Rust for MLOps

With Amazon SageMaker

Noah Gift

Duke Executive in Residence
Founder Pragmatic AI Labs
5x Best Selling O'Reilly Author
AWS ML Hero

Introduction

Why?

- Brief overview of Rust, MLOps, and Amazon SageMaker
- Why Rust is a good choice for MLOps
- How Amazon SageMaker supports MLOps

Rust: 7 Years as the Most Loved Language

(Stackoverflow 2022)

Rust for MLOps

Rust language features that benefit MLOps?

OPERATIONAL SYNERGY

- Performance: High-speed execution
- Memory safety: Eliminates common bugs and security vulnerabilities
- Concurrency: Efficient parallelism for large-scale data processing
- Binary Deployment

Rust libraries for machine learning and data processing?

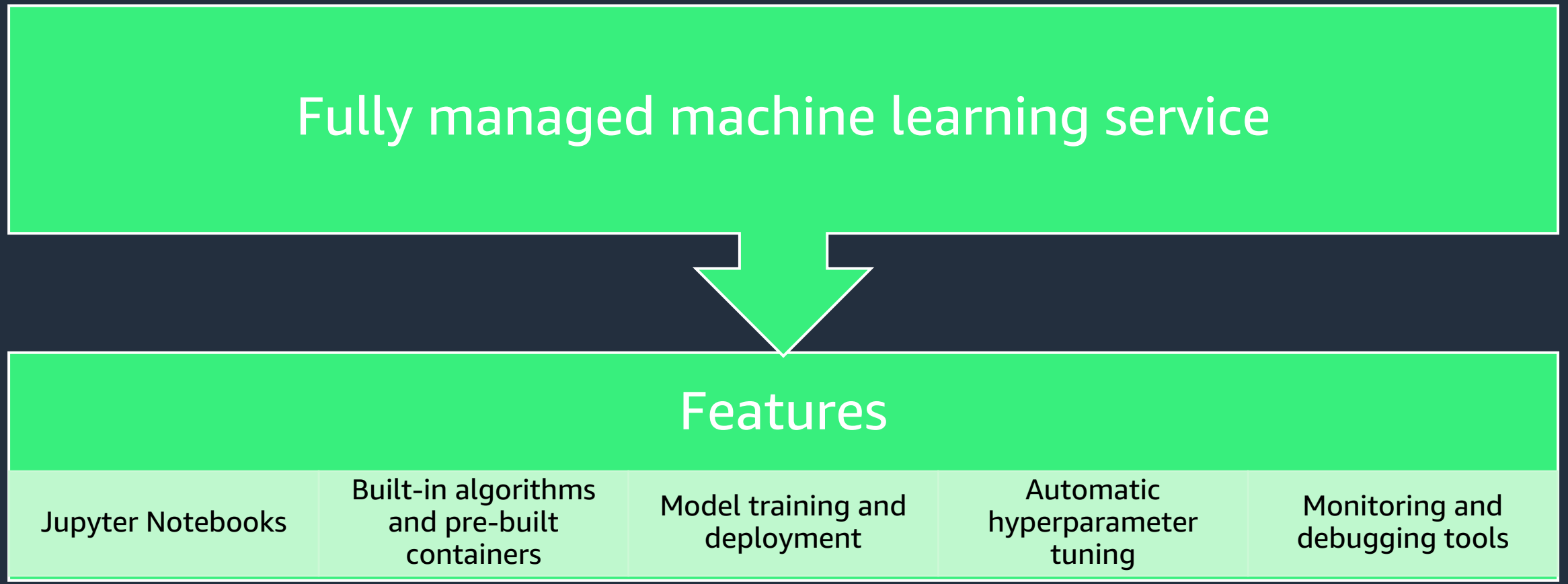
KEY LIBRARIES

- tch-rs: Rust binding for PyTorch
- linfa: Machine learning algorithms
- Polars: DataFrames
- ONNX: With ONNX bindings for Rust, you can incorporate ONNX models into your MLOps workflow

Amazon SageMaker Overview

Amazon SageMaker

OVERVIEW



Integration of Rust and Amazon SageMaker

Integration Touch Points

SOLUTIONS

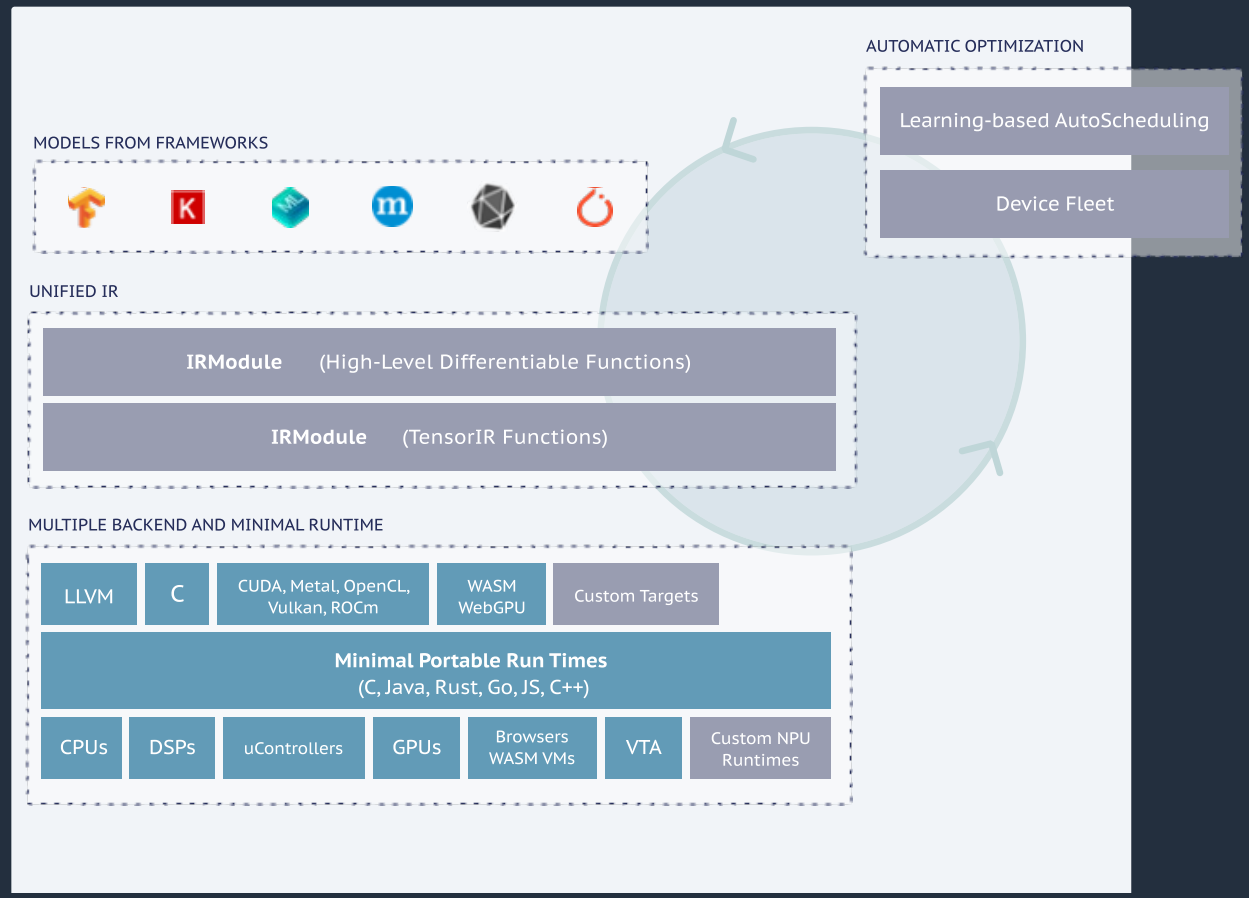
- Use Rust for AWS Lambda
- Create custom Rust containers for SageMaker
- Use Rust AWS SDK for Rust to interact with SageMaker APIs
- EFS to host ONNX inference

Amazon SageMaker Silicon (NEO)

SYNERGY WITH RUST ADVANTAGES

Rust to ONNX then ONNX to NEO

HIGH PERFORMANCE YET MINIMAL PACKAGING



Rust MLOps Workflow

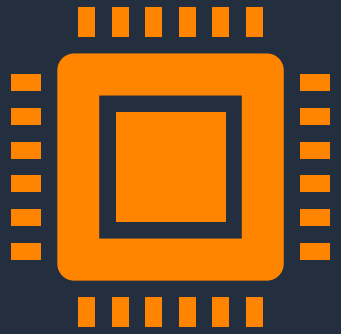
Rust MLOPs Workflows

Use	Training	Use	Model	Monitoring and debugging
Pre-processing: Use Rust libraries to clean, preprocess, and transform data	Model training: Train models using Rust-based ML libraries or custom Rust containers	Hyperparameter tuning: Use SageMaker's tuning capabilities to optimize Rust ONNX models	Model deployment: Deploy Rust models as SageMaker endpoints or use AWS Lambda for serverless deployments	Monitoring and debugging: Leverage SageMaker and CloudWatch for real-time monitoring and debugging

Case Study

Case Study

DETAILS



Present a real-world example of an organization successfully implementing Rust MLOps with AWS Sagemaker

Detail challenges, solutions, and outcomes

ONNX

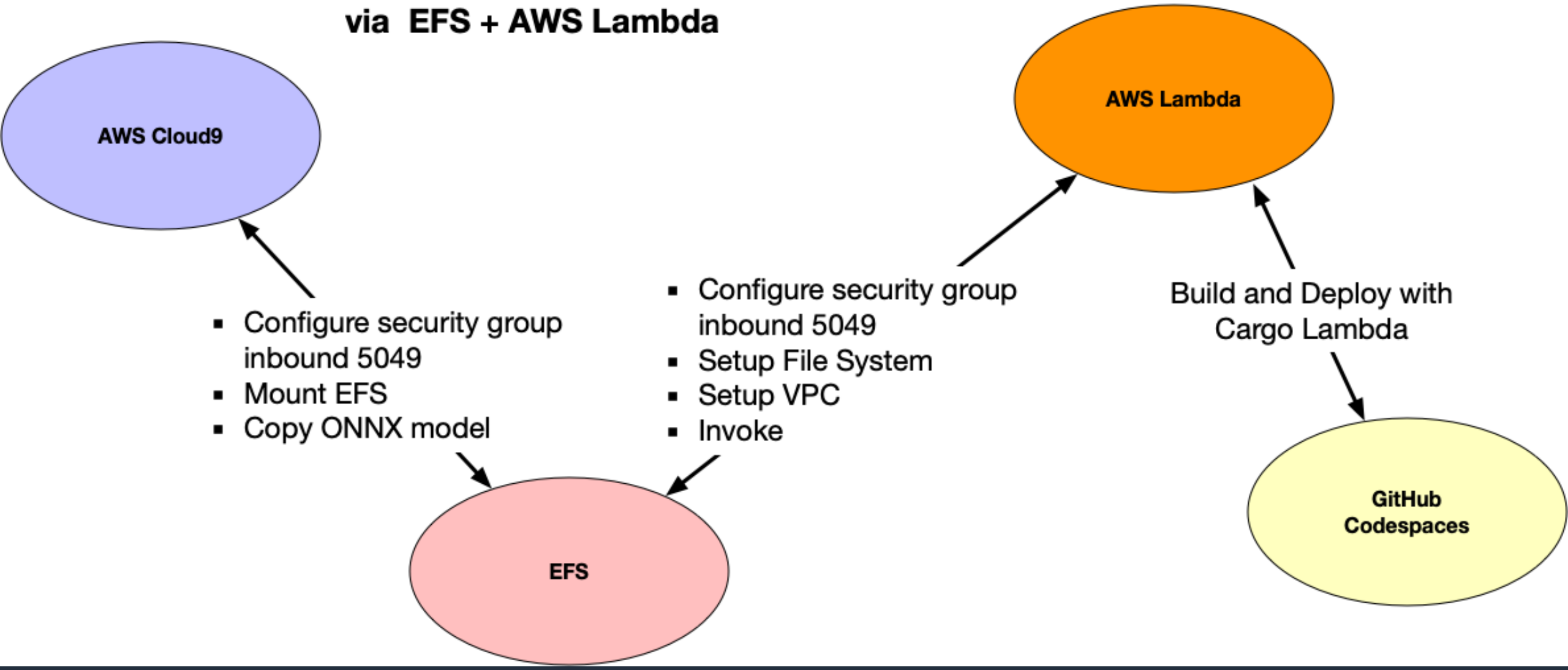
```
use onnxruntime::{environment::Environment, tensor::OrtOwnedTensor};
use std::error::Error;
use std::path::Path;

fn main() -> Result<(), Box<dyn Error>> {
    let env = Environment::builder()
        .with_name("test")
        .build()?;

    let mut session = env.new_session_builder()?
        .with_optimization_level(0)?
        .with_model_from_file(Path::new("your_onnx_model.onnx"))?;

    // Perform inference and process the output.
}
```

MLOPs Inference with ONNX via EFS + AWS Lambda



Demo: Rust Async S3 Lambda Tool

Conclusion

Resources

GitHub Repo
Rust for MLOps Template

GitHub Tutorial Website
Small Rust Tutorial for MLOps

Rust Website
Rust Language

GitHub Tutorial Website
Small Rust Tutorial for MLOps

Thank you!

Noah Gift

Linkedin:
https://www.linkedin.com/in/noahgift/

GitHub Profile:
https://github.com/noahgift

YouTube:
https://www.youtube.com/c/pragmaticailabs

Pragmatic AI Labs:
https://paiml.com