comes from the edge, not from code.

- APPEND: the new fact *extends* the old. *Earth tones*, then *navy*: keep both. A palette broadens.
- REPLACE: the new fact *supersedes* the old. A budget of $50 that becomes $80. The latest value wins.
- MERGE: conflicting facts *coexist and are ranked*.
  - by constraint priority: a peanut allergy outranks a later satay request
  - by confidence: a stated size promotes over an inferred one
  - by recency with decay: an old "no" fades unless repeated

Agentic Recommendation Evaluation

Task Definition

Simulation-based evaluation (informal)

Generate **synthetic user behavior** to stress-test recommendation policies *before* live deployment.

Core mappings

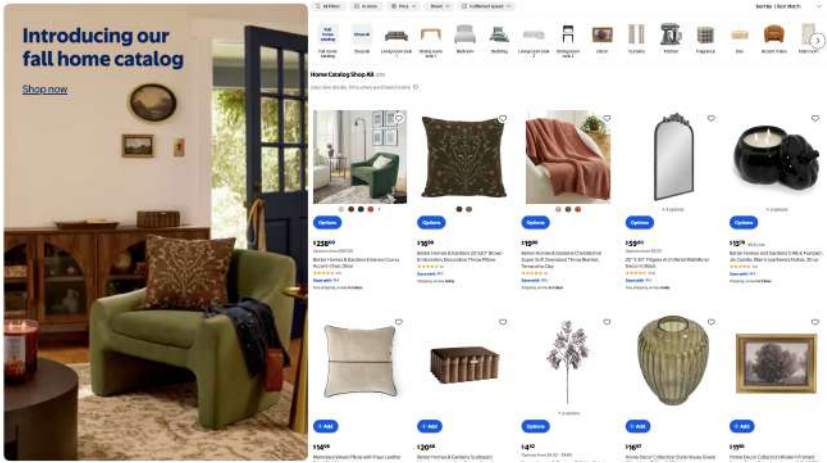
$$R_\phi : \mathcal{X} \to \mathcal{S}^L \quad \text{(recommender maps state } x \text{ to } L \text{ items)}$$

$$\mathcal{M}_{\theta,\omega} : \mathcal{X} \times \mathcal{S}^L \to \mathcal{A}_{\text{user}} \quad \text{(user simulator maps } (x, \text{list}) \text{ to an action)}$$

$\mathcal{A}_{\text{user}} \in \{\text{Select}, \text{Not Select}\}$ or richer (e.g. $\{$Click,Pass,Purchase$\}$).

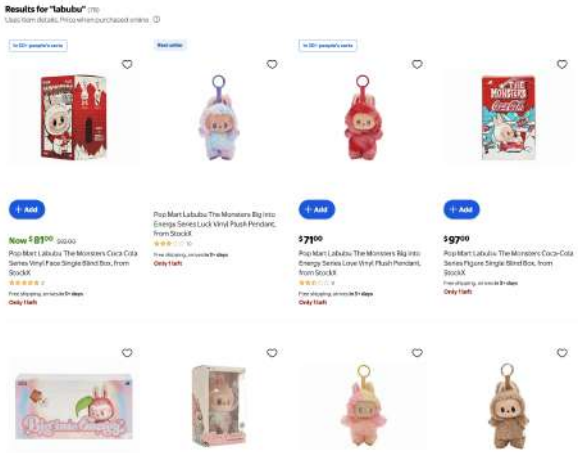
Agentic Recommendation Evaluation

Fall Seasonal Recommendation



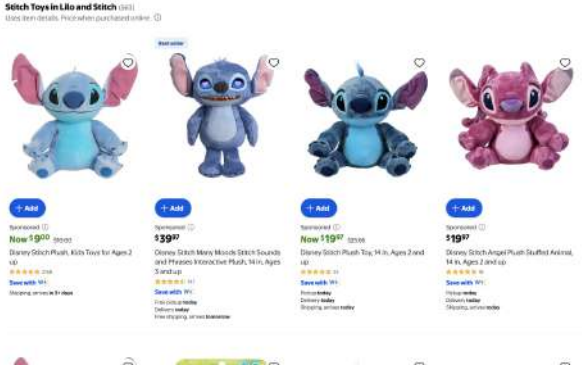
User Simulation

What is a new trend?



User Simulation

What is a new trend?



High-Level Goals (I)

- **De-risk A/B:** approximate user responses when real traffic is scarce or expensive
  - Sometimes, duration of the recommendation being live is short-lived (no time for AB),
  - Example: Seasonal Recommendation.
- **Fast iteration:** offline loops to evaluate design choices, ranking features, and guardrails
  - AB tests by design are slower (specially if we wait for stat sig results on niche general merchandise items)
- **Coverage:** explore edge cases (long-tail preferences, rare journeys) at scale
  - Hard to perform AB test on user cohorts,
  - Example: all new users with account opening in the last year.
- **Safety:** detect regressions, policy violations, and mode collapse before serving
  - Better control on the edge cases because we can simulated edge cases at scale.

High-Level Goals (II)

- **Measurement:** CTR/CVR, diversity, novelty, calibration, fairness, robustness
  - Should optimize on the above metrics,
  - Hard to track for customer cohorts.
- **Session granularity:** short (*quick browse*) vs. long (*deliberative*)
  - In each AB test we want to analyse short term effect and long term effect of recommendation,
  - Short term: purchase within session,
  - Longer term: product return patterns,
  - Longerer term: User loyalty.
- **Population realism:** sample (θ, ω) to reflect heterogeneous users
  - θ: can represent user profiles.
  - ω: we can add controlled noise to better simulate user variability withing cohorts.
- **Explainability:** log rationales and session summaries for root-cause analysis

Formal Definition (I)

In Recommendation Evaluation and User simulation task

- We define a user/platform state $\mathcal{X}$,
- Recommend $\mathcal{S}^L$,
- Observe user behavior $\mathcal{A}_{\text{user}}$.

$$R_\phi : \mathcal{X} \to \mathcal{S}^L, \qquad x_t \mapsto s_t^{(L)}$$
$$\mathcal{M}_{\theta,\omega} : \mathcal{X} \times \mathcal{S}^L \to \mathcal{A}_{\text{user}}, \qquad (x_t, s_t^{(L)}) \mapsto a_t^{\text{user}}$$

Iterated for $t = 1, \ldots, T$ to generate trajectories $\{x_t, s_t^{(L)}, a_t^{\text{user}}\}_{t=1}^T$.

Formal Definition (II)

Task objective

Task objective is to estimate any relevant metrics for user interaction,

$$\Psi(R_\phi, \mathcal{M}_{\theta,\omega}) = \mathbb{E}\Big[\sum_{t=1}^{T} g\big(x_t, s_t^{(L)}, a_t^{\text{user}}\big)\Big],$$

where g can encode Select/Purchase reward, diversity/penalty, fairness, etc.

- Estimate Ψ under controlled distributions of (θ, ω)
- Simulate and estimate results via Monte Carlo simulation.

Formal Definition (III): Population & Estimation

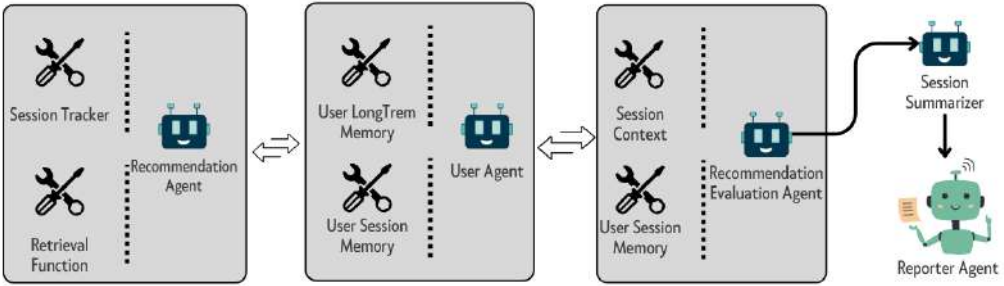
- **Population:** $(\theta, \omega) \sim P$ (PreferenceSampler)
- **Monte Carlo estimator:**

$$\widehat{\Psi} = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T} g\big(x_t^{(n)}, s_t^{(L,n)}, a_t^{\mathsf{user}(n)}\big)$$

- **Risk-aware variants:** Mearure metrics like
  - CVaR@ k,
  - worst-case across cohorts,
  - per-segment fairness gaps.

Architecture

The Visual



System Architecture: Agents

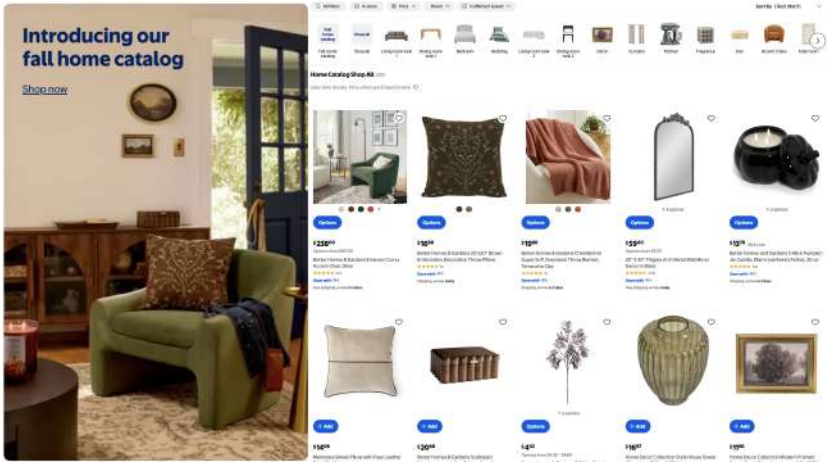
$\mathrm{MAS}_{\mathsf{sim}} = (\mathcal{A}, \mathcal{E}, \Pi)$

$$\mathcal{A} = \{A_{\mathsf{rec}}, A_{\mathsf{user}}, A_{\mathsf{note}}, A_{\mathsf{eval}}, A_{\mathsf{summ}}, A_{\mathsf{report}}\}$$

- A_{rec}: wraps R_ϕ; calls `SearchAPI`, `SessionTracker`
- A_{user}: implements $\mathcal{M}_{\theta,\omega}$ with structured memory
- A_{note}: durable logging of $(x_t, s_t^{(L)}, a_t^{\mathsf{user}})$
- A_{eval}: computes g_t; emits `eval_event`
- A_{summ}: *compresses* sessions via $\mathcal{R}_{\mathsf{sum}}$
- A_{report}: aggregates, tests, produces PDF/CSV dashboards

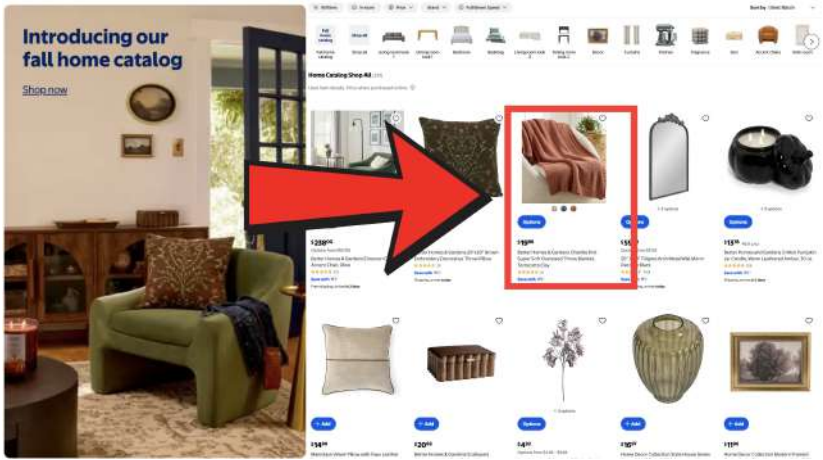
User Simulation

Fall Seasonal Recommendation



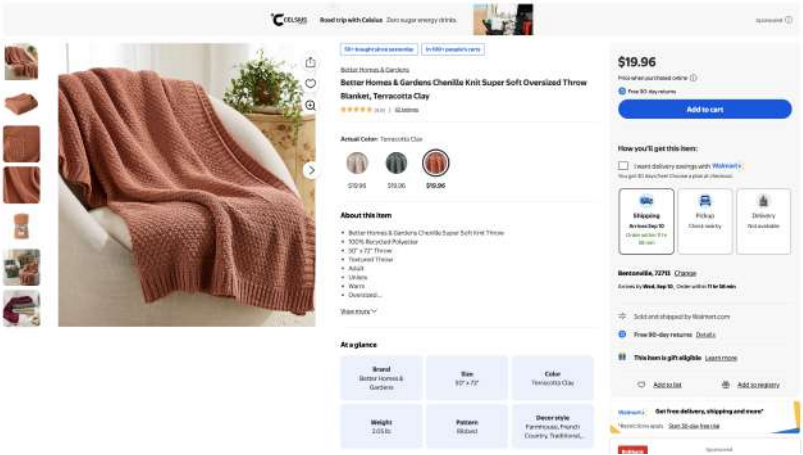
User Simulation

Fall Seasonal Recommendation



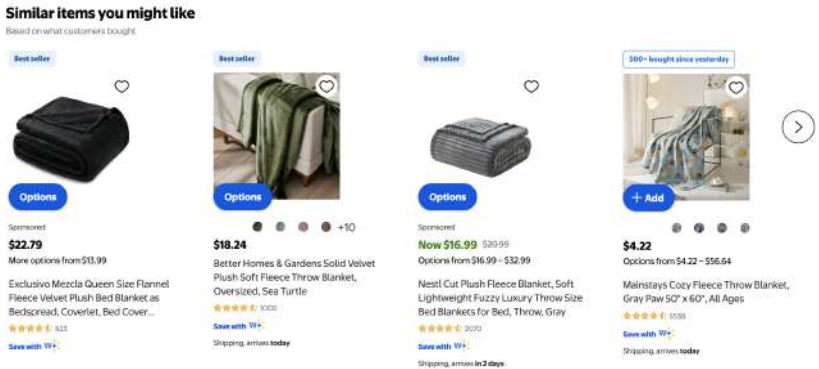
User Simulation

Fall Seasonal Recommendation



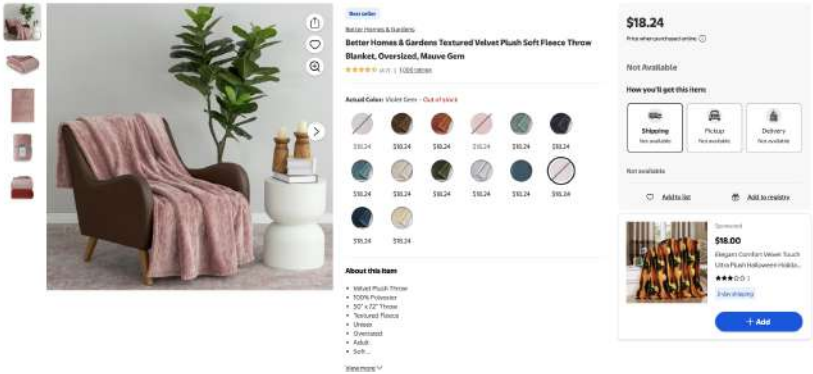
User Simulation

Scroll Down-Click on second item thrid color



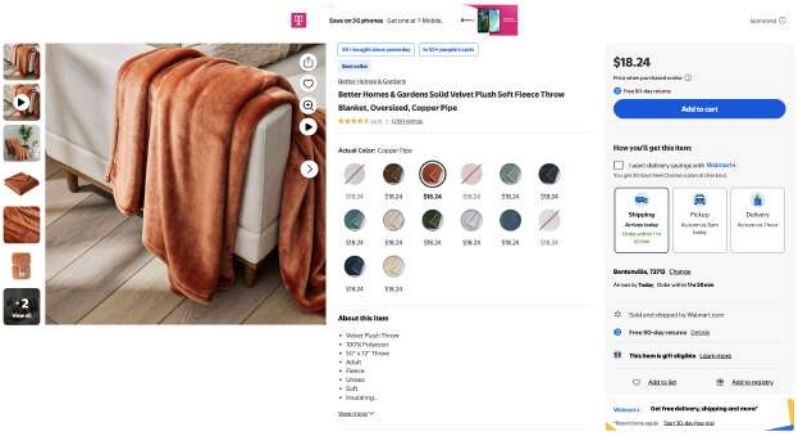
User Simulation

Color Out of Stock!



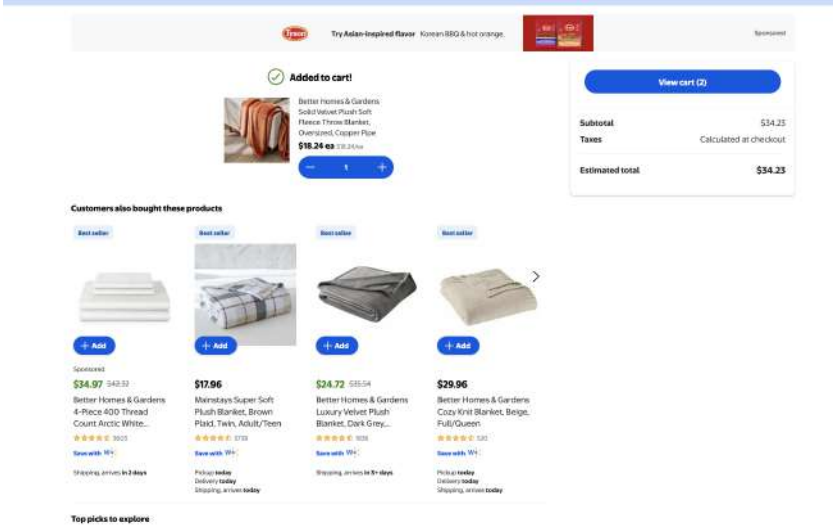
User Simulation

Add to correct the color you like



User Simulation

Added to cart-Rest of the Journey

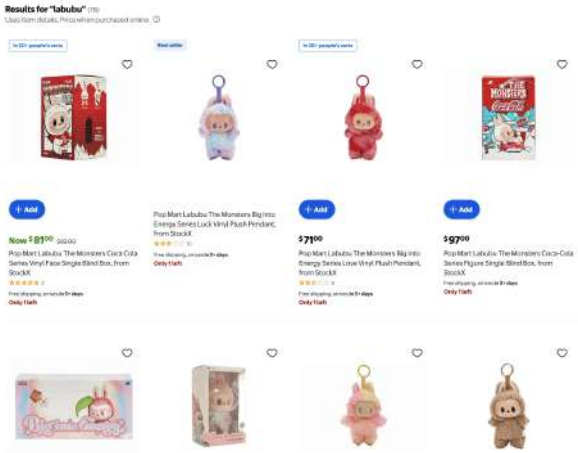


Benefits (I)

- **Cost-effective experimentation:** run $N \gg 1$ users in parallel; compare ϕ variants safely
  - Can study cohort groups better,
  - more granuality.
- **Cold-start mitigation:** sample sparse/unseen θ to probe failure modes early
  - new trends,
  - new items,
  - new users
- **Repeatability:** deterministic seeds $\Rightarrow$ reproducible KPI deltas
  - With different language models,
  - Same LLM to analyze variations in user journey trajectories

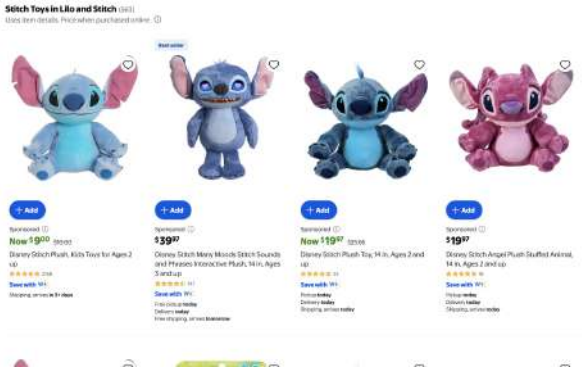
User Simulation

What is a new trend?



User Simulation

What is a new trend?



Benefits (II)

- **Bias diagnosis:** aggregate trends (price sensitivity, popularity bias, demographic skew)
  - Informed edge case handling,
  - Informed recommendation algorithm design.
- **Memory-driven realism:** SEM/EPI/PROC capture carry-over and bounded user attention
  - What is the effect on other recommendation algorithm?
  - what is the effect on long term user loyalty?
  - what is the effect of previous purchases in current user session behavior?
- **Modular extensibility:** add fairness/robustness evaluators without retraining R_ϕ
  - Custom metric evaluation.

Contextual & Multi-Modal Recommendation

fill the image with Boho style furniture

Dry Skin?

Figure: How can you connect a Humidifier to a Moisturizer?

Task Definition

Contextual & multi-modal recommendation (Informal)

Recommend concept-oriented, while adapting to different modalities.

Contextual & multi-modal recommendation (formal)

Let $\mathbf{x} \in \mathcal{T}$ be textual query (e.g. "Bohemian, earthy colors"), $\mathbf{v} \in \mathcal{V}^K$ a set of visuals, and $\mathbf{u} \in \mathcal{N}$ a long-term user profile. We model

$$\mathcal{R}_\phi : \ (\mathbf{x}, \mathbf{v}, \mathbf{u}) \ \longrightarrow \ \widehat{\mathbf{y}} = \langle s^1, s^2, \dots \rangle, \quad s^i \in \mathcal{S},$$

yielding a ranked set subject to aesthetic coherence & complementarity.

High-Level Goals (I)

- **Holistic curation:** move beyond single-item retrieval to a *cohesive* set
  - Focus can be on curation of non-similar bundles
  - Topic-oriented Recommendation,
  - Aesthetic Complementarity.
- **Context grounding:** leverage spatial cues (layout, lighting) and palette from images. Ground Concepts:
  - What does “Boho” Style exactly mean?
  - What does “Industrial” house furniture exactly mean?
- **Personalisation:** incorporate long-term preferences & budget constraints
  - Be aware of user long term features,
  - User’s behavior in similar sessions,
  - Other similar users behavior
- **Effort reduction:** “one-shot designer” experience vs. piecemeal searching
  - Reduce the time and effort of collection building by user.

High-Level Goals (II)

- **Brand/style awareness:** enforce on-brand style guides and user's declared themes
  - The model should be aware of high level queries
  - The model should be aware of brand to style relations
- **Consistency at scale:** ensure cross-item compatibility (materials, colors, sizes)
  - This is hard,
  - constructing cohesive bundles: a lot of edge cases,
- **Interactive refinement:** accept quick corrections
  - Design feature of this task
- **Ready-to-buy boards:** present ranked set with room mock-ups for fast decisions
  - Make life easier for the user.

CAL-RAG

an example of optimal layout generation

CAL-RAG: Retrieval-Augmented Multi-Agent Generation for Content-Aware Layout Design

Najmeh Forouzandehmehr, Reza Yousefi Maragheh, Sriram Kollipara, Kai Zhao, Topojoy Biswas, Evren Korpeoglu, Kannan Achan

Walmart Global Tech, Sunnyvale, California, USA

{najmeh.forouzandehmehr, reza.yousefimaragheh, sriram.kollipara, kai.zhao, topojoy.biswas, evren.korpeoglu, kannan.achan}@walmart.com

ABSTRACT

Automated content-aware layout generation—the task of arranging visual elements such as text, logos, and underlays on a background canvas—remains a fundamental yet underexplored problem in intelligent design systems. While recent advances in deep generative models and large language models (LLMs) have shown promise in structured content generation, most existing approaches lack grounding in contextual design exemplars and fall short in handling semantic alignment and visual coherence. In this work, we introduce CAL-RAG, a Retrieval-Augmented, Agentic framework for content-aware layout generation that integrates multimodal retrieval, large language models, and collaborative agentic reasoning. Our system retrieves relevant layout examples from a structured knowledge base and invokes an LLM-based layout recommender to propose structured element placements. A vision-language grader agent evaluates the layout based on visual metrics, and a feedback agent provides targeted refinements, enabling iterative improvement. We implement our framework using LangGraph and evaluate on the PKU PosterLayout dataset, a benchmark rich in semantic and structural variability. CAL-RAG achieves state-of-the-art performance across multiple layout metrics—including underlay effectiveness, element alignment, and overlap—substantially outperforming strong baselines such as LayoutPrompter. Our results demonstrate that combining retrieval augmentation with agentic multi-step reasoning provides a scalable, interpretable, and high-fidelity solution for automated layout generation.

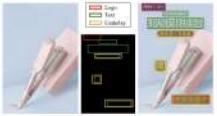


Figure 1: Example for content-aware layout generation: (a) Input canvas with background and content elements; (b) Generated layout based on visual and textual content awareness; (c) Final rendered presentation using the layout from (b).

offer limited generalization, particularly in diverse or semantically rich design contexts. The advent of deep generative models—such as GANs, VAEs, and transformers—has enabled more expressive layout synthesis by learning from annotated layout corpora [1, 2, 7, 9, 10]. However, these models are typically data-hungry, brittle to out-of-distribution inputs, and often struggle to incorporate visual-semantic alignment at inference time.

More recently, Large Language Models (LLMs) and Large Vision-Language Models (LVLMs) have demonstrated remarkable zero-

CAL-RAG

check out this paper

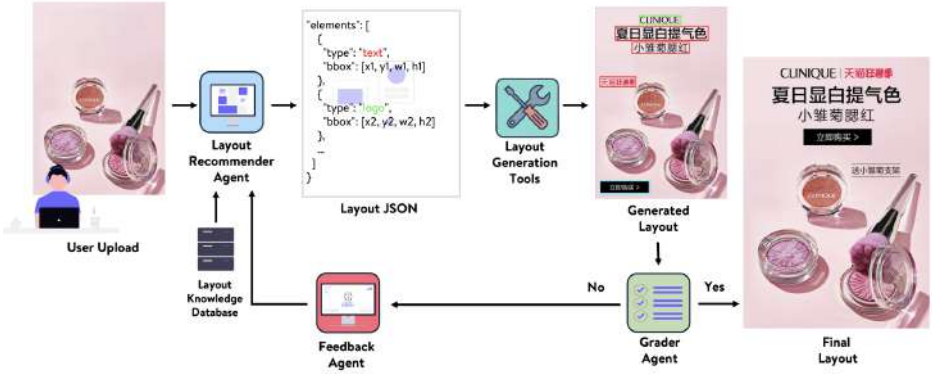
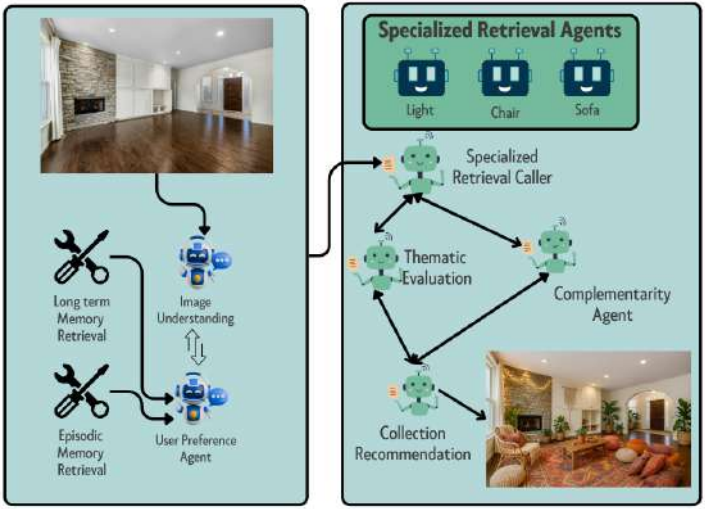


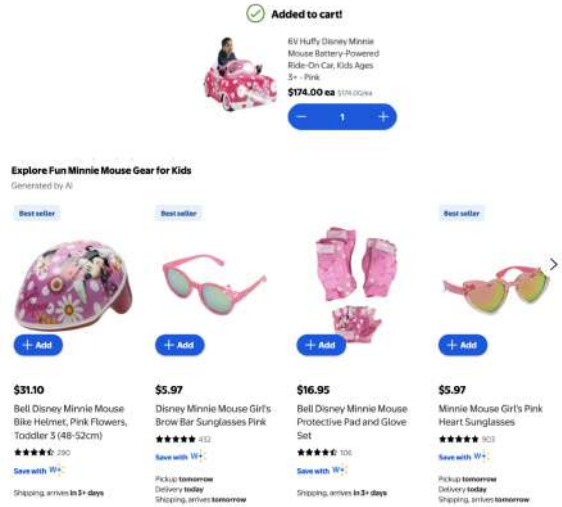
Figure 2: System Architecture Diagram of CAL-RAG.

Architecture: The Visual



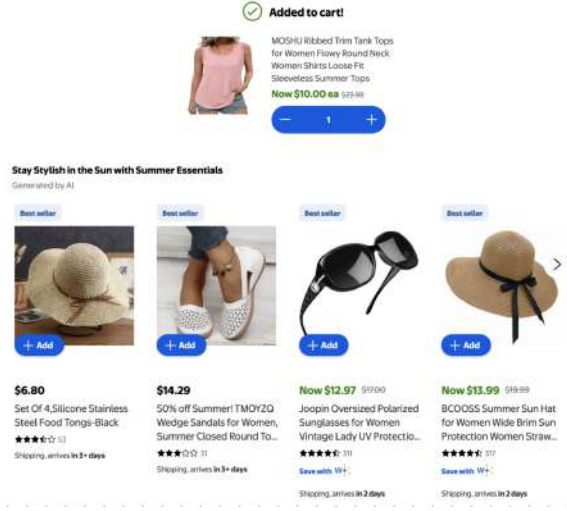
Example

Minnie Mouse



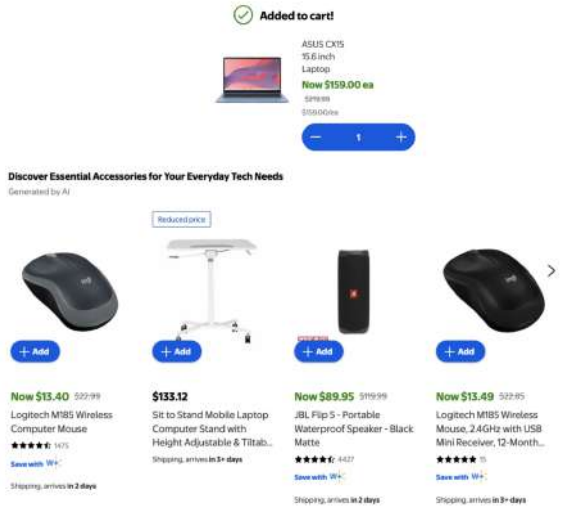
Example

Summer Dress



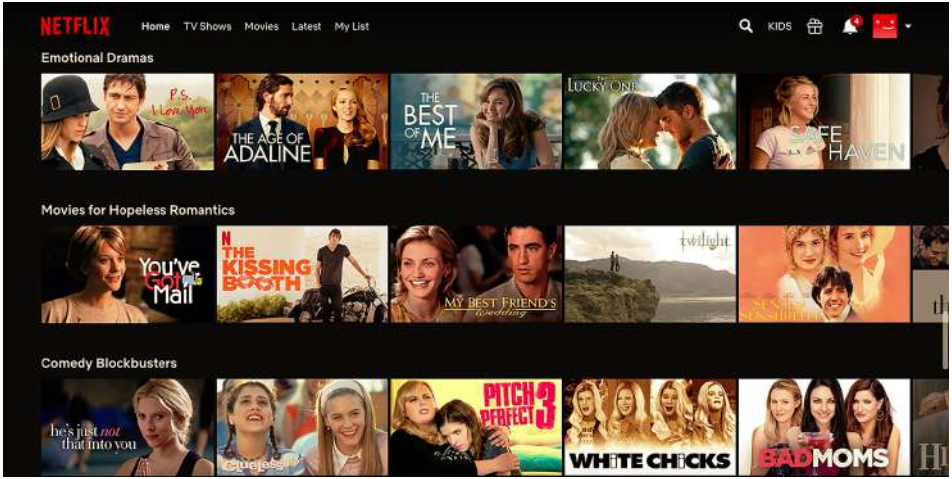
Example

Laptop for Everyday Use



Recommendation Explanation

Why are you recommending this?



Task Definition

Informal Definition

Recommendation Explanation is the process of generating intelligible and contextually relevant justifications for why particular items are recommended to a user.

Recommendation-Explanation maps items and user context to text:

$$\mathbf{r} \in \mathcal{S}^L, \quad \mathbf{u}_t = \left(\widehat{\mathcal{C}}_t^{\mathsf{SEM}}, \widehat{\mathcal{C}}_t^{\mathsf{EPI}}\right), \quad e_t = \Phi_\psi(\mathbf{r}, \mathbf{u}_t) \in \mathcal{E}_{\mathsf{text}}.$$

Constraints (consistency predicate)

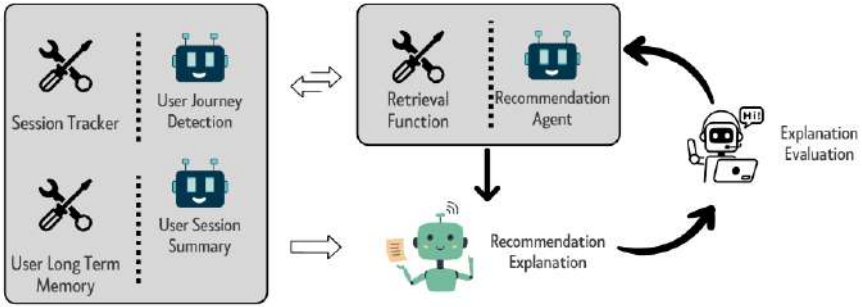
$$\mathrm{Consistency}(e_t) = \underbrace{\mathrm{Factual}(e_t : \mathbf{r}, \mathbf{u}_t)}_{\text{no hallucinations}} \wedge \underbrace{\mathcal{C}_{\mathcal{P}}(e_t) = 1}_{\text{brand/style compliance}}$$

Notes. $\widehat{\mathcal{C}}_t^{\mathsf{SEM}}, \widehat{\mathcal{C}}_t^{\mathsf{EPI}}$ obtained via memory retrieval $\mathcal{Q}$; $\mathcal{C}_{\mathcal{P}}$ is a policy checker (tone, vocabulary, legal).

High-Level Goal

- **Transparency:** expose the rationale behind recommended items in one concise narrative.
  - Explaining can help bridging the gap with what user wants and the recommendation,
  - makes the experience easier.
- **Trust & engagement:** increase user confidence and CTR with factual, user-aware explanations.
  - If the explanasion explains what user is looking for –¿ then the decision process is faster and conversion happens with more probability.
  - we have tested this and have evidance for it.
- **Brand consistency:** ensure on-tone language and regulatory alignment across languages & markets.
  - This is one of the most important issues, in large launches.
- **Low latency:** integrate into serving path with bounded overhead and revise-on-fail loop.
  - Cost benefit analysis, if doing so has a positive net impact!

Architecture



Formal Consistency Checks

Factuality (NLI/grounding):

$$\mathrm{Factual}(e_t : \mathbf{r}, \mathbf{u}_t) = \mathbf{1}\big[\forall\, \texttt{fact} \in e_t :\ \texttt{fact} \in \mathrm{Facts}(\mathbf{r}) \cup \mathrm{Facts}(\mathbf{u}_t)\big]$$

Brand/style compliance:

$$\mathcal{C}_{\mathcal{P}}(e_t) = \begin{cases} 1, & \text{all rules in policy } \mathcal{P} \text{ satisfied} \\ 0, & \text{otherwise} \end{cases}$$

Joint predicate: $\mathrm{Consistency}(e_t) = \mathrm{Factual}(e_t) \wedge \mathcal{C}_{\mathcal{P}}(e_t)$.

Optional score:

$$J(e_t) = \lambda_1 \,\mathrm{Readability}(e_t) + \lambda_2 \,\mathrm{Specificity}(e_t) - \lambda_3 \,\mathrm{Redundancy}(e_t),$$

maximized subject to $\mathrm{Consistency}(e_t) = 1$.

Any Questions?

Appendix-MCP

Appendix Recommendation Explanation

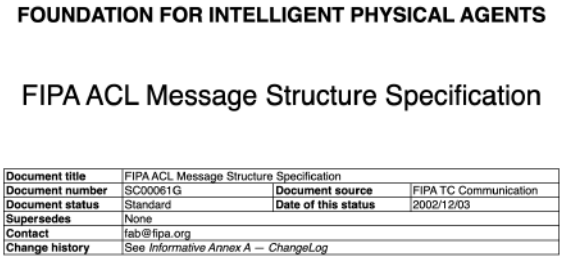
MCP in Multi-Agent RecSys: Why It Matters

- Agents must exchange **user context**, **item signals**, and **intermediate results** reliably.
- A robust **Model Context Protocol (MCP)** ensures message clarity, task negotiation, and state sync.
- Weak/adhoc protocols $\Rightarrow$ *bottlenecks*, *misinterpretations*, and *integration debt*.
- RecSys adds pressure:
  - **low latency**,
  - **high update rate**,
  - **clear semantics-in explanasion usecases**.

Protocol Standardization Problem & Goals

Snapshot: FIPA.org

- Goal: **plug-and-play** agents across teams/vendors via a shared syntax & semantics.A standard communication protocol.
- Legacy lesson (e.g., FIPA ACL): semantics help, but **complexity can hinder adoption**.



FOUNDATION FOR INTELLIGENT PHYSICAL AGENTS

FIPA ACL Message Structure Specification

Document title	FIPA ACL Message Structure Specification		
Document number	SC00061G	**Document source**	FIPA TC Communication
Document status	Standard	**Date of this status**	2002/12/03
Supersedes	None		
Contact	fab@fipa.org		
Change history	See *Informative Annex A — ChangeLog*		

2001!

Standardizing Agent Interoperability: The FIPA Approach

Stefan Poslad[1] and Patricia Charlton[2]

[1] Department of Electronic Engineering, Queen Mary, University of London,
Mile End Road, London, E1 4NS.
stefan.poslad@elec.qmul.ac.uk
[2] Centre de Recherche de Motorola-Paris
Espace Technologique – Commune de Saint Aubin
91193 Gif-sur-Yvette, France.
patricia.charlton@crm.mot.com

Abstract. A prolific number of different Multi-Agent Systems (MAS) and associated applications have been developed in numerous research institutes and industrial laboratories world-wide. Perhaps the most important barrier to MAS making a successful transition from this research environment towards widespread adoption for consumer products and businesses, is the lack of interoperability between heterogeneous MA Systems. In 1996, the **F**oundation for **I**ntelligent **P**hysical **A**gents (FIPA) was formed to provide a forum for developing specifications for agent systems. Since its formation, FIPA has increasingly focussed more on standardizing (multi-agent system) agent interoperability. As a result, it is often said that FIPA really stands for the **F**oundation for **I**nteroP**e**rable **A**gents. In this article, we discuss both technical and scientific issues in defining standards for interoperability between agents in

2007!

Specifying Protocols for Multi-Agent Systems Interaction

STEFAN POSLAD
Queen Mary, University of London

Multi-Agent-Systems or MAS represent a powerful distributed computing model, enabling agents to cooperate and complete with each other and to exchange both semantic content and a semantic context to more automatically and accurately interpret the content. Many types of individual agent and MAS models have been proposed since the mid-1980s, but the majority of these have led to single developer homogeneous MAS systems. For over a decade, the FIPA standards activity has worked to produce public MAS specifications, acting as a key enabler to support interoperability, open service interaction, and to support heterogeneous development. The main characteristics of the FIPA model for MAS and an analysis of design, design choices and features of the model is presented. In addition, a comparison of the FIPA model for system interoperability versus those of other standards bodies is presented, along with a discussion of the current status of FIPA and future directions.

Categories and Subject Descriptors: C.2.4 [**Computer-Communication Networks**]: Distributed Systems; I.3.6 [**Computer Graphics**]: Methodology and Techniques—*Standards*

General Terms: Standardization

Additional Key Words and Phrases: Multi-Agent systems, autonomy, semantics, social interaction, specifications, deployment

ACM Reference Format:

Poslad, S. 2007. Specifying protocols for multi-agent system interaction. ACM Trans. Autonom. Adapt. Syst. 2, 4, Article 15 (November 2007), 24 pages. DOI = 10.1145/1293731.1293735 http://doi.acm.org/ 10.1145/1293731.1293735

15

Protocol Standardization, The Design Trade-offs

- **Expressive vs. Lightweight**:
  - Privacy Concerns
  - Latency/Cost Concerns
  - Storage Concerns
- State Synchronization
  - Rapid changes induce **inconsistent memory** across agents.
  - Naive broadcast of all deltas $\Rightarrow$ network/CPU saturation.
  - Over-aggregation $\Rightarrow$ agents act on **stale** context.

$$\Delta : \ \mathcal{E}_t \rightarrow \mathcal{E}_{t+1} \quad \text{(environment update to be communicated coherently)}$$

Appendix-Challenges

Scaling Challenges

1. **Dynamic Agent Coordination:** Adaptive scheduling or routing (deciding which agents to invoke per request) could mitigate latency and cost surges; how to automate such coordination at scale is an ongoing research area.
2. **Cost-Aware Architectures:** Tools for balancing model accuracy with financial overhead remain nascent. Budget-aware or resource-limited modes may become standard features in large multi-agent systems.
3. **Multi-Lingual Model Consolidation:** Designing shared representations or truly multilingual models that preserve local performance while avoiding L-fold duplication is an active challenge, especially for low-resource languages.
4. **Observability and Debugging:** As agent count grows, tracing the contribution of each agent in a recommendation pipeline becomes non-trivial. New monitoring frameworks are needed to pinpoint bottlenecks, synchronization issues, or poor data hand-offs.

Appendix-User Simulation

Environment & Protocol

Environment

$\mathcal{E} = (\text{ProductDB},\ \text{PreferenceSampler},\ \text{LogStore})$

Protocol

$$A_{\text{rec}} \rightleftarrows A_{\text{user}} \to A_{\text{eval}} \to A_{\text{summ}} \to A_{\text{report}},$$

$\Gamma = \{\texttt{rec\_list}, \texttt{user\_action}, \texttt{log\_entry}, \texttt{eval\_event}, \texttt{session\_summary}, \texttt{final\_report}\}.$

$\mathbf{C}$ permits minimal paths to reduce chatter; logs are append-only.

End-to-End Flow

1. A_{rec}: produce $s_t^{(L)}$ for state x_t
2. A_{user}: act $a_t^{\text{user}} \sim \mathcal{M}_{\theta,\omega}(x_t, s_t^{(L)})$
3. A_{note}: log $(x_t, s_t^{(L)}, a_t^{\text{user}})$
4. A_{eval}: compute g_t and `eval_event`
5. A_{summ}: periodic $\mathcal{R}_{\text{sum}}$ summaries
6. A_{report}: aggregate $\Rightarrow$ KPIs, cohort analyses

Memory Requirements (I)

- A_{user}:
  - Ω^{SEM} (latent taste vector, price sensitivity, stable user features)
  - Ω^{STM} (current trajectory)
  - Ω^{EPI} (prior sessions / exposure effects)
  - Ω^{PROC} (navigation/click heuristics)
    - Encodes how the user proceeds in a session
- A_{eval}:
  - Raw text log + vector store for fast $\mathcal{Q}$ by item/action
  - Sliding-window stats (CTR, entropy) in STM

Tools & Data Access (II)

- `SearchAPI`, `ProductDB.query` for candidate context
- `PreferenceSampler` for (θ, ω) populations
- `LogStore.append`/`LogStore.scan` for $\mathcal{U}/\mathcal{Q}$
- Offline analytics: aggregation, stratified metrics, significance tests

Appendix-Contextual Recommendation

How to implememt?

Sample pseudo-algo

Goal: turn image + text + history into a coherent, feasible, on-budget bundle.

Key primitives in this module:

1. **Palette vector** $\mathbf{p}$: compact color/material signature of the image.
2. **Layout affordance** ℓ: spatial constraints (free space, obstacles, anchors).
3. **Compatibility kernel** $\kappa(\cdot, \cdot; \mathbf{p})$: pairwise harmony in a bundle.
4. **Total score** S_{tot}: personalized relevance + coherence − cost/penalties.

What is a *palette vector* $\mathbf{p}$?

Definition

A **palette vector** $\mathbf{p} \in \mathbb{R}^d$ is a compact, normalized descriptor of the dominant *colors & materials* present in the scene images $\mathbf{v} \in \mathcal{V}^K$:

$$\mathbf{p} = \big[p_1, \ldots, p_d\big], \quad p_i \geq 0, \quad \sum_{i=1}^{d} p_i = 1.$$

Instantiations (common in practice):

- **Hybrid** (color + material): concatenate color bins with material tags (e.g., rattan, linen, leather) and train a classifier.

Palette Vector: Role in the Pipeline

Usage: $\mathbf{p}$ conditions retrieval/ranking and the *compatibility function* $\kappa(\cdot, \cdot; \mathbf{p})$ so that selected items harmonize with the scene palette.

How to train: Maybe on co-purchase data! Or distill a large VLM to classifier.

Where it enters:

- *Category targeting*: prefer categories whose canonical palettes align with $\mathbf{p}$.
- *Per-item filtering*: discard items with palettes far from $\mathbf{p}$.
- *Bundle scoring*: $\kappa(s^i, s^j; \mathbf{p})$ rewards pairwise harmony under the observed scene palette.

What is a layout affordance ℓ?

Definition

Layout affordance ℓ encodes *spatial constraints* derived from the image including

- geometry: extract the floor plan
- free space dimensions,
- anchor elements in image like window, door, etc

to determine the test feasibility of placing a set $\widehat{\mathbf{y}}$ of items.

Feasibility predicate:

$\mathrm{Layout}(\widehat{\mathbf{y}}; \ell) = 1 \iff \exists$ non-overlapping placement satisfying extracted geometry, free space di

Compatibility Kernel $\kappa(\cdot, \cdot; \mathbf{p})$ (Examples)

$$\kappa(s^i, s^j; \mathbf{p}) = \underbrace{\mathrm{sim}\big(\mathrm{pal}(s^i), \mathrm{pal}(s^j) \mid \mathbf{p}\big)}_{\text{color harmony}} + \rho \underbrace{\mathrm{sim}_{\text{material}}(s^i, s^j)}_{\text{e.g., GloVe/CLIP on material tags}} + \tau \underbrace{\mathrm{sim}_{\text{style}}(s^i, s^j)}_{\text{style embedding cosine}} .$$

Bundle coherence:

$$S_{\text{comp}}(\widehat{\mathbf{y}}) = \frac{2}{L(L-1)} \sum_{i<j} \kappa(s^i, s^j; \mathbf{p}).$$

Notes:

- Color harmony condition by $\mathbf{p}$ to favor harmony around scene-dominant hues.
- Learned sim heads can be trained to match designer "goes-well-with" labels.

Explaining the *Total Score* S_{tot}

Decomposition

$$S_{\text{tot}}(\widehat{\mathbf{y}}) = \underbrace{\sum_{i=1}^{L} \alpha \, \text{Rel}(s^i \,|\, \mathbf{x}, \mathbf{u})}_{\text{personal relevance}} + \underbrace{\beta \, S_{\text{comp}}(\widehat{\mathbf{y}})}_{\text{coherence/complementarity}} - \underbrace{\gamma \, \text{Cost}(\widehat{\mathbf{y}})}_{\text{budget}} - \underbrace{\lambda \, \mathcal{P}_{\text{brand}}(\widehat{\mathbf{y}})}_{\text{style/brand penalty}} .$$

Hard vs. soft constraints:

$$\text{Hard: } \text{Compat}(\widehat{\mathbf{y}}; \mathbf{p}, \mathcal{G}) = 1; \quad \text{Soft: add penalties in } S_{\text{tot}}.$$

Choosing weights: $\alpha, \beta, \gamma, \lambda$ via grid/Bayes search or *learned* from designer judgments (pairwise ranking).

Optimizing S_{tot} under Constraints

Problem:

$$\widehat{\mathbf{y}}^{\star} = \arg\max_{\widehat{\mathbf{y}}} \; S_{\text{tot}}(\widehat{\mathbf{y}}) \quad \text{s.t. Budget} \le B, \; \text{Layout}(\widehat{\mathbf{y}}; \ell) = 1.$$

Solvers (latency/quality trade-offs):

- *Greedy w/ lookahead*: fast, good for serving.
- *ILP/MIP relaxations*: exact/near-exact, best offline.

Caching: precompute per-item relevance and per-pair κ for top categories to accelerate online assembly.

Another Formal Definition

$$\mathbf{x} \in \mathcal{T}, \quad \mathbf{v} \in \mathcal{V}^K, \quad \mathbf{u} \in \mathcal{N}, \quad \widehat{\mathbf{y}} = \langle s^1, \ldots, s^L \rangle, \; s^i \in \mathcal{S}.$$

$$\text{Objective:} \quad \widehat{\mathbf{y}}^\star = \arg\max_{\widehat{\mathbf{y}}} \sum_{i=1}^{L} \mathrm{Rel}\big(s^i \mid \mathbf{x}, \mathbf{v}, \mathbf{u}\big)$$

Subject to design constraints:

$$\mathrm{Compatability}(\widehat{\mathbf{y}};\, \mathbf{p}, \mathcal{G}) = 1, \qquad \mathrm{Budget}(\widehat{\mathbf{y}}) \le B, \qquad \mathrm{Layout}(\widehat{\mathbf{y}};\, \ell) = \text{feasible},$$

where $\mathbf{p}$ is a palette vector from vision, $\mathcal{G}$ style rules, and ℓ layout affordances.

Another Formal Definition

History retrieval: $\widehat{\mathcal{C}}_{\mathsf{SEM}} = \mathcal{Q}\big(\Omega^{\mathsf{SEM}},\, \tau(\mathbf{x})\big)$

Category selection: $C = \mathrm{Cat}(\mathbf{x}, \mathbf{p}, \widehat{\mathcal{C}}_{\mathsf{SEM}})$

Per-category retrieval: $\forall c \in C,\ \mathcal{F}_{\mathsf{search}}(c, \mathbf{p}, \mathbf{u}) \to \mathcal{S}_c$

Bundle assembly: $\widehat{\mathbf{y}} = \bigcup_{c \in C} \mathrm{TopK}(\mathcal{S}_c)$

Semantic & complementarity gates: $\mathrm{SemChk}(\widehat{\mathbf{y}}) = 1,\ \mathrm{CompChk}(\widehat{\mathbf{y}}) = 1.$

End-to-End Algorithm (Pseudo-Code)

1. $(\mathbf{x}, \mathbf{v}) \leftarrow$ user input; $\widehat{\mathcal{C}}_{\text{SEM}} \leftarrow \mathcal{Q}(\Omega^{\text{SEM}}, \tau(\mathbf{x}))$.
2. $\mathbf{p} \leftarrow f_{\text{img}}(\mathbf{v})$; $\ell \leftarrow$ affordance extractor$(\mathbf{v})$.
3. $C \leftarrow \text{Cat}(\mathbf{x}, \mathbf{p}, \widehat{\mathcal{C}}_{\text{SEM}})$.
4. For $c \in C$: $\mathcal{S}_c \leftarrow \mathcal{F}_{\text{search}}(c, \mathbf{p}, \mathbf{u})$; keep TopK per category.
5. Assemble $\widehat{\mathbf{y}}$ by maximizing S_{tot} under Budget, Layout.
6. Run SemChk, CompChk; if fail, revise C/constraints and re-solve.
7. Render mock-up; return bundle and visualization to A_{chat}.

System Architecture: Agents

$\mathrm{MAS}_{\mathrm{mm}} = (\mathcal{A}, \mathcal{E}, \Pi)$

$$\mathcal{A} = \{A_{\mathsf{chat}}, A_{\mathsf{image}}, A_{\mathsf{history}}, A_{\mathsf{cat}}, A_{\mathsf{caller}}, A_{\mathsf{semChk}}, A_{\mathsf{compChk}}, A_{\mathsf{collect}}\}$$

- A_{image}: palette & layout from images $(f_{\mathsf{img}}, \mathcal{F}_{\mathsf{layout}})$
- A_{history}: $\mathcal{Q}$ over Ω^{SEM} for stable tastes/budget
- A_{caller}: combines signals; calls specialized micro-MAS,
- $A_{\mathsf{semChk}}/A_{\mathsf{compChk}}$: rule compliance & harmony gates
- A_{collect}: rank & render; return set to A_{chat}

Message Flow & Protocol

$$
\begin{aligned}
A_{\text{chat}} &\to A_{\text{image}}\ (\mathbf{v}) \ \ \& \ \ A_{\text{history}}\ (\mathbf{x}) \\
&\to A_{\text{caller}}\ (\mathbf{x}, \mathbf{p}, \widehat{\mathcal{C}}_{\text{SEM}}) \\
&\to \{\text{micro-MAS}_c\}_{c \in C} \to A_{\text{semChk}} \to A_{\text{compChk}} \to A_{\text{collect}} \to A_{\text{chat}}
\end{aligned}
$$

Autonomy can happen based in Agent calling to adjust the retrieval process per query.

Memory Requirements (I)

- **Short-term (STM):** live prompt, extracted palette $\mathbf{p}$, current candidate sets
- **Semantic (SEM):** durable preferences (colors, materials, budget caps), style affinities
- **Episodic (EPI):** past styling sessions (accepted/rejected bundles, returns)
- **Procedural (PROC):** routing templates, per-category query schemes, brand/style rules

Tools & Data Access (II)

- `VisionEncoder` f_{img}, `LayoutRenderer` $\mathcal{F}_{\text{render}}$
- `VectorSearch` over `ProductDB` for semantic retrieval
- `UniversalSearch`/`PriceAPI` for stock/price validation
- `CompatEngine` for pairwise/multi-item harmony checks

Benefits

- **Designer-quality curation:** coherent, theme-consistent sets in one step
  - Now, near designer quality!
- **Lower cognitive load:** fewer iterations vs. item-by-item browsing
  - Easy user experience
- **Personal, grounded:** vision-aware + history-aware = higher acceptance
  - This was not possible before!
  - **Error containment:** semantic/complementarity gates reduce mismatch & drift
- **Modular extensibility:** add/replace micro-MAS for new styles or categories
  - Easier implementation.

Appendix Recommendation Explanation

System Architecture (Agents)

$$\mathrm{MAS}_{\mathsf{expl}} = (\mathcal{A}, \mathcal{E}, \Pi), \quad \mathcal{A} = \{A_{\mathsf{journey}}, A_{\mathsf{session}}, A_{\mathsf{rec}}, A_{\mathsf{expl}}, A_{\mathsf{eval}}\}.$$

- A_{session} (User Session Summary): $\mathcal{Q}$ over $\Omega^{\mathsf{SEM}} \cup \Omega^{\mathsf{EPI}}$ to form $\mathbf{u}_t$.
- A_{journey} (Journey Detector): tags live intent from clickstream (e.g., "holiday décor upgrade").
- A_{rec} (Recommender): returns $\mathbf{r} \in \mathcal{S}^L$ given $(\mathbf{u}_t, \text{intent})$.
- A_{expl} (Explanation Writer): $e_t = \Phi_\psi(\mathbf{r}, \mathbf{u}_t)$.
- A_{eval} (Explanation Evaluator): verifies factuality & brand; issues `revise` if needed.

Environment $\mathcal{E}$: ProductDB, UserProfileDB, BrandPolicy, LogStore.

Communication & Flow

Allowed edges ($\mathbf{C}_{ij}=1$):

$$A_{\text{session}} \to A_{\text{rec}}, \quad A_{\text{journey}} \to A_{\text{rec}}, \quad A_{\text{rec}} \to A_{\text{expl}} \to A_{\text{eval}} \to A_{\text{rec}}.$$

Interaction loop:

1. A_{session}: $\mathbf{u}_t \leftarrow \mathcal{Q}(\Omega^{\text{SEM}} \cup \Omega^{\text{EPI}}, \tau)$
2. A_{journey}: infer intent from clicks.
3. A_{rec}: produce $\mathbf{r}$ conditioned on $(\mathbf{u}_t, \text{intent})$.
4. A_{expl}: generate e_t.
5. A_{eval}: check Factual & $\mathcal{C}_{\mathcal{P}}$; if fail $\Rightarrow$ `revise` $\rightarrow$ (3–4).

Memory & Tools

Memory (by agent):

- A_{session}: Ω^{SEM} (stable prefs), Ω^{EPI} (recent sessions), Ω^{STM} (current turns).
- A_{expl}: Ω^{PROC} (brand tone exemplars, banned phrases, templates).
- A_{eval}: cached fact table for $\mathbf{r}$; policy store $\mathcal{P}$.

Tools $\mathcal{F}$:

- `UserData.fetch`, `ProductDB.lookup` (explanation and item grounding).
- `NLI.verify` (trace facts).
- `PolicyCheck` (brand/legal), `StyleTransfer` (tone repair).

Benefits & Discussion

- **Trust & CTR uplift:** clear, user-aware rationale improves acceptance.
- **Brand safety at scale:** evaluator veto enforces $\mathcal{P}$ with low integration cost.
- **Personal continuity:** explanations reuse the same $\Omega^{\text{SEM}}/\Omega^{\text{EPI}}$ driving ranking.
- **Modularity:** new style guides or fairness rules added via Ω^{PROC} and A_{eval} prompts.