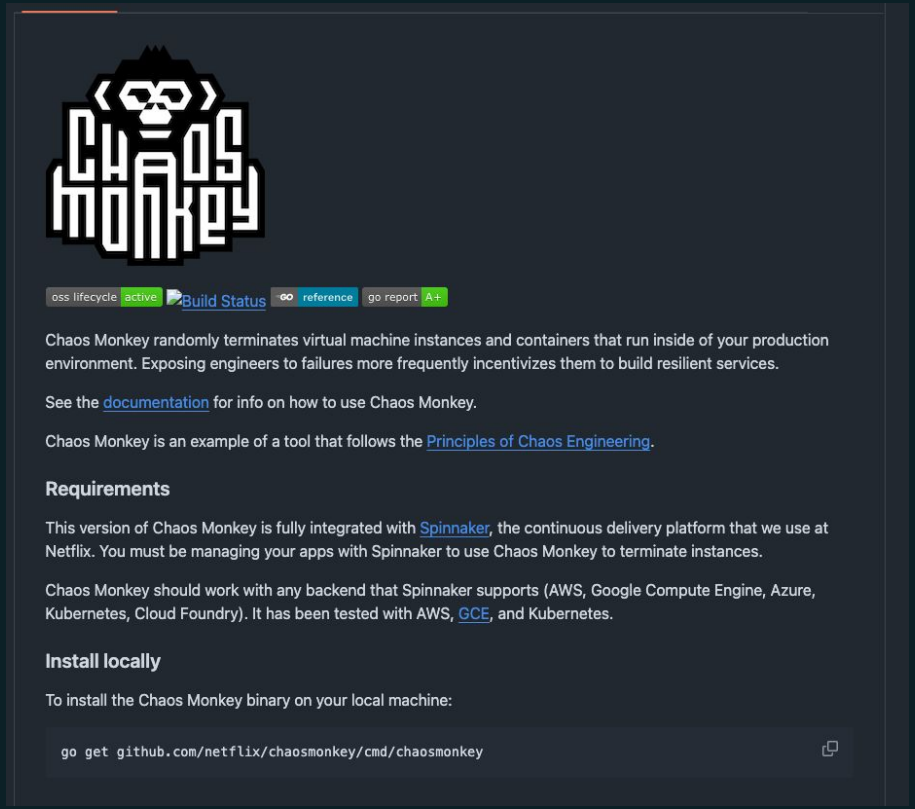
databricks

Chaos LLama

Genie Automated Optimization Framework w/ Introspective AI

Akil Thomas – AI/ML Sr. Specialist Solutions Architect
03/2025

Name Inspired in part by the Netflix Engineering team's Chaos Monkey



oss lifecycle active Build Status reference go report A+

Chaos Monkey randomly terminates virtual machine instances and containers that run inside of your production environment. Exposing engineers to failures more frequently incentivizes them to build resilient services.

See the documentation for info on how to use Chaos Monkey.

Chaos Monkey is an example of a tool that follows the Principles of Chaos Engineering.

Requirements

This version of Chaos Monkey is fully integrated with Spinnaker, the continuous delivery platform that we use at Netflix. You must be managing your apps with Spinnaker to use Chaos Monkey to terminate instances.

Chaos Monkey should work with any backend that Spinnaker supports (AWS, Google Compute Engine, Azure, Kubernetes, Cloud Foundry). It has been tested with AWS, GCE, and Kubernetes.

Install locally

To install the Chaos Monkey binary on your local machine:

```
go get github.com/netflix/chaosmonkey/cmd/chaosmonkey
```

Name also inspired by a concept in AI & Physics called Criticality. The line between Order & Chaos

The Interplay of Large Language Models, Reinforcement Learning, and Self-Organised Criticality

LLMs, like GPT-4 from OpenAI, have already shown promising capabilities, exhibiting emergent behavior as they scale up. The addition of reinforcement learning (RL) — learning from feedback and adjusting responses accordingly — further enhances their potential. Yet, I believe that the most fascinating conjecture stems from the application of the SOC concept to this domain.

SOC refers to the tendency of large systems to self-organize into a critical state, where a minor disturbance can cause large-scale effects — a phenomenon seen in a myriad of natural systems. Applying this to LLMs, I propose that as they grow and learn through RL, these Generative AI systems might self-organize and reach a critical point of complexity and learning

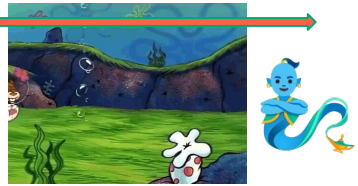
Problem Statement:
Genie customer engagements are a extremely manual process, riddled with idiosyncratic and esoteric edge cases.

Common Challenges with Genie Engagements

Highly Manual

- Error Prone
- No version control on prompts
- A laborious hunt to find relevant metadata for Genie space (example sql queries, functions, evaluation dataset, etc.)

custom solution



The backdrop of the engagement is typical bake offs from pre-existing solutions

- Frequently we are engaged after the customer has implemented a text2sql solution. If we dont make quick wins, customer can **lose confidence rapidly**

No advance quantitative measures to track progress. Leading to confusion

- If a change is made to either system instructions, column descriptions, or example sql queries what is the quantitative effect, on the overall performance?
- No ability to add custom metrics specific to the use case
- Benchmarks feature on genie is also highly manual (cant upload a eval dataset) with no custom metrics.

🤔

What if there was a solution that automates the highly manual process of optimizing the levers of Genie (metadata) to improve performance?

[WIP] 🦙Chaos LLama Architecture

Evaluation Dataset

Question	Ground-source truth sql query	Issues

Learning Rate Analogy - Request the LLM makes small incremental changes to metadata such that to mitigate risk of overshooting the target akin to nueral networks learning rates

DSPY

LLM

Introspection AI

Or

If accuracy is low, or lowered

Optimize / Update Metadata

Keep Metadata Constant

Metdata (System Instructions + Column Descriptions)

Unity Catalog

Genie Conversational API

generated sql

Databricks Mosaic Agent Evaluation

Various Gen-AI Metrics (Data Intelligence)

Metrics

LLM as a Judge

Global Guidelines

Row Level Guidelines

Our cost function is a sql generation accuracy metric. After each iteration (change to metadata) the DataIntelligenceAgent will observe the accuracy metric and suggest a new change.

🤓 Theory:

With a robust enough Evaluation Dataset and the correct array of ai judges and metrics, an agentic system can analyze this data intelligence to optimize the genie levers available to us, to converge on the solution to maximize performance.

Core AI Concepts of the Framework

🤔Introspective AI

In an iterative process, leveraging a closed feedback reinforcement framework or an llm to **reason** about how to optimize metadata based on data intelligence.

👨‍⚖️AI Judges

Use Mosaic AI Agent Framework to critique the generated sql query amongst various different perspectives from a metric standpoint.

🧠📈Data Intelligence

- Use Built-In Judges, Custom Judges, and general statistical analysis, etc. for telemery
- Consider this analogy. Imagine you had a camera (remote sensing) that was 1 foot above the ground. You could conclude that the earth is flat. If that camera was near the moon looking at earth, you can conclude that the earth is round. In juxtaposition from the moon, you cannot conclude that there were living organism in the ground unless you took a snapshot from 1 foot away. Data Intelligence strives to take as many snapshots from many different angles as is possible due to compute and time constraints

Aka meta-learning aka self-supervised refinement aka self feedback

What is Introspective AI?

Introspective AI

A new idea taking shape in the ai community!

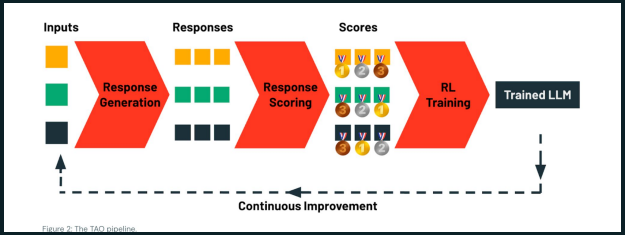


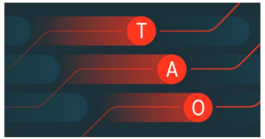
Figure 2: The TAO pipeline.

Picture this

- 🧠 ➜ 📝 (AI generates an answer)
- 🔍 ➜ 🪞 (Then it looks in a mirror, evaluating the logic of what it just did. Critiquing its assumptions and reasoning)
- ✍️ ➜ 🔧 (It tweaks its strategy based on what it saw)

Blog / Mosaic Research / Article

TAO: Using test-time compute to train efficient LLMs without labeled data



Published: March 25, 2025 | Mosaic Research | 8 min read | By The Mosaic Research Team

arXiv > cs > arXiv:2303.17651

Computer Science > Computation and Language

[Submitted on 30 Mar 2023 (v1), last revised 25 May 2023 (this version, v2)]

Self-Refine: Iterative Refinement with Self-Feedback

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegr
Hermann, Sean Welleck, Amir Yazdanbakhsh, Peter Clark

Like humans, large language models (LLMs) do not always generate the best output on their firs
through iterative feedback and refinement. The main idea is to generate an initial output using
supervised training data, additional training, or reinforcement learning, and instead uses a sing
generation to mathematical reasoning, using state-of-the-art (GPT-3.5, ChatGPT, and GPT-4)
with the same LLM using conventional one-step generation, improving by ~20% absolute on ave
our simple, standalone approach.

Comments: Code, data, and demo at this https URL
Subjects: **Computation and Language (cs.CL)**; Artificial Intelligence (cs.AI); Machine Learning (cs.LG)
Cite as: arXiv:2303.17651 **[cs.CL]**
(or arXiv:2303.17651v2 **[cs.CL]** for this version)
https://doi.org/10.48550/arXiv.2303.17651

DSPy *Programming—not prompting—LMs*

downloads/month 641k

DSPy is the framework for *programming—rather than prompting—language models*. It allows you to iterate fast **modular AI systems** and offers algorithms for **optimizing their prompts and weights**, whether you're buil classifiers, sophisticated RAG pipelines, or Agent loops.

DSPy stands for Declarative Self-improving Python. Instead of brittle prompts, you write compositional *Pyth* use DSPy to **teach your LM to deliver high-quality outputs**. This lecture is a good conceptual introductio community, seek help, or start contributing via our GitHub repo and Discord server.

DeepSeek-R1-Zero: The Proof That Reinforcement Learning Alone Can Induce Reasoning

The most radical idea behind DeepSeek-R1 started with an experiment:

> What if we removed supervised fine-tuning (SFT) entirely and trained a model using only reinforcement learning?

The result was **DeepSeek-R1-Zero**, a model that:

- **Developed self-verification, reflection, and structured reasoning** — without seeing human-labeled Chain-of-Thought (CoT) data.
- **Learned to reevaluate and correct its own mistakes.**
- **Naturally extended the depth of its reasoning over time.**

> Aha Moment #1: LLMs Do Not Need Human-Labeled Reasoning Steps to Learn Structured Thought

Traditionally, LLMs are fine-tuned on human-labeled **reasoning chains** before reinforcement learning. The belief has been that without this, models will **not generalize multi-step reasoning well.**

🥅 Goals of Chaos Llama

A tiered approach for levels of success and value creation

1️⃣ A Testing Framework

Create a simple framework to allow Genie Developers and owners to continually test genie robustly with the advance tools of Databricks' Mosaic AI Evaluation

2️⃣ Genie Optimization Copilot

Use the Chaos Llama Framework to output suggestions on how to adjust the metadata, and have the developer work off from that starting point, interactively working with genie to achieve the desire performance levels

3️⃣ Complete Automation (and world domination 😂)

Completely automate the highly manual process of optimizing Genie Rooms

Link to Quick Demo!

Future Considerations & Call to Action

- Implement DSPY
- Continue to Ideate on SQL LLM-as-a-judge metrics
- Stream the output of the metrics during the process to see the impact of genie sql generation query overtime.
- If success at Pepsi, educate the field to collect more real world examples
- Databricks is rapidly moving towards a future where we automate various tasks of an enterprise platform on the customer behalf (i.e. predictive optimization).
  - Imagine if we can have customers run chaos llama as a job -> produce a bevy of telemetry data and then perhaps have databricks offer a paid service to finetune off that metadata to provide a better genie experience.
- Use Open AI as LLM vs llama 3.3 70b as logic would beg that you should use the same llm that genie uses to introspect about why its thinking is wrong.
- Use a test set (out of sample of eval dataset) to test on questions Chaos LLama did not see in its finetuning process
- Use views to update data model (as a last resort)!
- Request for Developers!
- Add Validation testing
- Leverage open ai

Appendix

🥅 Risk in the Chaos Llama Framework

🔭 Pitfalls to watch out for

Overfit on the Evaluation Dataset

Mitigate this by creating a validation / test set that has been unseen by Chaos LLama during uthe optimization phase

Cost

- Cost must be controlled
  - Chaos llama uses sql warehouse to evaluate sql results (can be mitigated by looking at sql semantic equivalence)
- Can be mitigated with the parameters of Chaos llama. Chaos LLama can be ran as a job. Constrain cost by setting a limit on epochs

Pepsi Case Study

Pepsi Results

~3x improvement in accuracy from baseline of 25% to 80% on eval questions for 5 hours in iteration.