AI is becoming a part of our everyday lives

- Chat GPT
- Bard
- DALL·E 2
- Ansible Lightspeed with IBM Watson Code Assistant
- GitHub Copilot
- Bing Chat
Consumers Trust in AI is Growing

Create an image that represents AI for a presentation.

Sure, I’ll try to create an AI image for your presentation.
Generative AI and Foundation Model Adoption is Growing

CDO Agenda 2024: Navigating Data and Generative AI Frontiers
Operationalizing AI is not trivial

Every member of your team plays a critical role in a complex process

<table>
<thead>
<tr>
<th></th>
<th>Set goals</th>
<th>Gather and prepare data</th>
<th>Develop model</th>
<th>Integrate models in app dev</th>
<th>Model monitoring and management</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business leadership</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data engineer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data scientist</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ML engineer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>App developer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IT operations</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Operationalizing AI is still a challenging process

What is the average AI/ML timeline from idea to operationalizing the model?

Half of respondents (50%) say their average AI/ML timeline from idea to operationalizing the model is 7-12 months.

- 50% 7-12 months
- 15% 3-6 months
- 10% 1 year or more
- 4% Unsure
- 5% Haven’t done this yet / Still in experiment phase
- 26% 1 year or more
- 5% Haven’t done this yet / Still in experiment phase

Source: Gartner Peer Insights, Open Source AI for Enterprise survey, 2023
Complexities of operationalizing models

"a consistent application platform for the management of existing, modernized, and cloud-native applications that runs on any cloud."

"a common abstraction layer across any infrastructure to give both developers and operations teams commonality in how applications are packaged, deployed, and managed."

Source: https://www.redhat.com/en/technologies/cloud-computing/openshift
Our AI/ML strategy

**AI workload support**
Support **AI workload requirements** on Red Hat platforms
- e.g., hardware acceleration, GPU Operator

**Platform for AI-enabled apps**
Provide a consistent, hybrid cloud application platform for **customers** to build, train, and deploy AI-enabled applications
- e.g., Red Hat OpenShift AI

**AI-enabled platforms**
Use **AI models, tools, and services** to accelerate adoption of existing Red Hat products and services
- e.g., Red Hat Ansible Lightspeed, Red Hat Developer Hub
AI for the open hybrid cloud

Enterprise-grade open source hybrid AI and MLOps platform

Red Hat OpenShift AI

Develop, train, serve, monitor, and manage the life cycle of AI/ML models and applications, from experiments to production.

▸ Provide a unified platform for data scientists and intelligent application developers

▸ Scale to meet the workload demands of foundation models: data volume, training time, model size, acceleration, and scalability

▸ Deliver consistency, cloud-to-edge production deployment and monitoring capabilities

▸ Underlying platform for training, serving, and tuning foundation models in Red Hat Ansible Lightspeed with IBM Watson Code Assistant
Red Hat’s AI/ML engineering is 100% open source
Overview of Red Hat OpenShift AI

- Implemented interactive lecture and lab environment for computer scientists and engineers based on Red Hat OpenShift AI
- Currently over 300 users including over 100 concurrent
- Integrates with the Boston University online textbook material, also authored using the Red Hat OpenShift AI
- Fast time to solution: cloud services environment enabled BU to configure and deploy in December for classes that started in January
- Lowers cost: auto-scales based on demand; enables bursty interactive use cases at optimized cost
An open source platform for foundation models

Train or fine tune conversational and generative AI

Training and validation | Workflows

- **CodeFlare**
  - RAY
  - PyTorch
  - KubeRay
  - TorchX

- **MCAD**
  - Job dispatching, queuing, and pecking

- **InstaScale**
  - Cluster scaling

Tuning and interface | Domain specific APIs

- **KServe**
  - PyTorch

- **Hugging Face**
  - ONNX

- **Callikit**
  - Dev APIs, prompt tuning interface

- **TGIS**
  - Optimized text generation interface server

Red Hat OpenShift AI

Red Hat Openshift AI
Distribute workloads to enhance efficiency

**Focus on modeling, not infrastructure**
by dynamically allocating computing power

**Prioritize and distribute job execution**
using advanced queuing for tasks like large-scale data analyses

**Automate setup and deployment**
so you can get up and running with minimal effort

**Manage resources and submit jobs**
using a Python-friendly SDK, which is a natural fit for data scientists

**Streamline data science workflows**
with seamless integration into the OpenShift AI ecosystem
Configure distributed workload clusters more easily

**Process**

1. **Send request** for creation of cluster
2. **Evaluate clusters** for aggregated resources and dispatch when available
3. **Watch pending clusters** to provide aggregated resources and guarantee workload execution
4. **Develop and submit jobs** to retrieve statuses and logs from the clusters
5. **Send request** for deletion of clusters

---

Red Hat OpenShift Data Science

- Data scientist
- Multi-cluster dispatching
- Dynamic scaling
- Ray Python clusters

- Create
- Delete
- Add
Make model serving more flexible

- Use model-serving user interface (UI) integrated within product dashboard and projects workspace
- Serve open source models from providers like Hugging Face
- Support a variety of model frameworks including TensorFlow, PyTorch, and ONNX
- Choose inference servers either out-of-the-box options optimized for foundation models or your own custom inference server
- Scale cluster resources up or down as your workload requires
Serve, scale, and monitor your models

Configure model server
- Model server replicas
- Compute resources per replica
- Model server size
- Model route
- Token authorization

Deploy model
- Model framework
- Model location
- Existing data connection
- Model
- Storage-config
- Feature path
- Next data connection

Models and model servers
- Deployed models
- Tokens
- Tokens disabled

View your deployed model fleet endpoints
Access a range of model performance metrics to build your own visualizations or integrate data with other observability services

- Out-of-the-box visualizations for performance and operations metrics
- Monitor production models for any changes in measured bias
Data Science Pipelines

- Continuously deliver and test models in production
- Schedule, track, and manage pipeline runs
- Easily build pipelines using graphical front end

▸ Orchestrate data science tasks into pipelines
▸ Chain together processes like data prep, build models, and serve models
Flexibility at the edge

Device edge
- Device or sensor
- Red Hat OpenShift AI Model serving

Far edge
- Red Hat OpenShift AI Model monitoring
- Model registry

Near edge
- Red Hat OpenShift AI Pipelines
- Red Hat OpenShift AI Model training

Enterprise

Edge
- Unreliable connection

Core
- Last-mile network

Sensor data, telemetry, events, operational data, general information, etc.

Code, configuration, master data, machine learning models, control, commands, etc.
Opportunities for Research
In order to democratize access to AI for enterprises, models must be cheaper to run and the lineage of those models must be transparent and fully understandable.
Challenges Customers Face

High Cost of Inferencing

*Up to 90% of an AI-model’s life is spent in inference mode*

- Everyone is aware of the high compute cost (often in millions of dollars) in training large generative models.
- However, the high cost of training is “dwarfed by the expense of inferencing. Each time someone runs an AI model on their computer, or on a mobile phone at the edge, there’s a cost – in kilowatt hours, dollars, and carbon emissions” (linked source in reference).
- Training the model is a one-time investment in compute while inferencing is ongoing.
- Can AI be used to predict the cost of AI workloads for customers?

Source: [https://research.ibm.com/blog/ai-inference-explained](https://research.ibm.com/blog/ai-inference-explained)
High Level Stack Diagram

- Model Servers (KServe/RayServe/ModelMesh)
- Model Serving Runtimes (PyTorch, Triton, TGI)
- Software Accelerator (Python APIs)
- RHEL Driver APIs
- Kernel Drivers
- CPU, GPU, Habana
Software Accelerators for AI Inference

Generally provided by optimizing the software libraries that are used by data scientists.

- **Kernel Level Optimizations**: Such as vectorizations, and effective use of SIMD registers. Intel has done several optimizations with their libraries in this area. OpenShift AI uses Intel OpenVINO which is a software accelerator for inferencing.

- **Graph Optimizations**: Graph optimizations are essentially graph-level transformations including Convolution/ReLU fusion, redundant elimination, and constant folding. Refer to this page on Hugging Face for further details.

- **Quantizations**: Machine learning algorithms commonly store and process numbers that are in single precision. Model quantization implies reducing the numerical precision of the model weights for example from 32-bit float to 8-bit integer. Lower-precision models means better latency performance and energy efficiency but comes at the cost of lower model prediction accuracy.
AI at the Edge Faces Challenges

Too costly and not always practical to send all data back from all edge devices

- Smart storage, filtering and transmission of monitoring and training data
- Optimize for disconnected and intermittent internet situations
- Federated machine learning across edge devices
- Centralized monitoring of models across edge fleets
Corporate Challenges

Will AI-Infused Applications Pass Security Scans?

▸ InfoSec must ensure any data generated or transferred within a company is secured.
  • Audit trails of AI decisions and transparent model lineage are critically important.
  • Blackbox services like ChatGPT are a corporate nightmare.

▸ ProdSec must ensure there are no security vulnerabilities in AI-infused applications
  • Security scans must be possible on models, which often means access to the underlying code and data that created it.
  • Corporations must be able to address any vulnerabilities in a model with urgency.

▸ Do we have the technology or tools to do this?
  • Fairness, bias detection, explainability
Multiple enhancements and iterations to get OpenShift ready for large scale AI training.

Continue to improve and share information and code with Red Hat, and Cloud.
Large-scale distributed jobs slow down due to issues in the infra...

- GPU node failures: **1 every 4 days**
- Top 3 issues: GPU failure or performance issue, network performance issues between GPUs, backend network and service issues e.g. to NetApp
- This is not unique to IBM’s AI Cloud
  - META reports ~2 nodes lost per day while training OPT on Azure
    - 90 re-starts over the course of the training run; actual computation time ~ 30 days, total time to train > 2 months
- *Can we create an “auto-pilot” that steers distributed AI training on OpenShift while handling infra issues?*

**Key lessons:**
- Continuous monitoring and isolation of problem nodes necessary to keep high utilization
- Automation in software that navigate around node failures can help large-scale AI training jobs complete faster
AI Training Auto-pilot

- Auto-pilot is a collection of tools that steer AI training while handling infrastructure issues
  - Pre-flight checks:
    - Validates infrastructure before the start of the job
    - Swaps any sub-optimal components
  - In-flight checks:
    - Workload and system performance is continuously monitored
    - Detect anomaly, decide to continue or stop the job
    - Issue alert to end users
  - Post-flight learning:
    - Improve anomaly detection based on infrastructure validation data

Graphical representation:

```
Scheduler -> Start training job -> Job Running
          |                     | System Metrics (e.g., GPU Util)
          |                     | Application Metrics (e.g., wps)
          | Validate the       |
          | infrastructure     |
          |                   |
          | Yes               |
          | No               |
          | System perf       |
          | Anomaly?          |
          | Yes             |
          | Issue Alert      |
          | No              |
          | Stop the         |
          | Job?            |
          | Yes            |
          |                 |
          |                 |
```

Yes

No

Yes

No

Stop the Job?
Open Source AI at Red Hat