Enhancing Gen-AI with Rust: Leveraging Safety and Performance for Scalable AI Systems

Karan Khanna

Table Of Contents

Introduction to Gen-AI

- Generative Artificial Intelligence is subset of deep learning that employs training on supervised, semi supervised or even unsupervised learnings.
- Differs from discriminative in the manner it operates i.e. it creates new content based on the data it was trained on.
- Popular examples are ChatGPT, Dall-E, Gemini and many more.

Building Blocks of Gen-AI

- FMs (Foundational Models)
  a. LLMs (Large language Models)
  b. Domain Specific Models
- Inference Code (Bridge between generative model & applications)
- Data Pipelines (preprocessing, formatting)
- Libraries, APIs (Send & Receive), Applications

Introduction to Rust

- Rust is known to be systems programming language.
  a. Initiated at Mozilla Research in 2006
  b. First stable launch in 2015
- Known for its concurrency, performance, memory efficiency and safety.
- Adopted across the board by the likes of Dropbox, Yelp, Meta, AWS…
- ~ 3 Million coders use Rust
- Performance critical applications - embedded, internet scale servers…

Core Features of Rust

- Blazingly Fast
  a. LLVM as compiler infrastructure.
  b. Cache-Efficient Placing of data structures.
  c. No Garbage Collection

- Concurrency
  a. Channels
  b. Sync & Send Traits

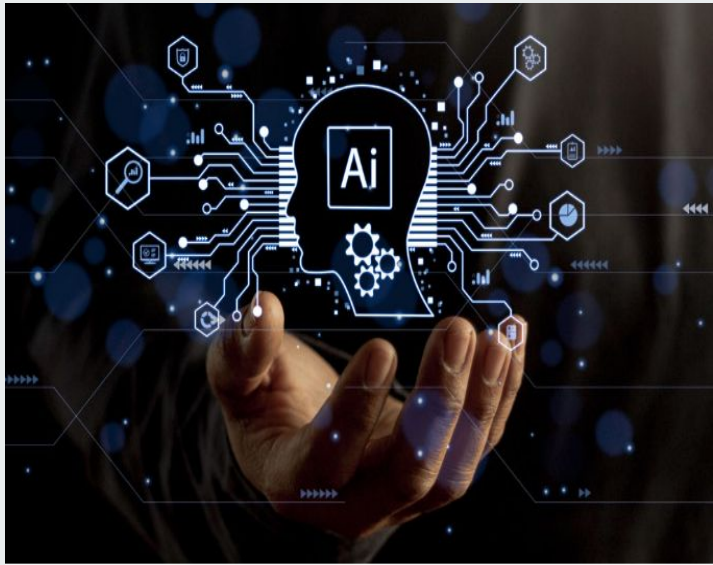
- Safety
  a. Compile time gates

- Memory Efficiency
  a. Ownerships & borrowing

Challenges in AI Applications

Technical Challenges in today’s scenario regarding building/running AI Application.

- Memory Leaks
- Critical Performance barrier
- Scalability
- Huge data sets
- Computational resource crunch
- Integrations



Why Rust in AI Applications ?

Challenges in AI Apps

- Memory Leaks
- Critical Performance barrier
- Scalability
- Huge data sets
- Computational resource crunch
- Integrations

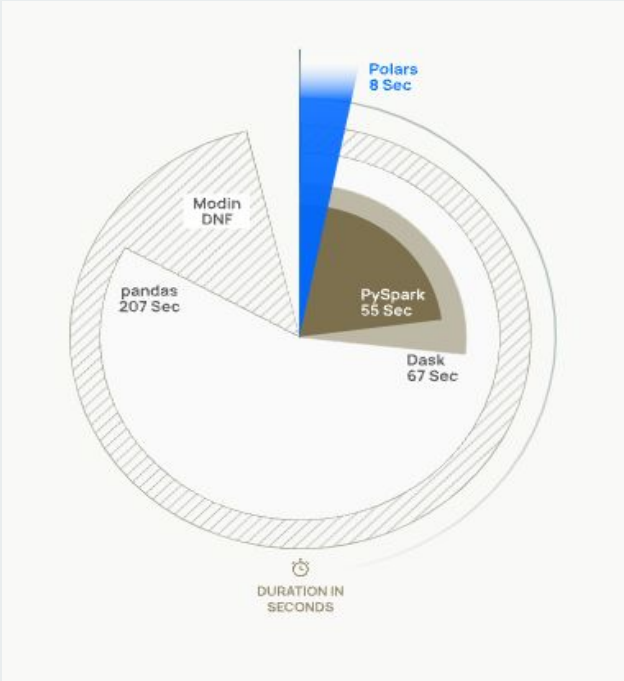
Core Features : Rust

- Blazingly Fast
- Concurrency
- Safety
- Memory Efficiency
- …

Case Study: Polars

Polars

- Data analysis library
- Polars is written in Rust
- Highly performant
- Benchmarked ~30X performance against pandas (python)



Current State of Rust: Overview

- Rust is popular and established in the mainstream engineering projects like Firefox, Dropbox file sync, Meta's Mononoke, Cloudflare Firewall products, Discord's Blurple and Redox OS.
- It's ecosystem for AI applications is strongly emerging and still maturing.
- Few opportunities today are:
  a. Ecosystem maturity
  b. Learning Curve
  c. Community & Support
  d. Tooling

Current State of Rust: Existing Ecosystem

- Few examples of Rust libraries and frameworks are:
  a. **Tch-rs:** A Rust binding for PyTorch, allowing the use of PyTorch's deep learning capabilities in Rust.
  b. **Rust-ML:** An organization and community working on various machine learning libraries in Rust.
  c. **Linfa:** A machine learning library in Rust, focusing on classical machine learning algorithms.
  d. **Burn:** A newer deep learning framework that leverages Rust's performance and safety guarantees.
  e. **HuggingFace-tokenizer:**Provides an implementation of today's most used tokenizers
  f. **Google-Research'sExactSubstr deduplication:** deduplicate language model datasets
  g. **Tensorflow/Rust:** Tensorflow bindings in Rust

- Integrations with other languages:
  a. Python using library PyO3
  b. C/C++ using Foreign Function Interface

- Newer libraries for varied use cases like Image processing, data framing, nlp, graphical modeling, GPU libraries are quickly coming up.

Current State of Rust: With AI

Despite the challenges, Rust is being adopted in various aspects of AI development:

1. **Model Deployment**: Rust is being used in deploying machine learning models due to its performance and safety features. Projects like `tangram` allow developers to train models in other languages (like Python) and then deploy them using Rust.

2. **AI Libraries**: AI and machine learning libraries are emerging in Rust. Libraries like `linfa` aim to provide a comprehensive toolkit for machine learning in Rust, covering tasks such as data processing, model training, and evaluation.

3. **Integration with Python**: Many developers use Rust for performance-critical parts of their AI applications while keeping the flexibility of Python for the rest. The `PyO3` and `rust-cpython` libraries enable seamless integration of Rust and Python, allowing developers to write Python bindings for Rust code.

4. **Edge AI**: Rust is being explored for deploying AI models on edge devices, where performance and resource constraints are more critical. The language's ability to produce small, efficient binaries is an advantage in these scenarios.

5. **AI apps:** AI applications are on the rise be it real time or data analysis functionalities. IOT, research, and large scale are other areas where Rust is seeing growth.

Future Prospects: Rust & AI

- AI apps are going to be near real time and more computational and scalable.
- While Rust has steep learning curve but it does fits the bill in every other aspect from safety to concurrency to performance for these use cases.
- Rust can fit in any stack in the Gen AI building blocks be it FMs, Deployments, libraries or services.
- Rust is going to be seen in more embedded/IOT/large traffic applications along with research and experimentational use cases.
- Since Rust is not yet mainstream for AI, this presents the opportunity for growth and adoption. Community support and Tooling needs to be more mature to support the growth.

Conclusion

- Rust core features enable it to be explored, adopted in Gen AI space.
- It’s main characteristics such as speed, safety, interoperability, cross platform development and performance optimization makes it apt for AI use cases.
- There are use cases like real time applications, IOT, embedded systems and even large scale applications where Rust could give best ROI.
- There is need to mature the tooling, community support and ease in learning curve to speedify the adoption of Rust in AI space.
- There is more upside and growth opportunities with the winning combination of Gen AI & Rust

References

1. https://www.rust-lang.org/
2. https://llvm.org/
3. https://github.com/vaaaaanquish/Awesome-Rust-MachineLearning
4. https://www.lpalmieri.com/posts/2019-12-01-taking-ml-to-production-with-rust-a-25x-speedup/
5. https://github.com/huggingface/tokenizers
6. https://github.com/google-research/deduplicate-text-datasets
7. https://www.arewelearningyet.com/
8. https://github.com/tensorflow/rust
9. https://engineering.fb.com/2021/04/29/developer-tools/rust/
10. https://www.youtube.com/watch?v=R_jW8yvc_GU
11. https://pola.rs/

Thank you !

Karan Khanna | LinkedIn