

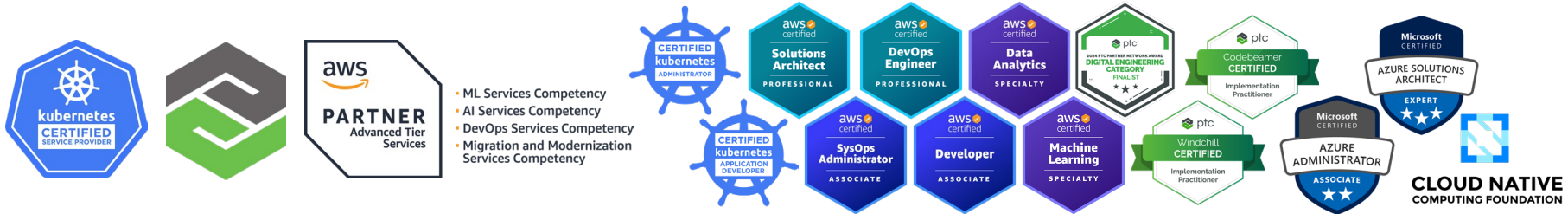
Self-Improving GenAI Without Fine -Tuning



About Source Allies



- IT Consultancy
 - Industries: Insurance, AgriTech, Energy, FinTech
 - Engineers (Data, AI/ML/LLM, Software, Cloud, DevOps)
- Builders Who Teach
- Simplest Architecture



About Our Speakers



Matt Vincent

- Founded Source Allies in 2002
- One of several teammates who started our AI practice 7+ years ago
- Fun fact: Dad led neural networks research group



Ben McHone

- Open source contributor: Phoenix, LangChain, DSPy
- Builds production AI systems (agents → multi-agent)
- Turns vague ideas into deterministic-ish systems



Agenda



ط ?Ä é ©îJÄP~~t~~ Xä .î©öXä î Tçã þ XIiã þ .A>Xêçìö=

ط F | ©Ä `ãXö§ãÄz ìÄÜC!

ط F | JöNã ß XTç ÄîöXT!!

ط &Xä ç

= §Xi f çãî ß XNçä XT ö| ìç§z| ç§ö= 😊



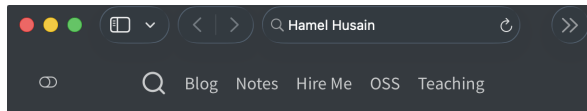


Most GenAI Projects Fail Due to Lack of Trust, Not Capability

Question: What are you doing to improve trust?



5 XîşîXä Xäö



Your AI Product Needs Evals

LLMS EVALS

How to construct domain-specific LLM evaluation systems.

AUTHOR
Hamel Husain

PUBLISHED
March 29, 2024

Motivation

I started working with language models five years ago when I led the team that created [CodeSearchNet](#), a precursor to GitHub CoPilot. Since then, I've seen many successful and unsuccessful approaches to building LLM products. I've found that unsuccessful products almost always share a common root cause: **a failure to create robust evaluation systems.**

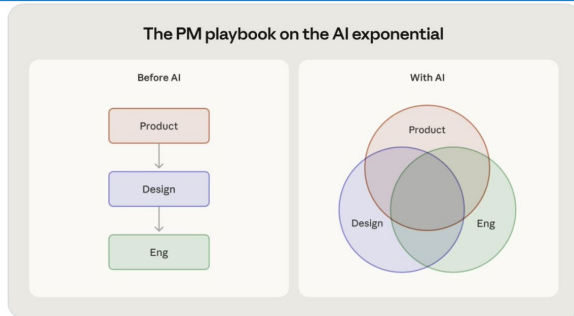


Aparna Dhinakaran
@aparnadhinak

The Head of Product at Claude Code ran an eval to test Claude Code every time a new model came out.

That's it. That's modern AI product management.

If you're not investing in evals, you're not really shipping—you're guessing.



Jason Lopatecki
@jason_lopatecki

The teams escaping the pilot trap are treating the pilot itself as a measurement exercise rather than a proof-of-concept. That means going into the pilot with defined success metrics, instrumented outputs, and a monitoring setup that can tell you whether those metrics are moving.



ashu garg
@ashugarg · Mar 31

Enterprise AI has a pilot problem.

The gap between a promising pilot and concrete change in how work gets done was one of the central themes of our recent CEO and CIO dinners. We heard from @djpersia of @databricks, Rajat Taneja of @Visa, and CIOs from ...

What it takes
to win
enterprise AI
deals

B2BaCEO
with Ashu Garg



MIT: State of AI in Business



The Primary Barrier with GenAI Systems Isn't Model Quality
It's that our GenAI Systems Don't Learn

- Don't retain feedback
- Don't improve over time
- Don't learn from how we work

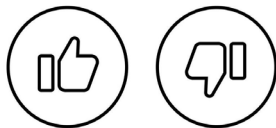
Question: What are you doing today to mitigate this problem?



MIT: State of AI in Business



The Primary Barrier with GenAI Systems Isn't Model Quality
It's that our GenAI Systems Don't Learn



Thanks for your rating

What went wrong?

Provide additional feedback

- ☐ Not factually correct
- ☐ Did what I asked, but response was inaccurate
- ☐ Not the format or style I wanted
- ☐ Offensive / unsafe
- ☐ Personal information exposed
- ☐ Legal issue

Data you share is used as described in the "Why We Process Service Data" section of the [Google Cloud Privacy Notice](#). Google doesn't use your data to train models.

Cancel

Submit

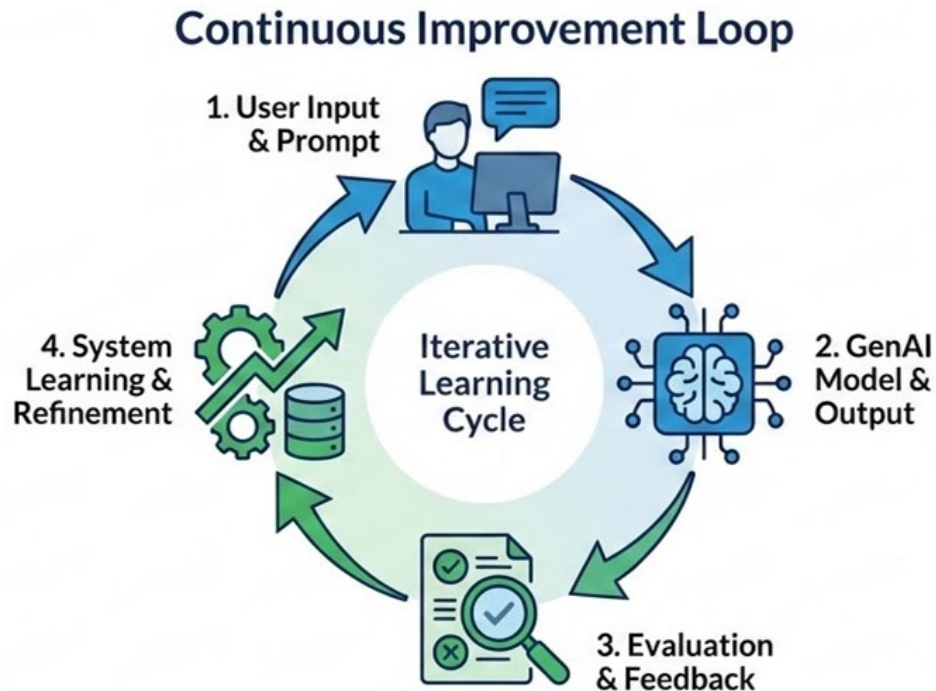
This feedback improves Google's model, not yours.





Generic GenAI tools shine for individual brainstorming, but stall in enterprise settings because they lack iterative learning.

Path forward involves moving beyond prompt-and-forget LLMs toward systems with continuous improvement loops.





RL/Fine-Tuning Can Be a Trap

- Requires Infrastructure & AIOps *
 - Lose "shared responsibility" comfort**
- Triggers heavier governance vs. consumable API

* Some fine-tuning APIs exist to create managed, no weight access models.

** Where cloud vendor takes primary resp for model's core behavior, safety, and updates.



5 çîö225 *JÄ§îXî ìX.ãîöî§N̈f çã *JÄ§îXî



225 î Tç ß | Jö©ç§ öẌ ö | Xä → çîöZÄ§îXî JìXJNö§J' ©
M̈T Äîöî§N̈f çãîP̈çöJ M̈T ä çTẌ→

5 Jã©êîçä êöZÄ§îXî JìXJNö§J' ©Üäçß' XTzXzJêî
ß | XîXä ÄîÄz êîÄN̈ä Xî ö | Jöß XîXäẌẌî îêXN̈äXT→

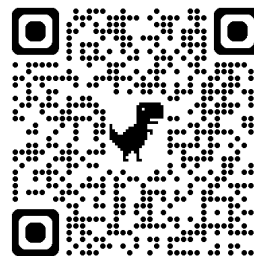
⚠ ?ä J' êîçä êöN̈ JãzXî ù ' JìzXç§öê§öTÄ XîXäNXî→



.ãîö§Ñ çãî &çãþ\$çä X.ã ö X225 #ç®



A| XiXÄãç êiXTX ãXT ê J©MçÜZçì | çß öç §îX225 î
X| XN¶¶X©Pisç ß XJìXNç XN¶¶X©TÄNç¶¶XiÄz ÄÄ îXJ f ä X→



(®Äf ãz Aç ç î &ç ã þ ? ç ¶X2XI ã Äz



F , A.A?8 2E(?

F , A? ?A.22 5 .?? .6 +

RL / Fine-Tuning

- Deep model optimization, proven at scale
- Expensive, slow to change, triggers governance overhead

RAG

- Better grounding with current, domain-specific facts
- Improves data, not reasoning strategy

5 Xä ç î©

đ \$çãfã\$Ä©JNçîî îXiîÄãîPäç îXL
êiç¶ÄÄz NçãöX©

- Stores history, doesn't extract lessons from it

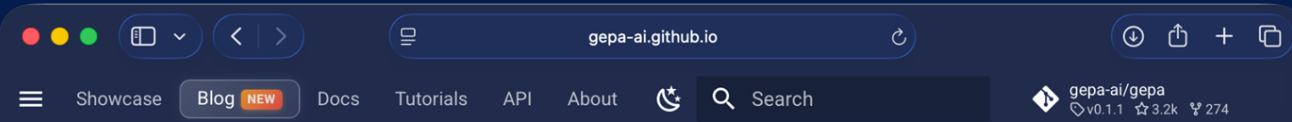
Hand-crafted prompts, skills, etc

- Fast to iterate, no infrastructure needed
- Doesn't scale; humans miss recurring failure patterns





What If We Could...



optimize_anything: A Universal API for Optimizing any Text Parameter

February 18, 2026 · Lakshya A Agrawal*, Donghyun Lee*, Wenjie Ma, Karim Elmaaroufi, Shangyin Tan, Sanjit A. Seshia, Koushik Sen, Dan Klein, Ion Stoica, Joseph E. Gonzalez, Omar Khattab, Alexandros G. Dimakis, Matei Zaharia

* Equal Contribution

Today we are introducing `optimize_anything`, a declarative API that optimizes any artifact representable as text (e.g., code, prompts, agent architectures, vector graphics, configurations). It extends GEPA (Genetic-Pareto, our state-of-the-art LLM prompt optimizer) far beyond prompts. You declare what to optimize and how to measure it; the system handles the search. Testing it across several domains, we find `optimize_anything` consistently matches or outperforms domain-specific tools, including some purpose-built for each task. With one API, you can:

- [create agent skills achieving near-perfect Claude Code task completion 47% faster,](#)
- [optimize cloud scheduling policies that cut costs by 40%, beating expert heuristics,](#)
- [find detailed system prompts to boost GPT's math reasoning accuracy,](#)
- [discover bespoke agent harnesses that nearly triple Gemini Flash's ARC-AGI accuracy,](#)
- [write custom solvers to match and exceed Optuna in blackbox mathematical optimization,](#)

Table of contents

The optimize_anything API

The Simplest Form

One Interface, Three Optimization Modes

Let's Take It for a Spin

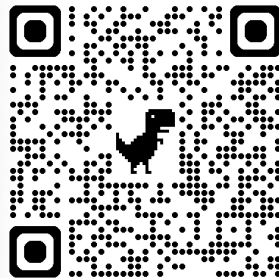
How It Works

Actionable Side Information (ASI)

Pareto-Efficient Search

Results

1. Optimize Agent Skills: Near-Perfect Claude Code Accuracy, 47% Faster
2. Discover Cloud Algorithms That Cut Costs up to 40%
3. Nearly Triple Gemini-Flash's ARC-AGI Accuracy via Agent Architecture Evolution
4. Improve GPT's Math Accuracy via Prompt Optimization (AIME)
5. Accelerate PyTorch with Custom CUDA Kernels
6. Outperform AlphaEvolve's

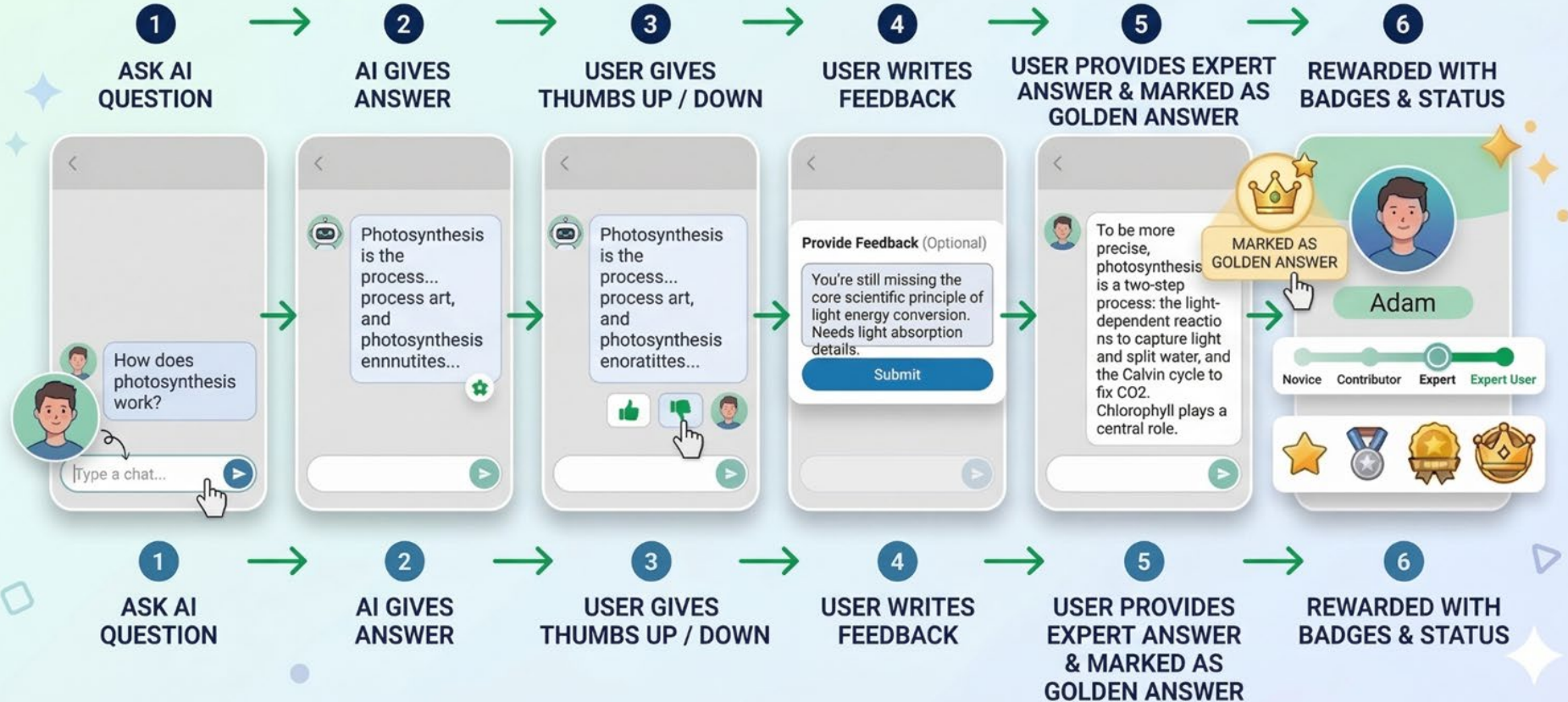




?êXNÄN(®äê'X



NORMAL FLOW IN A CHAT APP: AI INTERACTION & REWARDS





MICHELIN STAR

Coveted by many chefs but bestowed upon only to an excellent few.
Getting a star (or three) could change the fate of a restaurant.



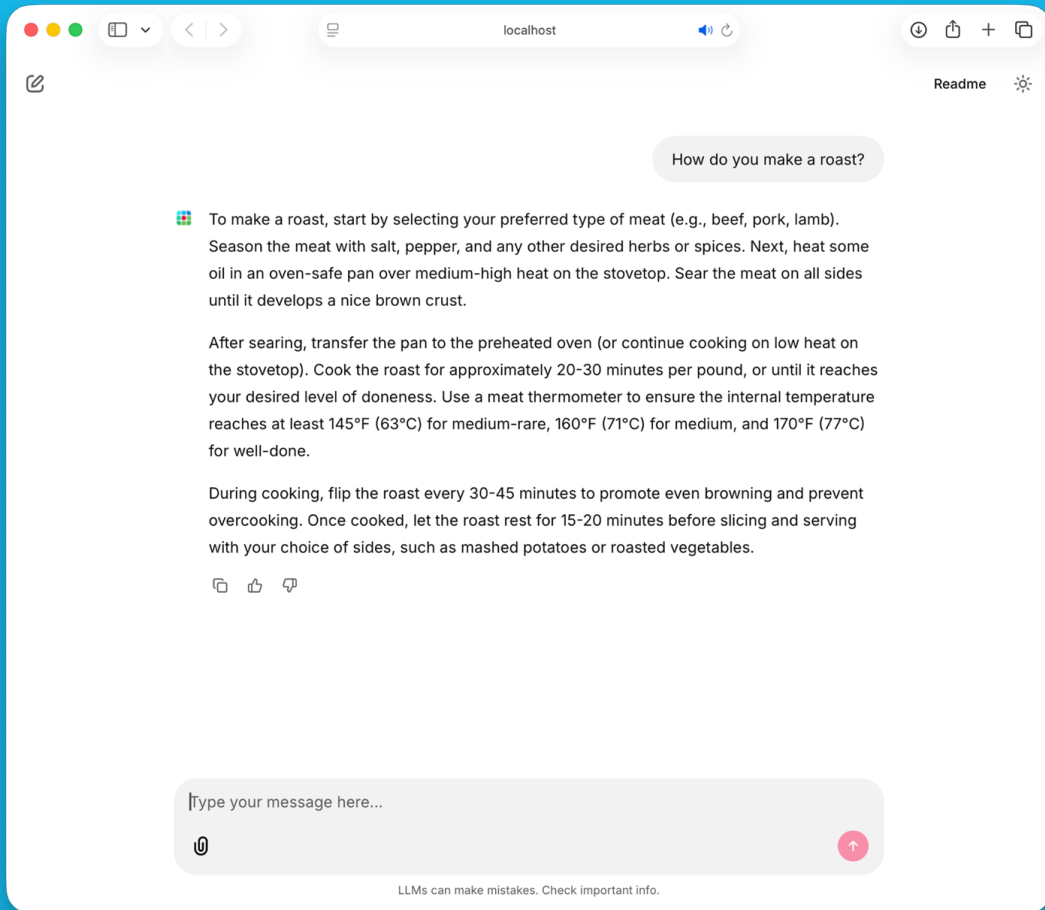
High quality
cooking,
worth a stop

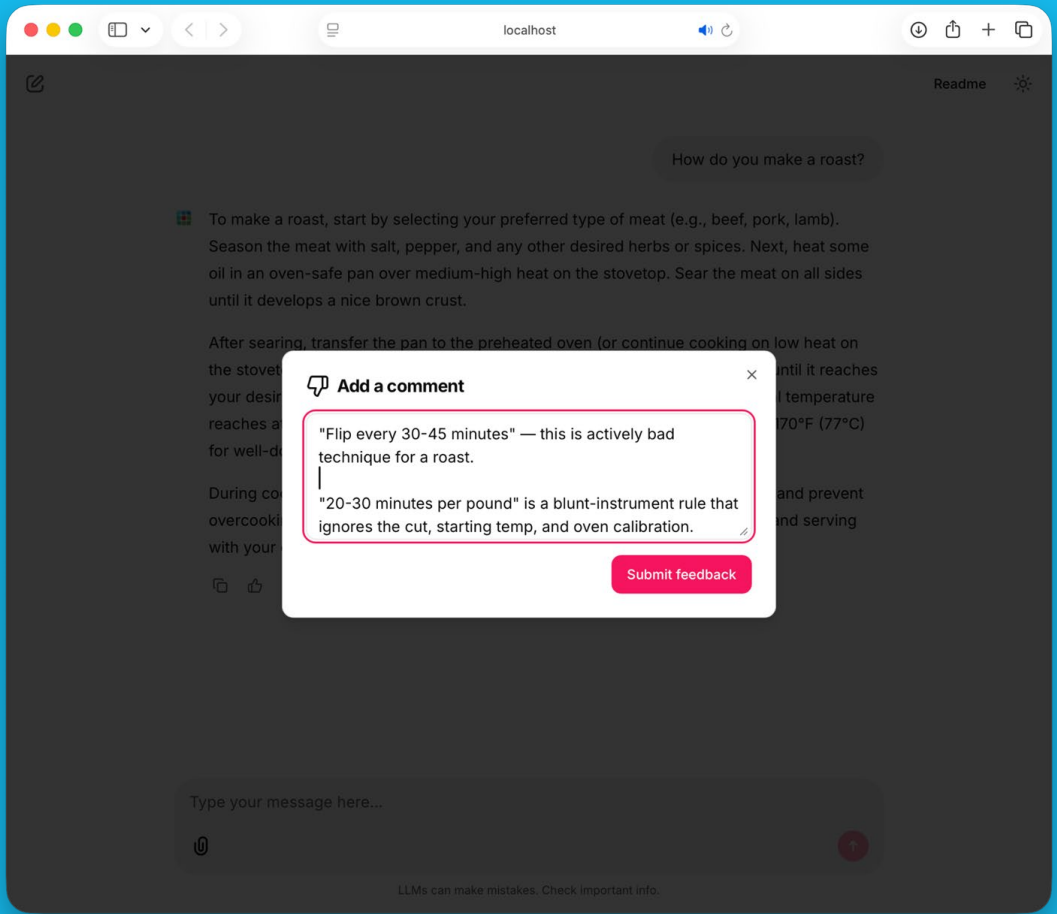


Excellent cooking,
worth a detour



Exceptional
cuisine, worth a
special journey







localhost

Readme

How do you make a roast?

To make a roast, start by selecting your preferred type of meat (e.g., beef, pork, lamb). Season the meat with salt, pepper, and any other desired herbs or spices. Next, heat some oil in an oven-safe pan over medium-high heat on the stovetop. Sear the meat on all sides until it develops a nice brown crust.

After searing, transfer the pan to the preheated oven (or continue cooking on low heat on the stovetop). Cook the roast for approximately 20-30 minutes per pound, or until it reaches your desired level of doneness. Use a meat thermometer to ensure the internal temperature reaches at least 145°F (63°C) for medium-rare, 160°F (71°C) for medium, and 170°F (77°C) for well-done.

During cooking, flip the roast every 30-45 minutes to promote even browning and prevent overcooking. Once cooked, let the roast rest for 15-20 minutes before slicing and serving with your choice of sides, such as mashed potatoes or roasted vegetables.

In a heavy roasting pan or cast-iron, lay down a mirepoix — roughly chopped onion, carrot, celery, and a whole head of garlic split horizontally. Add a few sprigs of thyme and rosemary. This is your roasting bed and the foundation of your jus. The vegetables should never be an afterthought.

5. The Sear

Get your pan screaming hot — a thin film of neutral oil with a high smoke point (grapeseed, not olive). Sear every surface of the roast until you achieve a deep mahogany Maillard crust. This is flavor, not just color. Do not rush it, and do not crowd the pan. A proper sear takes 8-12 minutes total for a large joint.

LLMs can make mistakes. Check important info.

How Do We Leverage This Data?



F XNã TXä çãîöJöXö Jö . Nã ìXvXNöçã Ä
çßã ZÄſiXi ß | Xä zÄXä XÖä ê Xi →

ℓ → + Jö Xi (Öä ê Xi ↑ \$ ç'' XNö Ç ç ſi öXi ä H
öçê • ÄîöJãNXi çZêççì . êXiZçia JãNX
ö Ä ß XXÜ →

□ → > Xi § Xi ö.ä êiç¶Xä Xäö†; iXi Xäöö X
ZÄXT ÄöXiJNſ çã öç ö X . JãTJîÜÄöç
zXäXiJöXJ iX äXT êiçä êöö Jöß çſ T
©ÄXT Mſ Xi iXi ſ ö →

▪ → .öXiJöX† > Xê Xiöö XêiçNXî Zçì XJN
XÖä ê X →

▫ → EJ ÄJöX† AXöö XäXB' ©zXäXiJöXT
êiçä êöJãT iX äXZſiö Xi Jî äXNXîJî© →

I used this prompt:

<prompt>

and this input:

<input>

That gave me this output:

<output>

But the output wasn't good for these reasons:

<reasons>

Give me a new <prompt> that helps fix this issue

Here is an updated prompt:

<new prompt>





Starting Prompt

Answer questions accurately and thoroughly, optionally using provided context.

Explain the reasoning and mechanisms behind your answer, not just surface-level facts.

When relevant, include specific details, caveats, and practical implications.



Optimized Prompt

Answer culinary and food science questions accurately and thoroughly, using precise technical terminology. Responses should be concise but complete – include every critical detail, but do not pad with unnecessary elaboration.

Key principles for responses:

Identify the exact scientific/technical nature of ingredients and techniques (e.g., specific protein types, chemical compounds, physical processes)

Explain the underlying mechanisms – why things happen at a molecular or structural level, not just what happens

Include all critical properties and caveats, especially those that distinguish similar concepts (e.g., elastin vs. collagen, hygroscopic vs. non-hygroscopic)

Domain-specific knowledge to apply:

CONNECTIVE TISSUE:

Collagen: structural protein in connective tissue that gelatinizes (converts to gelatin) with prolonged moist heat (160-180°F+), tenderizing tough cuts and creating silky body in braises and stocks

Elastin (silverskin): the pearlescent/silvery connective tissue sheath. Does NOT break down with heat – it shrinks, curls, and toughens, distorting the shape of the meat and remaining inedible. Must be physically removed before cooking.

FATS & OILS:





How do you make a roast?

To make a roast, start by selecting your preferred type of meat (e.g., beef, pork, lamb). Season the meat with salt, pepper, and any other desired herbs or spices. Next, heat some oil in an oven-safe pan over medium-high heat on the stovetop. Sear the meat on all sides until it develops a nice brown crust.

After searing, transfer the pan to the preheated oven (or continue cooking on low heat on the stovetop). Cook the roast for approximately 20-30 minutes per pound, or until it reaches your desired level of doneness. Use a meat thermometer to ensure the internal temperature reaches at least 145°F (63°C) for medium-rare, 160°F (71°C) for medium, and 170°F (77°C) for well-done.

During cooking, flip the roast every 30-45 minutes to promote even browning and prevent overcooking. Once cooked, let the roast rest for 15-20 minutes before slicing and serving with your choice of sides, such as mashed potatoes or roasted vegetables.



How do you make a roast?

How to Make a Classic Roasted Meat (Beef, Pork, or Lamb)

Step	What to Do	Why it Matters
1. Choose the Right Cut	• Beef: Chuck roast, rib roast, or prime rib (2–4 lb). • Pork: Shoulder (pork butt) or loin (tenderloin). • Lamb: Leg or shoulder (3–5 lb).	Larger, fattier cuts are best for long, slow roasts; leaner cuts need higher heat and shorter time.
2. Prepare the Meat	• Pat the roast dry with paper towels. • Trim excess fat if desired, but leave a fat cap for flavor. • Let it sit at room temperature 30–60 min before cooking.	Dry surface = better browning; room temp reduces cooking time.
3. Season Generously	• Basic rub: salt (1tsp per lb), black pepper (½tsp per lb). • Add herbs: rosemary, thyme, oregano (1–2tsp each). • Optional: garlic powder or minced garlic, onion powder, paprika. • For a "bistro-style" rub: mix with olive oil to make a paste. • Massage rub into all sides of the roast.	Seasoning penetrates the meat; herbs and aromatics infuse flavor.
4. Sear (Optional but Recommended)	• Heat a large oven-proof skillet or roasting pan on high heat; add a splash of oil. • Sear the roast 2–3 min per side until a deep brown crust forms.	Sear creates Maillard reaction, locks in juices, and gives a savory crust.
5. Roast in the Oven	• Preheat oven to 450°F (232°C) for a quick crust, then lower to 325–350°F (165–175°C). • Place roast on a rack in a shallow pan; add aromatics (2–3 sliced onions, 2–3 cloves	Lowering the temperature after searing allows even cooking without



github.com

x1xh1o1 / system-prompts-and-models-of-ai-tools

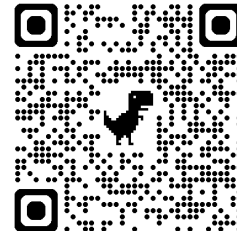
Code Issues 90 Pull requests 53 Agents Discussions Actions Projects Security and quality Insights

main system-prompts-and-models-of-ai-tools / Anthropic / Claude Sonnet 4.6.txt

x1xh1o1 Rename claude-sonnet-4.6.txt to Claude Sonnet 4.6.txt 54d15ea · last month History

Code Blame

```
1 <claude_behavior>
2
3 <product_information>
4 Here is some information about Claude and Anthropic's products in case the person asks:
5
6 This iteration of Claude is Claude Sonnet 4.6 from the Claude 4.6 model family. The Claude 4.6 family currently consists of Claude
7
8 If the person asks, Claude can tell them about the following products which allow them to access Claude. Claude is accessible via
9
10 Claude is accessible via an API and developer platform. The most recent Claude models are Claude Opus 4.6, Claude Sonnet 4.6, and
11
12 Claude does not know other details about Anthropic's products, as these may have changed since this prompt was last edited. If as
13
14 When relevant, Claude can provide guidance on effective prompting techniques for getting Claude to be most helpful. This includes
15
16 Claude has settings and features the person can use to customize their experience. Claude can inform the person of these settings
17
18 Anthropic doesn't display ads in its products nor does it let advertisers pay to have Claude promote their products or services i
19 </product_information>
20
21 <refusal_handling>
22 Claude can discuss virtually any topic factually and objectively.
23
24 Claude cares deeply about child safety and is cautious about content involving minors, including creative or educational content
25
26 Claude cares about safety and does not provide information that could be used to create harmful substances or weapons, with extra
27
28 Claude does not write or explain or work on malicious code, including malware, vulnerability exploits, spoof websites, ransomware
29
30 Claude is happy to write creative content involving fictional characters, but avoids writing content involving real named public
```



Introducing GEPA



Genetic Pareto, a framework can optimize text systems, like system prompts, using AI and **written feedback** .

"Evolutionary search works now because LLMs can perform the mutation intelligently"

Reflective prompt evolution can **outperform reinforcement learning**

Research: DSPy GEPA, Quality Diversity Optimization



Introducing GEPA



GEPA Leverages LLMs to Understand:

- **Principles:** Extracts the underlying general rules or gaps that cause the failures in the first place.
- **Practices:** Discovers unspoken rules or “tribal knowledge” within our task.
- **Strategies:** Is able to “reason” ahead of time by reflecting on known failures or undesirable behavior that we are able to define.
- **Techniques:** The optimization result provides us specific patterns or analogies that resonate better with the language model.



The Magic - Reflective Prompt Evolution



Traditional Optimization

Score: 0.7

Numeric feedback only

GEPA

```
## Feedback
Correctly included: Unlike collagen, does not
break down or gelatinize with heat.
Correctly included: Remains tough/inedible.
Missing Information: Silverskin is elastin,
type of connective tissue.
```

Natural language guides mutations

A| Xä çTX ìXJîçãî JMçşöÄî çβ ã ZÄşîXi JãT X||ç ¶Xi
M† Xi ÄîöşN'çãî



● Gradient Descent

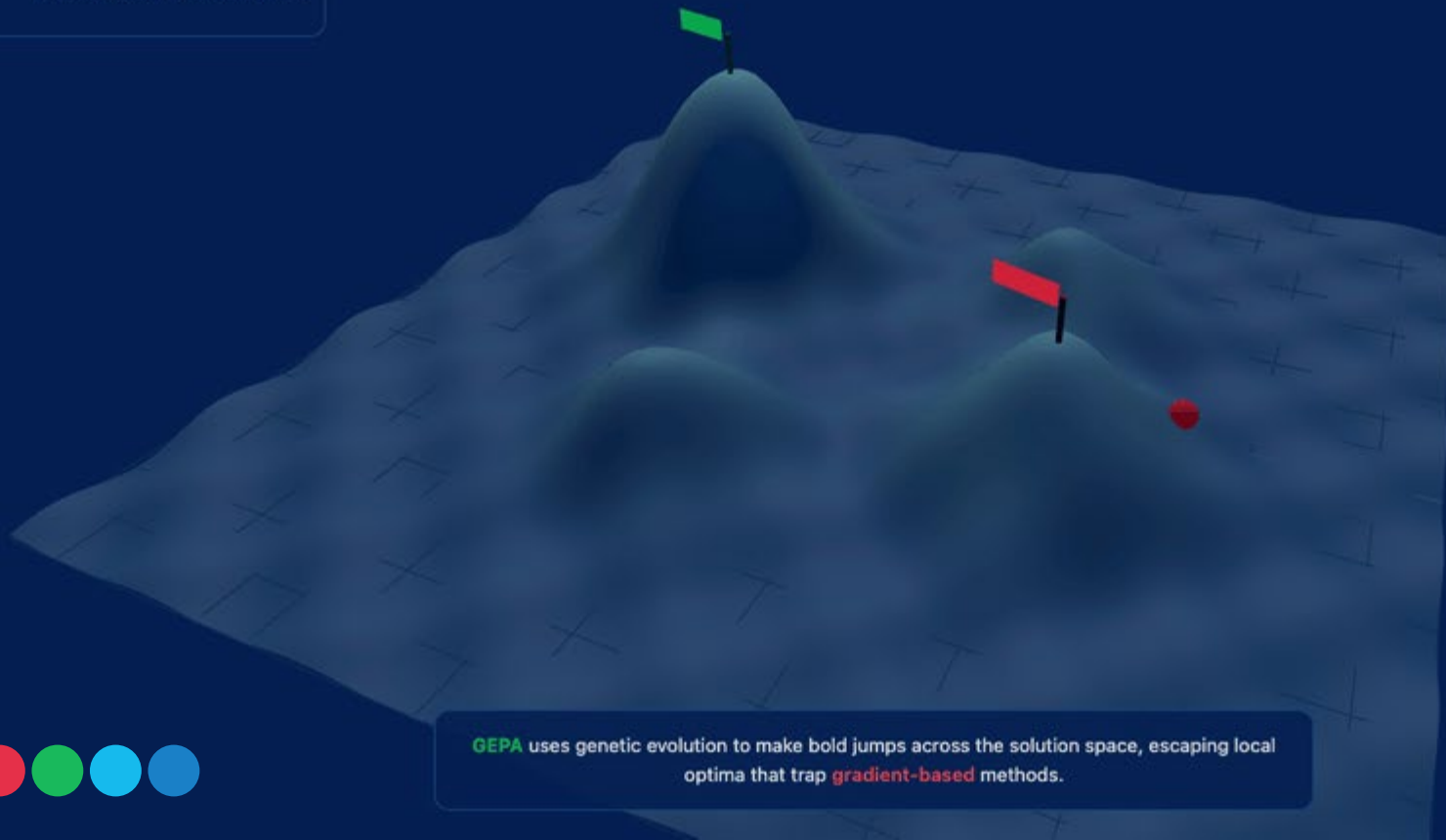
Tiny careful steps uphill
→ Gets stuck at local optimum

● GEPA Prompt Evolution

Large evolutionary leaps
→ Explores globally, finds best peak

■ Local Optimum (Good)

■ Global Optimum (Best!)



GEPA uses genetic evolution to make bold jumps across the solution space, escaping local optima that trap gradient-based methods.





Where else is text used?

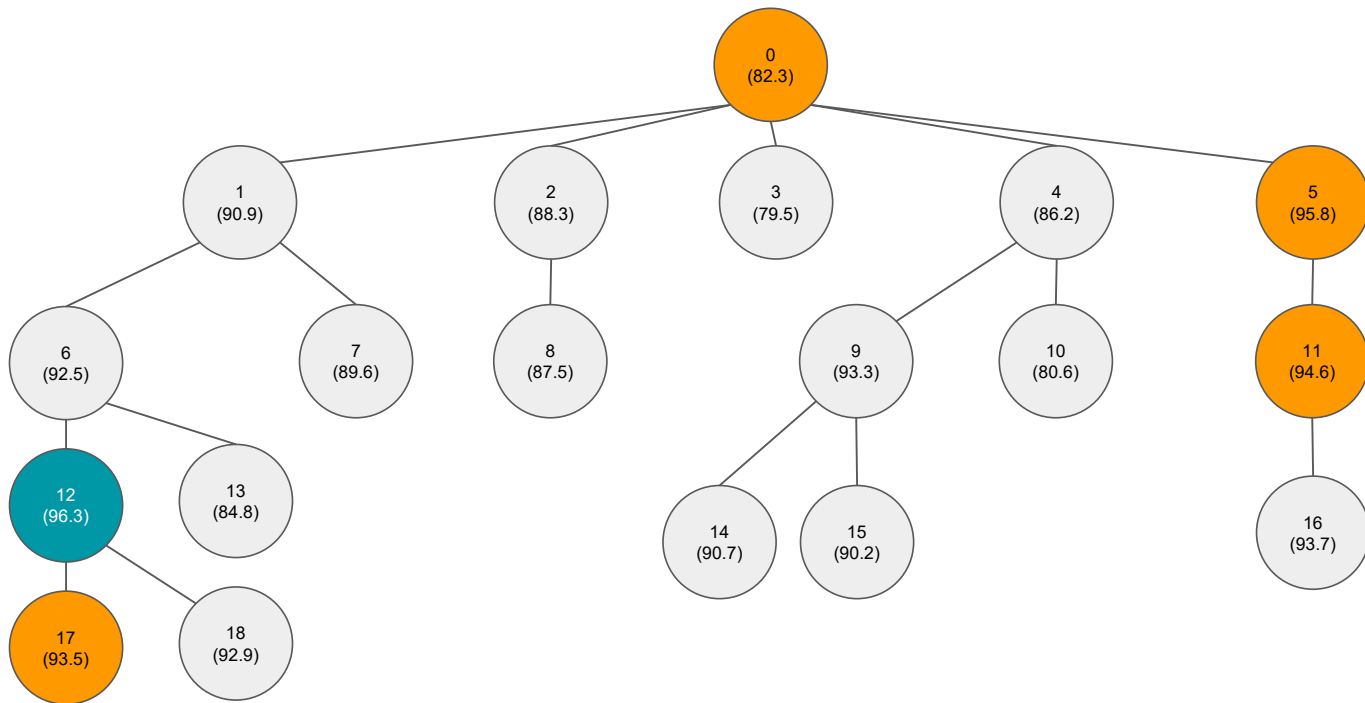


What Else Can GEPA Optimize?

- Chat system prompts (this demo)
- LLM-as-judge prompts (this demo)
- Skill instructions and AGENTS.md
- MCP server descriptions
- Tool-routing logic
- Meta-optimization for custom instructions
- Any module in a compound AI system



; JìXöç 8 êf ä J \$ JãTÄJöX?XNř çã



Pareto Frontiers as Trust Metrics



Old Way

"This version feels better"

GEPA Way

"This version **dominates** the previous one across agreed trust dimensions"

Instead of asking "Is the model good enough?"

Illegible text in a light blue box, possibly representing a corrupted or placeholder image.



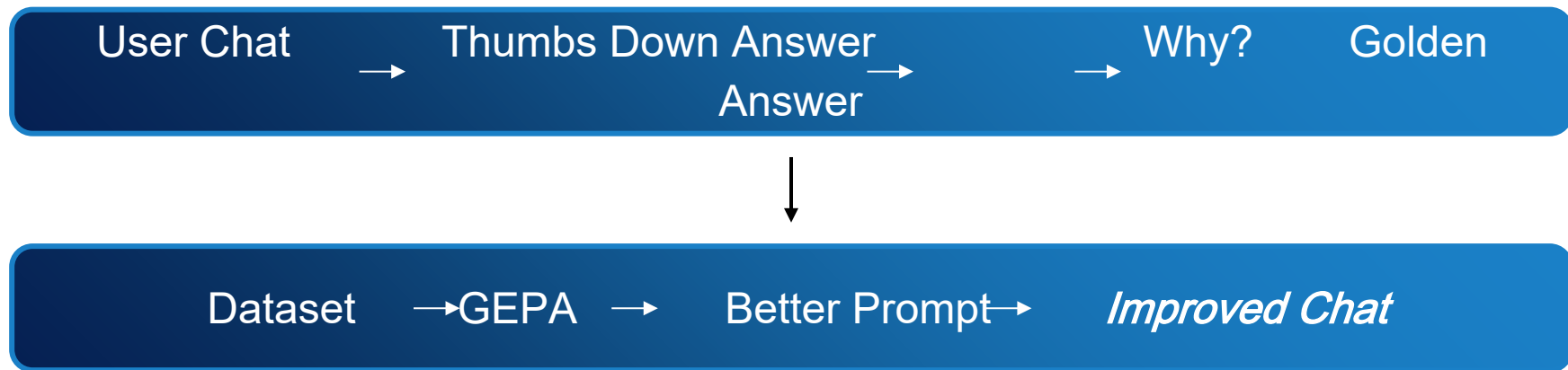
Side-by-Side Comparison



Dimension	RL/Fine-Tuning	GEPA
What Changes	Model Weights	Instructions & Logic
Infrastructure	GPUs, hosting, MLOps	Serverless inference
Vendor Flexibility	OSS Only	OSS + Claude/GPT/Gemini
Cost Model	Fixed + High	Variable + Low
Time to Revert	Days - Weeks	Minutes



The Self-Improving Loop



Your feedback improves **YOUR** company's AI,
not just the foundation model



Before: The Baseline Prompt



```
class AssistantSignature(dspy.Signature):  
    """Answer questions, optionally using provided context."""  
  
    context: str = dspy.InputField(desc="Optional context to inform  
    question: str = dspy.InputField(desc="The user's question to answer  
    answer: str = dspy.OutputField(desc="A clear, accurate, and helpful
```

Generic, minimal domain knowledge

67%

Avg score

35%

Strict accuracy



x Xi ↑+ (; l8 êf ä ÄXT ; içä êö



Task inputs:

- A question about a specific chemical reaction or process related to cooking, baking, or food preparation
- Optional context...

Generated outputs:

- A detailed explanation of the chemical reaction or process
- Clear description of what happens chemically, why it's important



x Xî ↑ + (; l8 ê f ä ÄXT ; ì ç ä ê ö



Domain-specific knowledge:

- Chemical reactions: Maillard reaction, caramelization, emulsification
- Cooking techniques: sautéing, roasting, boiling, steaming
- Food chemistry: starches, proteins, carbohydrates, fats

Strategies:

- Use analogies to explain complex chemical concepts
- Provide step-by-step explanations of cooking processes



A| X>Xî §' öï



Baseline

67.3% Avg score
35.3% Strict accuracy

8 êf ä ÄXT

◦◦ ~~ç~~ ¶z îNçìX
◻◻ ~~è~~ ?ö Äö JNN§ìJN©

î Ä → Ä êîç ¶Xä Xäö Ä J ¶XîJzXîNçìX
î' → Ä êîç ¶Xä Xäö Ä îö Äö JNN§ìJN©

+ (; îöï ↑ ℓ ° % N ä T Ä J ö X P s ◻ ç ä ; J î X ç Z ç ä f X î P s • % X ¶ J § J f ç ä î

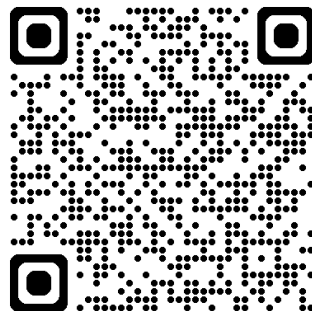


GEPA at Scale



	One-Shot (GPT-5 API)	Agentic + DSPy (self-hosted) Recommended
 Cost	Baseline	↓ ~75x cheaper
 Coverage	~13% of active shops (cost-constrained)	↑ ~100% of active shops ✓
 Quality	Baseline	↑ ~2x F1

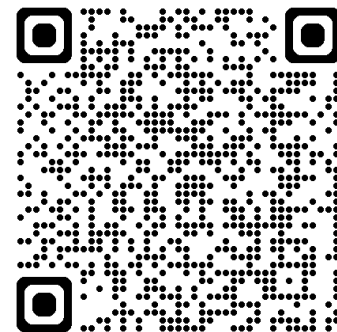
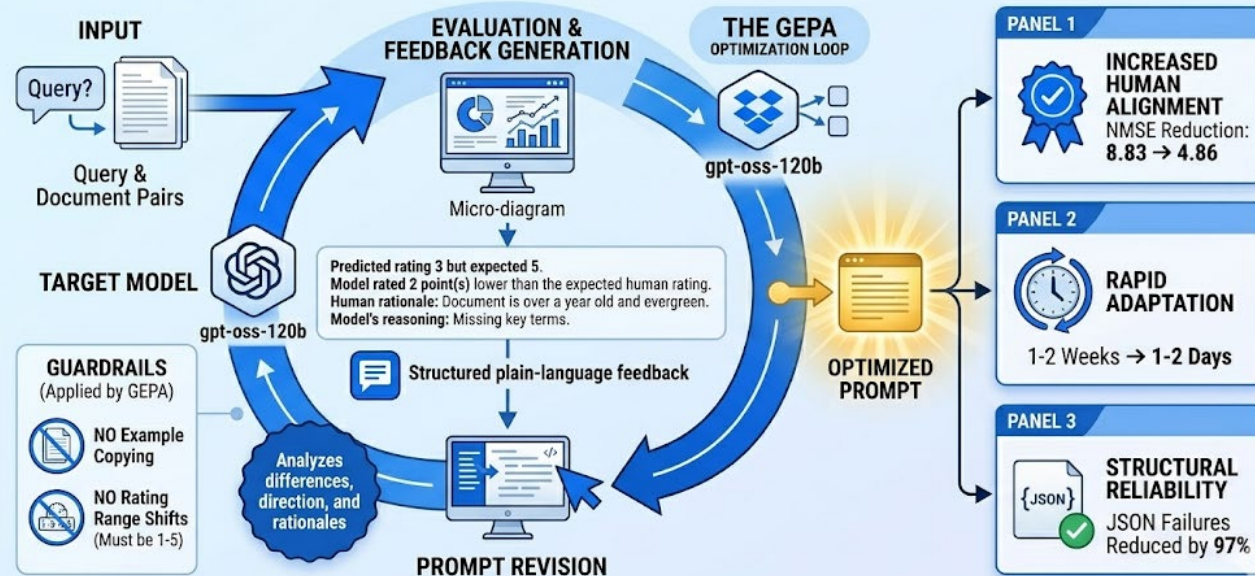
- The old system couldn't run at full coverage — the API costs were prohibitive. We could only afford to analyze a fraction of shops daily.
- Self-hosted Qwen + DSPy optimization: **75x cheaper, 7.5x more shops, 2x better quality.**
- The cost ceiling was the coverage ceiling. Removing it meant the entities that most needed scrutiny — smaller, newer shops — finally got analyzed.



+ (; Jö?N' X



GEPA: The Optimization Engine Behind the Efficiency Breakthrough



+ (; Jö?N' X



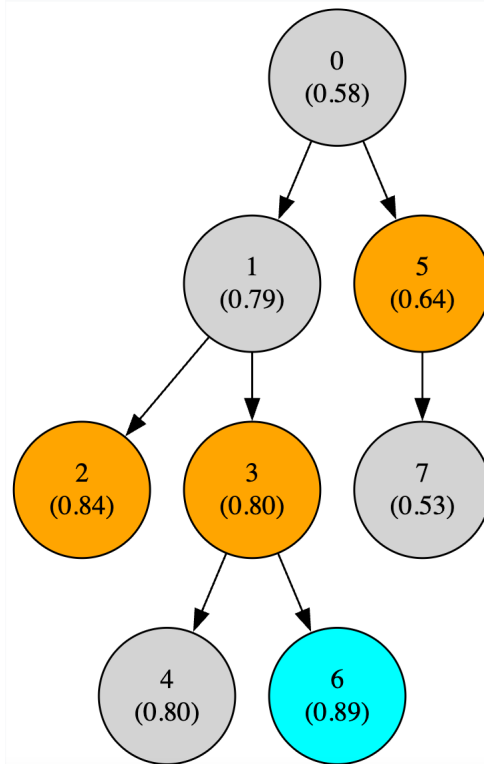
\$§iîXäö\$ Ääö

⤴ + (; §îXT öç' XIiã êiçêXi í §Xi©îXä Jäf'N Zçì J
TJöJ' JÜX

⤴ #JîXÄX● P8 êfä ÄXT' Ä +ℓ +JÄ=

⤴ 2XIiãXTîöJãTJîTêJ†XiãîßÄÄöXTJöJ

⤴ &JöJîXößJîJMXöçMXRöJNöXTZçäXRÄfãz
iXëçifãzJãT¶JÄJöXT

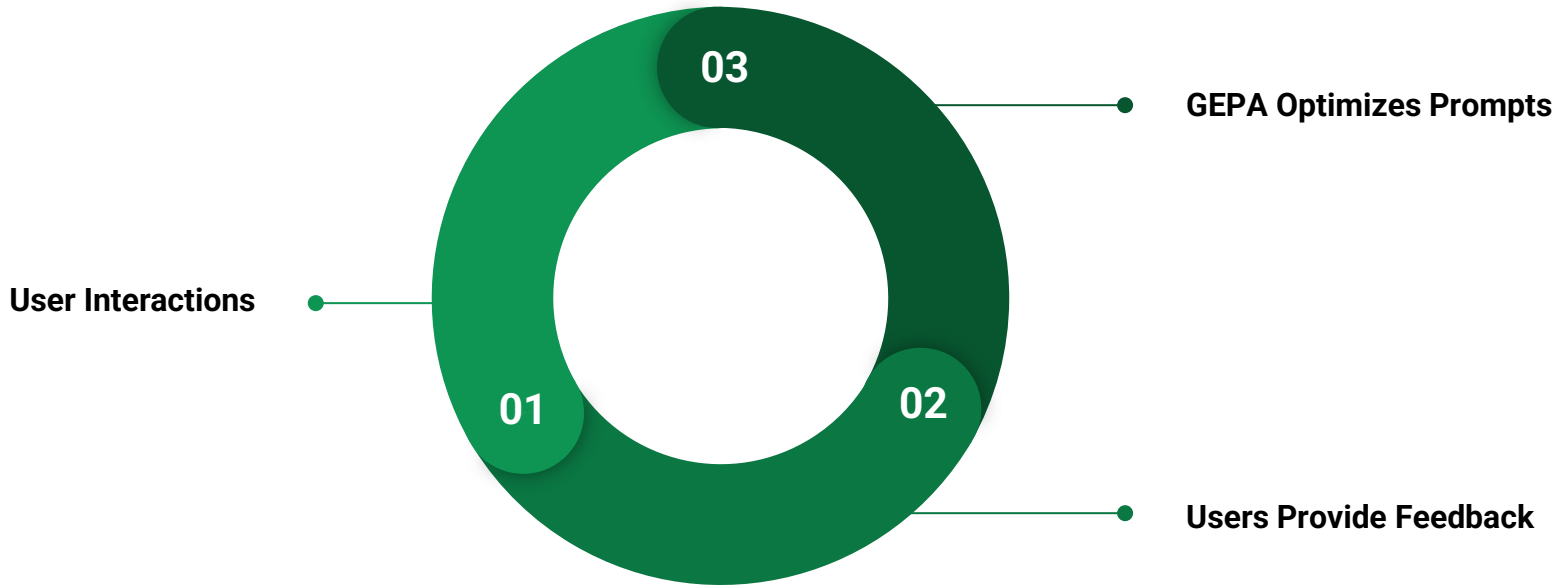




Before	After
Generic	Domain-Specific (culinary chemistry)
No Examples	Task Examples From Failures
No Strategies	Learned Strategies
Vague Outputs	Prescriptive Outputs

?? Z§ ©J§TÄJMXJãT iX||XiîÄMX=





Regular Prompt Optimization Allows AI to Learn



Flipping the Script



Every time you upvote/downvote on ChatGPT, you're helping OpenAI make a better foundation model.

You're not helping YOUR company's AI get better.

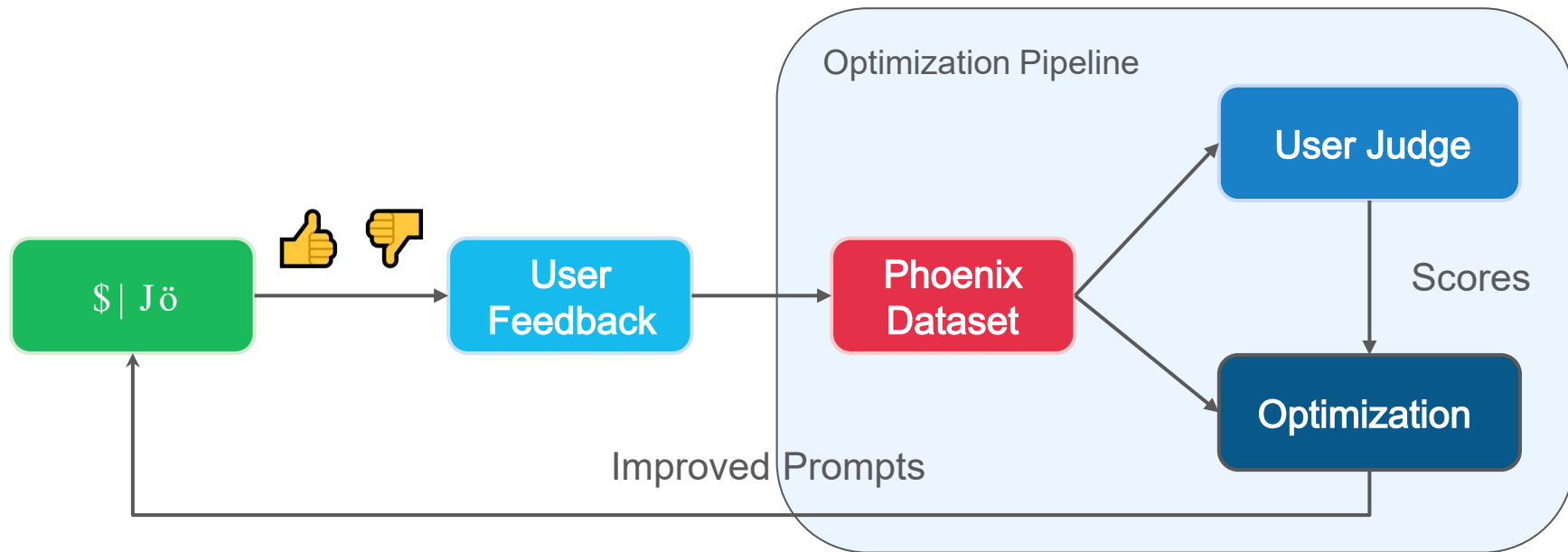
GEPA flips this: Your feedback improves YOUR AI

Create competitive advantage, not just “rising tide lifts all boats”





User Feedback Loop



How Do We Leverage This Data?



We can demonstrate that AI can reflect on its own failures when given examples.

1. **Gather Examples:** Collect your team's top 5 instances of poor AI performance this week.
2. **Request Improvement:** Present the failed interaction to the AI and ask it to generate a refined prompt that would yield better results.
3. **Iterate:** Repeat the process for each example.
4. **Validate:** Test the newly generated prompt and refine further as necessary.

I used this prompt:

<prompt>

and this input:

<input>

That gave me this output:

<output>

But the output wasn't good for these reasons:

<reasons>

Give me a new <prompt> that helps fix this issue

Here is an updated prompt:

<new prompt>



Tomorrow!



On Wednesday, Start Here 🧐

1. **Pick one AI workflow that gets complaints** : customer-facing chat, internal doc search, whatever gets the most thumbs-down.
2. **Add evals if you don't have them** : even simple pass/fail from user feedback counts.
3. **Run Optimize-Anything against it** : one prompt, one afternoon. Read the principles it extracts.
4. **Show the diff to your team** : the before/after is the proof. Plain English, fully auditable.



Questions?



Pick one up from me or Ben or at our booth 😎

