

# Implementing AI on Power11: Introducing the IBM Spyre Adapter

Tim Simon

Harshitha C

Anjil Reddy Chinnapatlolla

Ricardo Contreras

Florian Dahlitz

Abdul Haleem

Blake Hoskinson

Bijay Dev K M

Henrik Mader

Manish Mukul

Arnold Ness

Nnamdi Okore-Affia II

Praveen Kumar Pandey

Manvanthara Puttashankar

Daniel Schenker

Vaibhav Shandilya

David Spurway

Rajalakshmi Srinivasaraghavan

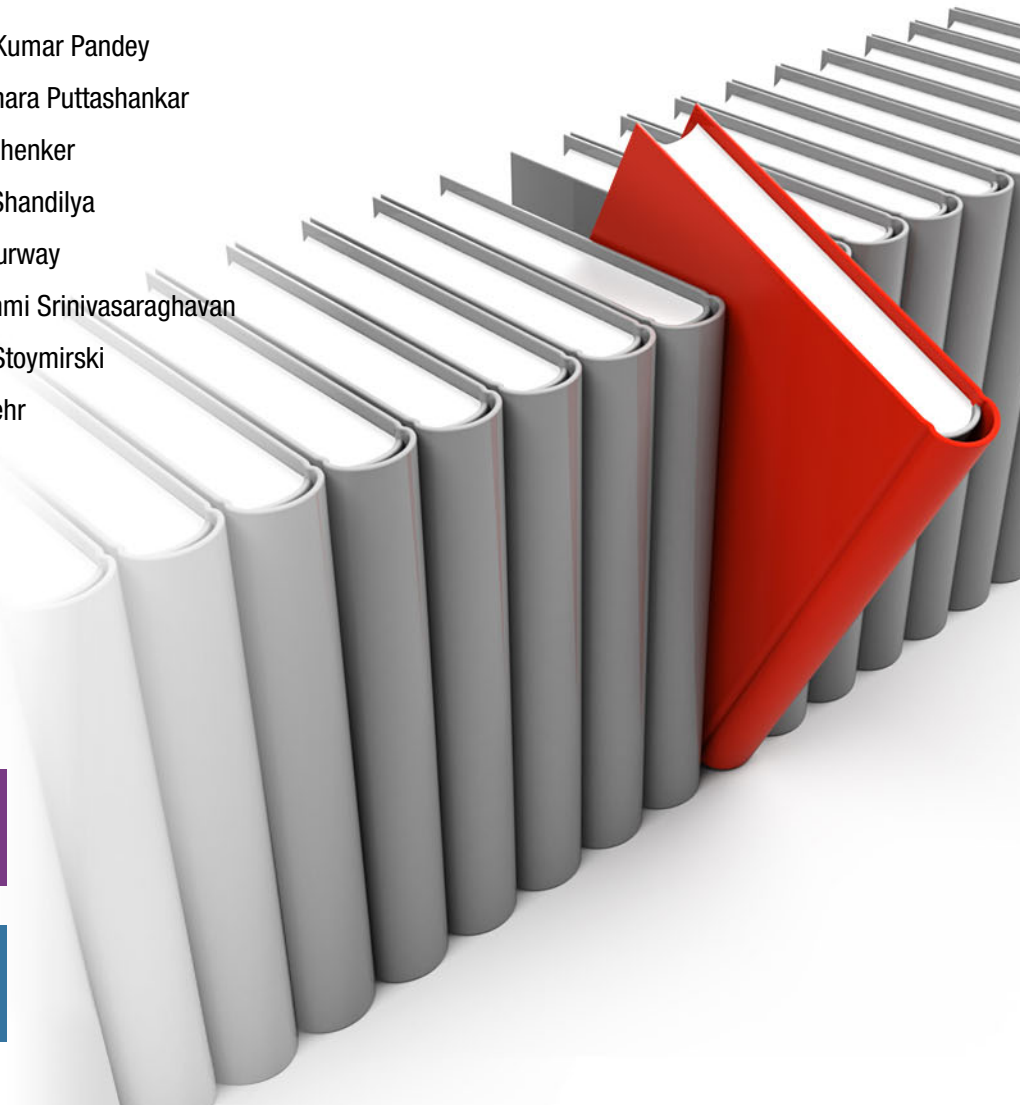
Borislav Stoymirski

Jack Woehr

Henry Vo

**IBM Power**

**Artificial Intelligence**







IBM Redbooks

**Implementing AI on Power11**

January 2026

**Note:** Before using this information and the product it supports, read the information in “Notices” on page iii.

**First Edition (January 2026)**

This edition applies to  
Spyre (GA1) and the IBM Power11 servers  
Red Hat Enterprise Linux (RHEL) version 9.6 or later  
Red Hat AI Inference Server 3.1

© Copyright International Business Machines Corporation 2026. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

# Trademarks

IBM, the IBM logo, and [ibm.com](https://www.ibm.com/legal/copytrade.shtml) are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at <https://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

AIX®	IBM Security®	Redbooks (logo)  ®
Db2®	IBM Spyre™	Spyre™
Granite®	IBM watsonx®	System z®
Guardium®	IBM Z®	watsonx®
IBM®	Orchestrate®	watsonx Orchestrate®
IBM Cloud®	Power9®	watsonx.ai®
IBM Cloud Pak®	PowerVM®	z/VM®
IBM Cloud Satellite®	QRadar®	
IBM Research®	Redbooks®	

The following terms are trademarks of other companies:

Evolution, are trademarks or registered trademarks of Kenexa, an IBM Company.

Intel, Intel Xeon, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Microsoft, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Ansible, Fedora, OpenShift, Red Hat, are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

VMware, and the VMware logo are registered trademarks or trademarks of VMware, Inc. or its subsidiaries in the United States and/or other jurisdictions.

Other company, product, or service names may be trademarks or service marks of others.

# Contents

<b>Notices</b> .....	iii
Trademarks .....	iv
<b>Preface</b> .....	ix
Authors .....	ix
Now you can become a published author, too! .....	xiii
Comments welcome .....	xiii
Stay connected to IBM Redbooks .....	xiii
<b>Chapter 1. Introduction to AI on IBM Power11</b> .....	1
1.1 Artificial Intelligence and IBM Power11 .....	2
1.2 Defining AI and understanding its limitations .....	2
1.2.1 Understanding AI techniques .....	3
1.2.2 Misconceptions about AI .....	4
1.3 Overview of AI trends in enterprise computing .....	4
1.4 Power11 as a strategic platform for AI .....	5
1.4.1 Power11 is a trusted enterprise server .....	6
1.4.2 Hardware assisted AI Features in Power11 .....	6
1.4.3 IBM Power: Transforming Business Through Seamless AI Integration .....	9
1.4.4 AI Services: Turnkey AI .....	10
1.5 Comparison with previous generations .....	11
1.5.1 POWER9 .....	12
1.5.2 Power10 .....	13
1.6 Summary .....	14
<b>Chapter 2. Use Cases and Industry Applications</b> .....	15
2.1 AI Deployment Patterns and Use Cases .....	16
2.1.1 Enterprise AI-ready deployment pattern on Power systems .....	16
2.1.2 Traditional AI/ML on Power Systems – on-premises AI/ML deployment pattern ..	19
2.1.3 Hybrid Agentic AI with IBM watsonx Orchestrate and On-Premises Power .....	22
2.1.4 Pre-built AI use cases with IBM Spyre for Power .....	24
2.2 AI for industries .....	26
2.2.1 Banking and Finance .....	26
2.2.2 Government .....	28
2.2.3 Manufacturing .....	29
2.2.4 Healthcare .....	30
2.2.5 Airlines .....	32
2.2.6 Retail .....	33
2.2.7 Agriculture .....	34
2.3 Cross industry AI .....	35
2.3.1 AI in IT Operations .....	35
2.3.2 AI DevOps .....	40
2.3.3 Real-time inference .....	41
<b>Chapter 3. On-Chip Acceleration</b> .....	47
3.1 Overview of AI acceleration features .....	48
3.2 AI Acceleration on IBM Power11: MMA and SIMD Optimization .....	48
3.3 Training of Machine Learning Models .....	49
3.3.1 The Training Process .....	49

3.3.2	Computational Complexity and Hardware Requirements	50
3.3.3	Hardware and Energy Considerations	51
3.3.4	Summary	51
<b>Chapter 4.</b>	<b>IBM Spyre</b>	<b>53</b>
4.1	Introduction to IBM Spyre	54
4.1.1	Role in AI workload orchestration and optimization	55
4.2	Turnkey AI with Spyre	55
4.2.1	Architecture and Design Principles	55
4.2.2	Integration with IBM Spyre Accelerator for Power	56
4.2.3	Supported AI Use Cases	56
4.2.4	Deployment and Operations	57
4.3	Spyre adapter design	58
4.4	Power Implementation	59
4.4.1	Physical implementation	61
4.4.2	Expansion Drawer support	63
4.5	Examples and supported workloads	69
4.6	Understanding LLM Performance	71
4.7	Hybrid Compute - Spyre and CPU	72
4.8	Futures	73
<b>Chapter 5.</b>	<b>AI Workload Optimization on Power11</b>	<b>75</b>
5.1	Memory bandwidth and cache utilization	76
5.2	NUMA-aware scheduling and resource allocation	76
5.3	Enhancements in Python performance on Power11	78
5.3.1	Power11 Hardware Enhancement	78
5.3.2	Compiler Toolchain and Library-Level Enhancements	79
5.3.3	Python Interpreter and Runtime Enhancements	79
5.3.4	Workload Validation	80
5.4	Support for AI/ML frameworks	81
5.4.1	Overview of Frameworks	81
5.5	Compatibility with OpenBLAS, NumPy, and other scientific libraries	83
5.5.1	Inspecting the OpenBLAS Configuration:	83
5.5.2	Inspecting the NumPy Configuration.	84
5.6	Developer productivity and tooling improvements	89
<b>Chapter 6.</b>	<b>AI and Security</b>	<b>91</b>
6.1	Secure model deployment	92
6.2	Data privacy and encryption in AI pipelines	94
6.2.1	Data Privacy Threats in ML Systems	95
6.2.2	Key Measures for Protecting ML Systems	95
6.3	Security and AI on IBM Power	96
6.3.1	Why Security is important	97
6.3.2	The IBM Framework for Securing Generative AI	97
6.3.3	IBM Power resists Cyber Attacks and can recover fast	100
6.4	Securing AI on IBM Power	100
6.4.1	PowerSC: Security for Hybrid Cloud and AI	101
6.4.2	Expanding Security with AI on IBM Power: Partner Ecosystem	101
6.5	Controlling data and models in your AI ecosystem	102
6.5.1	Keep control of your data with AI on IBM Power.	102
6.5.2	Keeping control of the model with AI on IBM Power.	104
<b>Chapter 7.</b>	<b>Tools, Frameworks, and Ecosystem</b>	<b>107</b>
7.1	Power Inference Microservice (PIM)	108

7.1.1 PIM Architecture . . . . .	108
7.1.2 Creating a Bootable Container Image to run on IBM Power . . . . .	111
7.1.3 Using the Python Ecosystem on IBM Power . . . . .	111
7.2 IBM Watsonx and AI APIs . . . . .	112
7.3 Open-source tools and libraries . . . . .	113
7.3.1 IBM i . . . . .	115
7.4 Access to packages and high level installation . . . . .	116
7.4.1 Installing Container Tooling and Utilities on RHEL 9 . . . . .	117
7.4.2 Preparing the host environment . . . . .	118
7.4.3 RHAIS and API shape . . . . .	120
7.4.4 Obtaining and managing models . . . . .	120
7.4.5 Environment variables and tuning . . . . .	121
7.4.6 Running RHAIS with Podman (examples) . . . . .	121
7.4.7 Validating the server . . . . .	124
7.4.8 Troubleshooting and best practices . . . . .	124
7.5 Other tools . . . . .	125
7.6 Using the Wallaroo AI Starter Kit . . . . .	126
<b>Related publications . . . . .</b>	<b>127</b>
IBM Redbooks . . . . .	127
Online resources . . . . .	127
Help from IBM . . . . .	127



# Preface

This IBM Redbooks publication *Implementing AI on Power11: Introducing the Spyre™* presents IBM's next-generation platform for enterprise AI, showing how Power11 strengthens on-chip acceleration, system design, and hybrid AI deployment. It outlines key AI concepts and industry trends, emphasizing the need for secure, high-performance inference close to mission-critical data.

A major highlight is IBM Spyre™, a new low-power PCIe accelerator designed for scalable AI inference. With a highly parallel architecture and support for modern AI numerics, Spyre enables efficient execution of LLMs, embeddings, RAG, and other enterprise AI workloads. Integrated with Red Hat AI and vLLM, it delivers on-premises AI with predictable performance and data-sovereign control.

The publication also addresses end-to-end lifecycle requirements—security, governance, MLOps, and industry use cases—showing how Power11 and Spyre support regulated, data-intensive environments. Combined with IBM's broader software and open-source ecosystem, the platform provides a complete blueprint for deploying enterprise-grade AI on IBM Power.

## Authors

This book was produced by a team of specialists from around the world working at IBM Redbooks, Remote Center.

**Tim Simon** is a Redbooks® Project Leader in Tulsa, Oklahoma, USA. He has over 40 years of experience with IBM® primarily in a technical sales role working with customers to help them create IBM solutions to solve their business problems. He holds a BS degree in Math from Towson University in Maryland. He has worked with many IBM products and has extensive experience creating customer solutions using IBM Power, IBM Storage, and IBM System z® throughout his career.

**Harshitha C** is a Performance Analyst at IBM India System Development Labs, specializing in end-to-end performance characterization and optimization of enterprise workloads on IBM Power architectures. Her work focuses on full-stack performance engineering for large-scale banking applications, involving deep-dive analysis across the microarchitecture, runtime, middleware, and application layers to identify efficiency gaps and drive throughput, latency, and scalability improvements on the Power platform. Harshitha holds an MS by Research in Computer Science from IIT Palakkad, where her research focused on post-silicon validation workflows, including functional verification, performance correlation with pre-silicon models, and architecture-level anomaly detection using trace analysis. She also holds a B.Tech in Computer Science from the Cochin University Of Science and Technology. Her professional interests span high-performance computing, microarchitecture-aware software optimization, workload characterization, and systems performance tuning. She is passionate about bridging architecture-level insights with application-level behavior to build highly efficient, performance-portable systems on POWER.

**Anjil Reddy Chinnapatlolla** is a Technical Lead at IBM Power Systems, based in IBM India, with over 19 years of experience in systems software and performance engineering. He leads the AI Libraries development team at IBM, focused on accelerating AI workload performance on IBM Power platforms. His work focuses on unlocking CPU- and accelerator-based AI performance through low-level kernel optimization, quantization, and hardware feature enablement. His work

centers on full-stack optimization across hardware, system software, and AI frameworks to unlock platform capabilities. He collaborates closely with architecture, research, and product teams to bring hardware innovations into production environments. His efforts help improve performance, efficiency, and scalability for enterprise AI workloads. Anjil is passionate about bridging system architecture with practical customer deployments.

**Ricardo Contreras** is an IBM Power Brand Technical Specialist based in Chile. With six years at IBM and more than 15 years in the IT industry, he designs and modernizes IBM Power platforms for mission-critical workloads—covering AIX®, open-source databases and AI workloads (ML & LLM) on Linux—using hybrid architectures across on-premises, IBM Cloud®. He architects and implements solutions to stringent RTO/RPO objectives, leveraging IBM Power and cloud platforms to ensure resilience, scalability, and performance. Previously, he was a LinuxONE mainframe specialist for South America and a high-performance computing engineer at Chile's National Laboratory for High Performance Computing (NLHPC), supporting the scientific community and operating HPC environments.

**Florian Dahlitz** is an AI on IBM Power Technical Specialist in Germany. He has eight years of experience in software engineering and focuses on AI for the past three years. He holds a master's degree in computer science from the Karlsruhe Institute of Technology (KIT). He has delivered numerous presentations about AI on IBM Power and helps open-source projects support the ppc64le architecture.

**Abdul Haleem** is a senior Linux on Power Test Engineer at IBM India Systems Development Lab with over 15 years of experience in Linux testing, automation, and enterprise system validation across IBM Power and IBM System z platforms. He brings deep expertise in functional verification, I/O driver testing, and CI/CD infrastructure design for IBM hardware, enabling seamless integration and delivery pipelines for mission-critical systems. Currently leading the IBM Spyre Functional Test team, Abdul drives AI-powered test automation strategies for IBM Power platforms, ensuring reliability, scalability, and continuous improvement in enterprise solutions. A passionate advocate for Linux and open source, he actively contributes to and maintains key Linux test automation projects, fostering innovation and code quality across the ecosystem.

**Blake Hoskinson** MBA BS is currently a technical sales leader in the USA. He has over 30 years of experience in enterprise IT, with a background in IBM Power, PowerVS, IBM Storage, and business resiliency. Blake has led technical sales and service delivery teams, founded and operated a successful consulting firm, architected and implemented solutions, and played a key role in hundreds of system implementations. His expertise spans solution architecture, infrastructure modernization, and customer engagement across diverse industries.

**Bijay Dev K M** is a Hardware Developer at IBM India Pvt. Ltd. He has over seven years of experience in the information technology industry. He holds a Master's degree in Computer Science and Engineering from the National Institute of Technology Surathkal, India. His areas of interest include artificial intelligence and machine learning, AI accelerators, computer networks, and cloud computing.

**Henrik Mader** is an AI on IBM Power Subject Matter Expert based in Germany. He holds a B.Sc. in Business Informatics from the Technical University of Berlin with a specialization in Data Science, as well as an M.Sc. in Information Systems from the Technical University of Munich, focusing on artificial intelligence. Henrik has been working on AI on IBM Power for the past three years, building technical assets, contributing to the open-source community, and supporting enterprise clients in deploying AI workloads and defining their AI strategy on Power systems. His work ranges from implementing modern AI architectures to optimizing LLM and ML workloads for on-premises environments. Before joining IBM, Henrik worked as a Data Engineer for a DAX-40 company. He completed his master's thesis with IBM Research®, where he focused on ensemble methods and uncertainty estimation in machine learning.

**Manish Mukul** is a Senior Advisory Software Engineer at IBM India Pvt. Ltd., bringing over 17 years of industry experience in developing advanced hardware validation tools for IBM Power systems. Renowned for his ability to bridge hardware and software disciplines, Manish architects and delivers scalable, high-performance validation frameworks that ensure the reliability, functional integrity, and performance of next-generation IBM Power servers. In his current leadership role, he works closely with silicon design, firmware, and test engineering teams to build robust, reusable validation solutions packaged as an intuitive, user-level tool suite. His work enables seamless validation workflows across development teams and contributes directly to the quality and readiness of emerging Power platforms. Manish holds a Bachelor of Engineering degree in Information Technology from Pune University. He is driven by a passion for engineering excellence, automation, and staying at the forefront of next-generation server and systems design.

**Arnold Ness** P.Eng is IBM Canada's National Hybrid Cloud Technology Leader, driving AI innovation and workload optimization for enterprises. With decades of experience at IBM and Ciena, he helps organizations harness hybrid cloud, AI, and quantum safe cybersecurity to gain competitive advantage. Arnold holds an MBA in Information Technology Management from Royal Roads University, an Electrical Engineering degree from the University of Alberta, and completed IBM's Client Executive Program at Harvard. He specializes in solution design and delivery across x86, IBM Power, IBM Z®, and cloud platforms. In 2022, Arnold received the Lou V. Gerstner Jr. Award for Client Excellence for delivering hybrid-cloud solutions with the Government of Canada.

**Nnamdi Okore-Affia II** is a Technology Expert Labs Delivery Consultant in the United Kingdom. He has 7 years of experience with Linux on Power Systems. He holds a degree in Computer Science from Northumbria University. His areas of expertise include Red Hat Enterprise Linux Systems Administration and AI on Power Systems. He has worked extensively on Deep Learning and the impact of AI.

**Praveen Kumar Pandey** is a Senior Software Engineer at IBM with more than 15 years of experience in functional verification and performance engineering across IBM Power and IBM System z platforms. He specializes in designing and implementing CI/CD infrastructure for IBM hardware, enabling streamlined integration and delivery pipelines that support enterprise-class systems development. Praveen is an active contributor and maintainer within several open-source communities, including the Linux kernel and Linux test automation projects. His contributions help drive innovation while ensuring the reliability and quality of mission-critical components. He is the architect of the Spyre CI/CD framework on IBM Power11, a key solution that accelerates development, validation, and continuous improvement workflows for next-generation IBM systems. In his current role, Praveen is responsible for ensuring the functional quality and reliability of supported Linux distributions on IBM Power servers. He applies automation, rigorous testing methodologies, and iterative enhancement practices to deliver robust and resilient enterprise solutions. Praveen holds a Bachelor's degree in Computer Science and Engineering from the Govind Ballabh Pant Institute of Engineering & Technology in Uttarakhand, India.

**Manvanthara Puttashankar** is a Linux on Power QA Architect at the IBM India Systems Development Lab in Bangalore. With over 22 years of experience working on operating systems (Linux, AIX, IBM z/VM®), software development (firmware and kernel), server platforms (IBM Power, IBM System Z, and other enterprise server architectures), I/O drivers (network and storage across multiple vendor ecosystems), and Spyre (AI), he brings deep technical expertise across the stack. In his extended role as an AI Lead, he has been driving AI transformation across various IBM platforms. His contributions have been recognized with several awards, including the prestigious OTAA and Tech2025 honors. He is a strong customer advocate for IBM Power and IBM System Z platforms and has been closely collaborating with clients and vendors across multiple geographies.

**Daniel Schenker** is a Software Performance Analyst and AI/Linux Toolchain Developer based in Texas, with three years of experience at IBM. He holds a Bachelor's degree in Computer Engineering from the Rochester Institute of Technology. His work focuses on the software

enablement for the Spyre Accelerator cards, where he develops, maintains, and validates end-to-end software stacks supporting AI workloads on Linux systems. His areas of expertise include AI performance benchmarking and analysis, containerized deployment workflows, CI/CD integration, and the development of automation and tooling for executing and measuring AI workloads. He also contributes to and manages both open-source and internal GitHub projects that support scalable, reliable, and reproducible AI software deployments.

**Vaibhav Shandilya** is a seasoned technology leader with over 28 years of industry experience and more than 17 years at IBM. He currently leads APAC Client Engineering pilots and proof-of-concept initiatives, helping clients innovate with hybrid cloud, AI/ML, and Generative AI solutions. Throughout his career, Vaibhav has held key leadership roles, including managing worldwide IBM Power Techline and Power assurance teams, where he drove technical excellence and partner enablement across global markets. His work has influenced multi million-dollar sales opportunities by aligning technology strategies with business goals and mentoring cross-functional teams. Vaibhav's expertise spans cloud architecture, IBM Cloud Paks, watsonx@.ai, watsonx.data, AI/ML, and Generative AI. He holds a Master's degree in Computer Science and has published numerous technical blogs, sharing insights on emerging technologies and best practices.

**David Spurway** is the IBM Power AI and Security Principal, which sees him running around Europe, the Middle East and Africa, talking about how our servers and solutions can help customers run mission critical applications reliably and for less money. He talks all over the world about IBM Power, sometimes in person, regularly in virtual events, doing presentations which are tailored to the audience to entertain and inform. He works with customers to show them how they can lower their costs, resist Cyber attacks and also help lower their carbon footprints. He also works with customers to show them how they can use AI to modernize and deliver new business functions.

**Rajalakshmi Srinivasaraghavan** is a software performance analyst and AI infrastructure strategist passionate about bridging software and hardware for optimal system performance. At IBM, she works across architecture, performance analysis, and open-source enablement to bring forward-looking capabilities into today's software ecosystem. Her work seamlessly integrates software architecture with low-level hardware features—particularly for AI acceleration and compute-intensive systems. Through her open-source contributions, she has delivered measurable performance gains and fostered broader community adoption on IBM POWER, leaving a lasting impact on the industry.

**Borislav Stoymirski** is a Hardware Product Engineer for IBM Power Servers at IBM Bulgaria, specializing in diagnosing and resolving complex hardware and software challenges across the IBM Power ecosystem, including IBM AIX, VIOS, HMC, IBM i, PowerVC, PowerVM® NovaLink, PowerVM, Linux on Power, and Red Hat OpenShift. Since joining IBM in 2015, he has delivered reactive, proactive, preventative, and cognitive support to clients worldwide, driving system reliability and performance improvements across enterprise environments. Borislav earned a Master's degree in Computer and Software Engineering from the Technical University of Sofia, Bulgaria, a Master's degree in Transport Machinery and Technologies from the Technical University of Sofia, Bulgaria, and another Master's degree in Ecology and Environmental Protection from the University of Chemical Technology and Metallurgy, Bulgaria. He is also an active contributor to the technical community, having authored multiple publications and led training programs both within IBM and externally. His research and professional interests include Bioinformatics, Green Blockchain, Artificial Intelligence, Machine and Deep Learning, Cybersecurity, IoT and Edge Computing, Cloud, and Quantum Computing, with a strong focus on sustainability and intelligent infrastructure.

**Jack Woehr** is an IBM Champion for Power and Quantum based in Colorado, USA. As an independent consultant, his primary focus is IBM i modernization. With more than 40 years of experience in software development, he spent two decades as a contributing editor for Dr. Dobb's Journal. A frequent speaker at industry conferences, he has delivered numerous presentations on

Quantum Computing and has traveled internationally to speak on Large Language Model (LLM) AI.

**Henry Vo** is an IBM Redbooks Project Leader with more than 10 years of experience in IBM. He has technical expertise in business problem solving, risk/root-cause analyze, and writing technical plans for business. He has had multiple roles in IBM such as Project management, ST/FT/ETE Test, Back End Developer, DOL agent for NY and is certified in IBM zOS Mainframe Practices, IBM Z System programming, Agile, and Telecommunication Development Jumpstart. Henry holds a Master of MIS (Management Information System) from the University of Texas at Dallas since 2012.

Thanks to the following people for their contributions to this project:

Andrew Laidlaw, Senior Partner Technical Specialist - Power  
IBM UK

## Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an IBM Redbooks residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- ▶ Use the online **Contact us** review Redbooks form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- ▶ Send your comments in an email to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- ▶ Mail your comments to:

IBM Corporation, IBM Redbooks  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

## Stay connected to IBM Redbooks

- ▶ Find us on LinkedIn:

<https://www.linkedin.com/groups/2130806>

- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

<https://www.redbooks.ibm.com/subscribe>

- ▶ Stay current on recent Redbooks publications with RSS Feeds:

<https://www.redbooks.ibm.com/rss.html>



# Introduction to AI on IBM Power11

This chapter introduces the evolving role of Artificial Intelligence (AI) in enterprise computing and positions IBM Power11 as a strategic platform for AI deployment. It outlines the foundational concepts of AI, including machine learning, deep learning, and natural language processing, and explains how these technologies are transforming industries through automation, decision-making, and personalization. The chapter also addresses common misconceptions about AI, such as its supposed autonomy or human-like reasoning, and emphasizes the importance of responsible deployment. It categorizes AI into types—from narrow AI to speculative self-aware systems—and highlights current enterprise applications like predictive analytics, fraud detection, and customer engagement.

The chapter then explores how IBM Power11 advances AI capabilities beyond previous generations. Compared to POWER9 and Power10, Power11 offers enhanced performance, broader AI precision support, and a mature ecosystem. Key innovations include on-chip Matrix Multiply Assist (MMA), expanded SIMD (Single Instruction, Multiple Data) acceleration for vectorized numerical workloads, the IBM Spyre off-chip accelerator. These features enable high-throughput inferencing and generative AI workloads with reduced latency and energy consumption. Power11 also delivers enterprise-grade reliability, quantum-safe security, and hybrid cloud flexibility, making it ideal for deploying AI where data resides. With integration into Red Hat OpenShift AI and IBM watsonx, Power11 supports turnkey AI lifecycle management, positioning it as a trusted platform for mission-critical AI applications.

This chapter includes the following topics:

- ▶ Summary
- ▶ Defining AI and understanding its limitations
- ▶ Overview of AI trends in enterprise computing
- ▶ Power11 as a strategic platform for AI
- ▶ Comparison with previous generations
- ▶ Summary

## 1.1 Artificial Intelligence and IBM Power11

Artificial Intelligence (AI) is the field of computer science focused on creating systems that can perform tasks that typically require human intelligence. These tasks include learning from data, reasoning, problem-solving, understanding natural language, and making decisions.

AI encompasses various techniques such as machine learning (ML), deep learning, and natural language processing (NLP), enabling applications like predictive analytics, computer vision, and generative AI. In enterprise computing, AI is used to automate processes, enhance decision-making, and deliver personalized experiences at scale.

IBM Power11 is designed to accelerate enterprise AI adoption by delivering innovations in performance, security, and scalability. It features advanced AI acceleration through on-chip MMA AI accelerators, expanded SIMD (Single Instruction, Multiple Data) vector processing, and the IBM Spyre off-chip accelerator, providing high throughput for inference and next-generation generative AI workloads. Power11 also offers hybrid cloud flexibility, enabling consistent deployment across edge environments, on-premises infrastructure, and IBM Cloud through Power Virtual Server. This approach allows enterprises to run AI workloads wherever data resides, maintaining compliance, and reducing latency.

In terms of resilience and security, Power11 delivers 99.9999% uptime with zero planned downtime and includes IBM Power Cyber Vault for ransomware detection in under one minute. Quantum-safe cryptography adds protection against future threats. As an enterprise-grade AI platform, Power11 integrates with Red Hat OpenShift AI and IBM watsonx to provide model governance, transparency, and lifecycle management. Co-created with customers, it is built for turnkey AI deployment across mission-critical workloads.

## 1.2 Defining AI and understanding its limitations

Artificial Intelligence (AI) is a broad discipline. Today, the term AI is often used more narrowly to describe the application of Large Language Model (LLM)-based transformers for logic programming and user interaction. In this Redbook, we aim to maintain a broader perspective that includes machine learning, intelligent systems, and other essential components of Artificial Intelligence as a whole.

AI can be categorized into several types, each with distinct capabilities and enterprise applications:

- ▶ **Narrow AI (Weak AI)**

Narrow AI is designed for specific tasks within a limited domain. Enterprise examples include chatbots for customer service, recommendation engines in e-commerce, and fraud detection systems in banking.

- ▶ **General AI (Strong AI)**

General AI would perform a wide range of tasks similar to human intelligence. While still theoretical, enterprises envision its use in autonomous decision-making across multiple business functions, such as supply chain optimization and strategic planning.

- ▶ **Superintelligent AI**

A hypothetical form of AI that surpasses human intelligence in all aspects. In an enterprise context, this could mean fully autonomous corporations capable of innovating and managing operations without human intervention. This remains speculative.

- ▶ **Reactive Machines**  
These systems respond to inputs without memory or learning. Enterprises use them in basic automation, such as rule-based quality checks in manufacturing or simple customer query routing.
- ▶ **Limited Memory AI**  
Most modern AI systems fall into this category. Examples include predictive maintenance in industrial equipment, real-time fraud detection in financial transactions, and autonomous vehicles for logistics.
- ▶ **Theory of Mind AI**  
Still in research, this type would understand emotions and intentions. Enterprises could use it for advanced customer engagement systems that adapt to emotional cues or human-like virtual assistants for HR and employee wellness.
- ▶ **Self-Aware AI**  
A futuristic concept where AI systems have consciousness. In theory, enterprises could leverage this for fully autonomous leadership roles, but this remains purely speculative and raises ethical concerns.

## 1.2.1 Understanding AI techniques

AI uses techniques such as machine learning, deep learning, and natural language processing to enable applications like predictive analytics, computer vision, and generative AI. In enterprise computing, AI plays a critical role in automating processes, improving decision-making, and delivering personalized experiences at scale.

### **Machine Learning (ML)**

Machine Learning is a subset of AI that enables systems to learn patterns from data and improve performance without being explicitly programmed. ML algorithms use statistical techniques to make predictions or decisions based on input data.

Some example uses for ML in enterprise computing are:

- ▶ Predictive analytics for demand forecasting
- ▶ Fraud detection in banking
- ▶ Customer churn prediction

### **Deep Learning (DL)**

Deep Learning is a specialized branch of ML that uses artificial neural networks with multiple layers to model complex patterns in large datasets. It excels in tasks like image recognition, speech processing, and generative AI because it can handle unstructured data at scale.

Some example uses for deep learning are:

- ▶ Computer vision for quality control in manufacturing
- ▶ Autonomous vehicle navigation
- ▶ Advanced recommendation systems.

### **Natural Language Processing (NLP)**

NLP focuses on enabling machines to understand, interpret, and generate human language. It combines linguistics and ML techniques to process text and speech. NLP powers applications like chatbots, sentiment analysis, and language translation.

Enterprise examples for NLP are:

- ▶ AI-driven customer support chatbots
- ▶ Document summarization for legal firms
- ▶ Voice assistants for call centers.

## 1.2.2 Misconceptions about AI

Artificial Intelligence (AI) is transforming industries and reshaping how businesses operate, but misconceptions about what AI can and cannot do remain widespread. These misunderstandings often lead to unrealistic expectations or unnecessary fears, which can hinder effective adoption. From assumptions that AI thinks like humans to beliefs that it can learn without data or operate fully autonomously, these myths obscure the reality of AI's capabilities and limitations.

Here are some common misconceptions about AI:

- ▶ **AI Thinks Like Humans**

Many people believe AI has human-like reasoning or emotions. In reality, AI systems process data using algorithms and statistical models. They do not possess consciousness, intuition, or feelings.

- ▶ **AI Can Learn Without Data**

AI models require large amounts of high-quality data to learn patterns. Without data, they cannot improve or make accurate predictions.

- ▶ **AI Is Always Objective**

AI can inherit biases from the data it is trained on. If the training data is biased, the model's outputs will reflect those biases, making governance and fairness critical.

- ▶ **AI Will Replace All Jobs**

AI automates repetitive tasks and augments human decision-making, but most roles evolve rather than disappear. New jobs emerge in AI development, governance, and maintenance.

- ▶ **AI Understands Context Perfectly**

AI models often struggle with nuanced language, cultural context, or ambiguous scenarios. They rely on patterns, not true comprehension.

- ▶ **Bigger Models Are Always Better**

While large models like LLMs are powerful, they are not always the best solution. Smaller, specialized models can outperform them for domain-specific tasks.

- ▶ **AI Is Fully Autonomous**

Most enterprise AI systems require human oversight for governance, compliance, and ethical decision-making. AI is a tool, not a self-governing entity.

Understanding the truth behind these misconceptions is essential for enterprises to deploy AI responsibly and maximize its value.

## 1.3 Overview of AI trends in enterprise computing

Enthusiasm for modern approaches to Artificial Intelligence has dominated the technological landscape. Early impressions of transformer models and their ability to engage in human-like

dialogue sparked a rapid wave of adoption and significant investment aimed at expanding the scope, capabilities, and market reach of agent-based architectures.

However, this initial excitement has been tempered by a growing recognition of the limitations inherent in these architectures and the challenges encountered when attempting to broaden the functionality and scalability of large language model (LLM)-based agents.

Limitations include

- ▶ Difficulty in preventing exfiltration of data
- ▶ Emergent behavior (unexpected and startling outcomes)
- ▶ Technical, existential, and epistemological challenges to validation of output
- ▶ Nearly unfathomable security and trust issues
- ▶ Disappointments in automatic (“vibe”) coding
- ▶ Often unsatisfactory return on investment (ROI)<sup>1</sup>

Among the checks encountered is the failure of “super-intelligent” models, trained on orders of magnitude more data than the most popular models yet yielding inferior results, or at least disappointing results relative to the investment made in their development.<sup>2,3,4</sup>

The mood overall in the enterprise is, at this writing in Q3 of 2025, one of caution and hesitation, but there is no diminution of expectations regarding the ultimate outcome. Teams of engineers and system programmers explore tools and scenarios and await imminent developments, particularly in the Power architecture audience.

Meanwhile, impressive demonstrations of the genuine capacity of the LLM and Transformer architecture, combined with encouraging advances in deployment patterns, make it clear that despite hype, overexuberance, and the likelihood of an all-too-familiar technology investment bubble, LLMs are here to stay and will revolutionize many aspects of the delivery of goods and services while shaking the foundations of at least the field of Information Technology, and perhaps the foundations of the entire industrial workforce.

Over the past decade and several revisions, IBM has had the foresight to modify the Power architecture incrementally and fundamentally so that Power may serve as a trusted and muscular platform for machine learning, training, and inferencing workloads. The arrival of the Power 11 architecture at maturity for deploying such workloads, and the tools and techniques of this deployment, form the subject matter of this Redbook.

## 1.4 Power11 as a strategic platform for AI

IBM Power11 marks a new milestone in enterprise computing, bringing together industry-leading reliability, advanced security, and deeply integrated AI acceleration to support the mission-critical workloads that modern organizations depend on. As businesses expand their use of AI, hybrid cloud, and data-intensive analytics, Power11 provides a trusted foundation engineered to run core operations with uncompromising resilience while seamlessly extending into next-generation AI and data processing. Building on decades of enterprise leadership, Power11 combines proven IBM i, AIX, and Linux robustness with new

<sup>1</sup> ““ROI of AI”, IBM in collaboration with Lopez Research, December, 2024

<sup>2</sup> “Why Super intelligent AI Isn’t Taking Over Anytime Soon”, WSJ, 2025-06-13

<sup>3</sup> “The Fever Dream of Imminent Super intelligence Is Finally Breaking”, NYT, 2025-09-03

<sup>4</sup> “The Illusion of Thinking: Understanding the Strengths and Limitations of Reasoning Models via the Lens of Problem Complexity”, Apple Computer, 2025

hardware-assisted AI capabilities, enabling organizations to modernize securely and confidently without disrupting the systems that keep their business running.

### **1.4.1 Power11 is a trusted enterprise server**

IBM Power11 represents the next generation of enterprise computing, designed to deliver unmatched reliability, security, and performance for mission-critical workloads. As organizations accelerate their adoption of AI, hybrid cloud, and advanced analytics, Power11 provides a trusted foundation that combines proven enterprise resilience with cutting-edge innovation.

#### **Enterprise-Grade Reliability**

Power11 continues IBM's legacy of delivering systems with industry-leading uptime. With 99.9999% availability, zero planned downtime, and advanced features like dynamic firmware updates, businesses can maintain continuous operations even during maintenance cycles. This level of resilience ensures that critical applications remain available, supporting uninterrupted business processes.

#### **Security Built for the Future**

Security is at the core of Power11's architecture. It incorporates quantum-safe cryptography to protect against emerging threats and includes IBM Power Cyber Vault, which enables ransomware detection in under one minute. These capabilities provide enterprises with confidence that their data and workloads are safeguarded against both current and future attack vectors.

#### **Performance for Modern Workloads**

Power11 is optimized for AI-driven workloads with integrated on-chip MMA accelerators, expanded SIMD (Single Instruction, Multiple Data) vector processing, and the IBM Spyre off-chip accelerator, delivering high throughput for inference and generative AI tasks. Support for multiple precisions of operations ensures efficient model execution, while advanced memory bandwidth and I/O capabilities enable rapid data movement for analytics and AI applications.

#### **Hybrid Cloud Flexibility**

Enterprises can deploy Power11 consistently across on-premises environments, edge locations, and IBM Cloud using Power Virtual Server. This flexibility allows organizations to run workloads where data resides, reducing latency and ensuring compliance with data sovereignty regulations.

#### **Integrated AI and Open Ecosystem**

Power11 integrates seamlessly with Red Hat OpenShift AI and IBM watsonx, providing a robust platform for AI lifecycle management, governance, and transparency. Co-created with customers, this ecosystem supports turnkey AI deployment across mission-critical workloads, enabling enterprises to innovate with confidence.

### **1.4.2 Hardware assisted AI Features in Power11**

AI's rise was fueled by transformer models and their human-like dialogue, driving rapid adoption and investment. Today, focus has shifted to overcoming LLM limitations and optimizing performance. GPUs remain key for training but pose latency and cost challenges for real-time enterprise use.

In contrast, modern enterprise platforms such as IBM Power11 integrate on-chip AI accelerators and specialized hardware like IBM Spyre to deliver high-throughput inference with lower latency and improved energy efficiency. This approach enables organizations to run AI workloads closer to where data resides, reducing dependency on external GPU clusters and supporting hybrid deployments across edge, on-premises, and cloud environments.

## **Matrix Multiply Assist on Power11**

Matrix operations form the mathematical foundation of modern machine learning, powering algorithms that handle large-scale data transformations and computations. These operations are essential for tasks such as training neural networks, performing linear algebra calculations, and optimizing models for inference.

The IBM Power11 CPU builds on this foundation with the Matrix-Multiply Assist (MMA) feature, originally introduced in Power10<sup>5</sup>, to accelerate matrix computations directly in hardware. MMA enables high-performance execution of matrix multiplication, a critical operation in AI workloads, by leveraging specialized instructions that reduce computational overhead.

This capability is complemented by support for Vector Multimedia Extension (VMX) and Vector Scalar Extension (VSX) instructions, which provide optimized vector processing functions. Together, these features allow Power11 to deliver exceptional throughput for machine learning and deep learning tasks, enabling enterprises to run complex AI models efficiently and at scale.

## **SIMD Acceleration on IBM Power11**

IBM Power11 introduces significant enhancements to its SIMD (Single Instruction, Multiple Data) capabilities, extending the vector-processing foundation of the Power ISA to better support modern AI, analytics, and HPC workloads. Like MMA and Spyre, SIMD on Power11 plays a critical role in accelerating the data-parallel computations that dominate enterprise AI pipelines.

Power11 builds on and extends IBM's long-standing vector architecture by incorporating a "big, wide SIMD engine" within each core, designed to provide higher per-cycle throughput for vector and tensor-friendly operations. This represents an architectural uplift over Power10, emphasizing increased parallelism and improved execution bandwidth for vector instructions.

This wide SIMD design ensures that workloads involving dense linear algebra, signal processing, encryption, and real-time analytics can take full advantage of Power11's increased clock speeds and improved memory subsystem.

## ***VSX and Power ISA Vector Enhancements***

Power11 continues to support and enhance the Vector-Scalar Extension (VSX), the evolution of IBM's SIMD model historically rooted in AltiVec/VMX. VSX on Power11 provides a unified set of 128-bit vector registers and instructions used for floating-point, integer, and logical operations. This design enables broad compiler and intrinsic support, making it straightforward to port SIMD-optimized routines from other architectures such as x86 AVX or ARM NEON.

Key benefits of the mature VSX ecosystem include:

- ▶ High portability for SIMD-accelerated libraries (Eigen, OpenBLAS, FFT implementations)
- ▶ Efficient execution of vector loops, particularly when combined with compiler auto-vectorization

---

<sup>5</sup> [MMA in IBM Power10 processor](#), Sridhar Venkat, blog article on IBM TechXchange, 2022-08-29

- ▶ Support for mixed integer and floating-point vector workloads—highly relevant for AI pre- and post-processing operations

Complementing MMA's matrix-centric acceleration, VSX SIMD continues to serve as the general-purpose vector engine for a broad spectrum of enterprise workloads.

### ***Integration with Power11's Memory and I/O Bandwidth***

Power11's SIMD performance is further amplified by substantial memory subsystem improvements, including a 32-channel DDR5 OMI architecture offering up to four times the bandwidth of prior generations. This provides SIMD workloads with the sustained memory throughput they require for vector-dense operations, especially when processing large tensors, streaming datasets, or real-time analytics pipelines. [wccftch.com]

High-speed memory and expanded SIMD width work synergistically to reduce pipeline stalls and improve utilization of vector execution units.

### ***Role of SIMD in the Power11 AI Stack***

Within Power11's AI-acceleration stack:

- ▶ MMA provides high-efficiency matrix math for inferencing and generative AI.
- ▶ SIMD/VSX accelerates vector math used in activation functions, normalization, quantization/dequantization, and data layout transformations.
- ▶ Spyre handles large-scale off-chip AI acceleration for high-end generative workloads.

SIMD thus occupies the essential middle ground: a flexible, general-purpose acceleration layer that ensures high performance even for AI tasks not offloaded to MMA or Spyre.

Power11's enhanced SIMD capabilities benefit:

- ▶ AI frameworks (PyTorch, TensorFlow) via optimized back-end libraries
- ▶ Enterprise analytics through faster vector scanning, filtering, and transformation
- ▶ Cryptographic operations such as hashing and block-cipher acceleration
- ▶ High-performance scientific workloads where vector operations dominate compute kernels

These improvements make SIMD an integral part of Power11's value proposition for enterprises seeking scalable, CPU-centric AI performance.

## **Spyre Accelerator**

The IBM Spyre Accelerator, now integrated into Power11 systems, represents a major leap forward in AI hardware acceleration for enterprise workloads. Designed to support generative and agentic AI, Spyre enables low-latency inferencing and real-time responsiveness, which are critical for modern AI applications that rely on dynamic decision-making and multi-model processing.

Spyre is a PCIe-attached system-on-a-chip, built using 5nm node technology, and features 32 individual accelerator cores and 25.6 billion transistors. Each card consumes 75 watts and can be clustered, up to 16 cards in a Power11 system, to scale AI capabilities significantly[1]. This clustering allows Power11 servers to handle increasingly complex AI workloads, including large language models (LLMs), without compromising on throughput or system resilience.

Unlike previous generations that relied heavily on external GPUs, Spyre brings on-premises AI acceleration directly into the Power11 architecture. This integration helps enterprises maintain data locality, enhancing security and compliance by avoiding the need to offload sensitive data to external cloud environments.

Spyre also supports multi-model AI processing, which allows systems to use different types of models in tandem to improve inference accuracy. For example, initial predictions might be handled by the Power11 processor, while more complex or uncertain cases are escalated to Spyre for deeper analysis.

In summary, the Spyre Accelerator transforms Power11 into a high-performance AI platform, capable of executing demanding AI workloads with greater efficiency, scalability, and security. It reflects IBM's commitment to delivering enterprise-grade AI infrastructure that meets the evolving needs of modern data-driven organizations.

### 1.4.3 IBM Power: Transforming Business Through Seamless AI Integration

IBM Power servers are redefining enterprise computing by embedding AI directly into the heart of business operations. Whether integrating AI into transactional systems, scaling generative models, or deploying applications across hybrid environments, Power delivers the performance, security, and flexibility needed to support innovation without disruption.

#### AI Where Your Data Lives

In today's distributed data landscape, spanning on-premises, cloud, and edge environments, AI must be both intelligent and strategically placed. IBM Power brings AI inferencing directly to where data is generated and resides, eliminating the need to move sensitive information across networks or rely on costly external GPUs. This approach reduces latency, enhances security, and lowers operational costs. With the IBM Spyre Accelerator and IBM watsonx.data on Power, organizations can unlock faster insights by combining a modern data lakehouse with advanced off-chip acceleration, bridging trusted data with powerful inferencing capabilities.

#### Security for AI Workloads, 24x7

AI workloads are valuable assets and must be protected from evolving threats. IBM Power servers offer security integrated across the entire stack, including transparent memory encryption that safeguards insights without compromising performance. For demanding tasks like generative AI, Power enables scalable inferencing with confidence. The platform also features IBM Power Cyber Vault, which provides ransomware threat detection in under one minute<sup>6</sup>, and delivers up to 99.9999% uptime<sup>7</sup>, ensuring AI systems remain resilient and continuously available.

#### Hybrid Flexibility Without Friction

Deploying AI across hybrid environments requires seamless integration. IBM Power servers support optimized enterprise software for a frictionless hybrid cloud experience, allowing workloads to move easily between on-premises systems and IBM Power Virtual Server. Whether training models in the cloud or running inference locally, Power provides the flexibility to adapt. With support for more than 130 open-source AI tools and packages, teams can build, deploy, and scale AI solutions without encountering compatibility issues.

#### Sustainable Performance on Demand

Meeting sustainability goals while running AI workloads is no longer a contradiction. IBM Power11 delivers twice the performance per watt compared to x86-based servers, enabling organizations to reduce energy consumption without sacrificing capability.<sup>8</sup> The new Energy Efficient mode on Power11 servers offers up to 28% better energy efficiency compared to

<sup>6</sup> This guarantee covers only the displaying of an alert in less than one minute.

<sup>7</sup> Based upon unplanned downtime of a single Power E1180 system as calculated in the Power11 ProcessorBased Systems RAS (see section: 99.9999% uptime) <https://www.ibm.com/downloads/documents/usen/10a99803d9afd776>

Maximum Performance mode<sup>9</sup>, helping enterprises align AI operations with environmental objectives.

### Acceleration Without Compromise

IBM Power servers are engineered to run AI at scale and in real time, without disrupting existing workflows. With high parallelism, expansive memory, and integrated on-chip acceleration, Power enables AI to be embedded directly into business processes. Data scientists can run workloads without refactoring code, streamlining deployment. For large language models, Power S1022 processes up to 42% more batch queries per second than comparable x86 servers under peak load<sup>10</sup>, while maintaining sub-second inferencing latency. This translates to faster insights, smoother operations, and real-time responsiveness.

## 1.4.4 AI Services: Turnkey AI

IBM Spyre for Power is an integrated portfolio of AI software products optimized for the IBM Spyre Accelerator for Power infrastructure. The portfolio covers proven adoption patterns and a catalog with pre-built AI services (IBM Open-Source AI Foundation for Power), an inferencing platform (Red Hat AI Inference Server on IBM Power) integrated and optimized with accelerated infrastructure (IBM Spyre Enablement Stack for Power), and updates the enablement stack (IBM Spyre Stack Updates for Power).

This optimized full-stack portfolio offers a powerful combination of easy-to-install AI services, rapid deployment to production capabilities, and acceleration that enables turnkey AI for enterprises. Turnkey AI enables rapid transition from pilot to production, integrates easily and seamlessly with existing enterprise workflows (such as reflected in ERP and IT systems), and ensures data sovereignty through a trusted, self-contained on-premises infrastructure. AI Services are part of the [IBM Open-Source AI Foundation for Power](#) and are designed for IBM Power11 systems. They provide adoption patterns, pre-built service components, and packaged capabilities that integrate with inference platforms such as Red Hat AI Inference Server. AI Services are also optimized for the IBM Spyre Accelerator for Power, which ensures consistently high performance across the Power11 AI infrastructure.

AI Services are created for simplicity, speed, and turnkey deployment. They allow enterprises to stand up AI capabilities quickly and install them with minimal effort. This helps organizations accelerate digital transformation, streamline business processes, and improve operational productivity. Each AI Service is a fully self-contained deployment unit with advanced inference capabilities. The portfolio supports a wide range of models, including large language models, embedding models, and re-ranking models. With IBM Spyre for Power and flexible consumption options for LLM inferencing, organizations can scale AI workloads efficiently while adopting a turnkey AI approach that reduces complexity and shortens time to value. More detail on Turnkey AI with Spyre is provided in section 4.2, “Turnkey AI with Spyre” on page 55.

<sup>8</sup> Based upon Quantitative Performance Index (QPI) data as of 15 May 2025 from IDC available at <https://www.idc.com/about/qpi> and utilization. IBM Power E1150 (4x30c Power11 at 3.0-4.1 GHz) QPI of 241,000E versus HPE Compute Scale-up Server 3200 (4x60-core Intel cores at 1.9 GHz) QPI of 208,898 and utilizations of 75% for E1150 based on IBM Power Performance Utilization Guarantee and 40% for x86.

<sup>9</sup> Based upon IBM measurements of performance per watt on servers comparing Maximum Performance Mode to Energy-Efficient Mode while running compute-, disk-, and memory-based workloads on Power11 systems with fully configured sockets and memory as follows: E1180 with 4x10c / 64x64GB DDIMM, E1150 with 4x16c / 64x32GB DDIMM, S1124 with 2x16c / 32x32GB DDIMM, S1122 with 2x16c / 32x32GB DDIMM

<sup>10</sup> Comparison based on IBM internal testing of question and answer inferencing using PrimeQA model (<https://github.com/primeqa>, based on Dr. Decr and CoBERT models). Results valid as of Aug 22, 2023, and conducted under laboratory conditions, individual results can vary based on workload size, use of storage subsystems and other conditions. Comparison is based on total throughput in score (inferences) per second on IBM Power S1022 (1x20-core/512GB) running SMT 8 versus Intel Xeon Platinum 8468V-based (1x48-core/512GB) systems

## 1.5 Comparison with previous generations

The IBM Power11 platform marks the next step in IBM's evolution of enterprise AI on Power Systems. POWER9 laid the groundwork with robust support for AI workloads, high memory bandwidth, and GPU integration. Power10 built on this foundation by introducing enhanced processor acceleration and architectural advancements. Now, Power11 pushes the boundaries further, offering greater performance, expanded AI precision capabilities, and a more mature ecosystem—designed to meet the escalating demands of modern AI applications. Table 1-1 shows a comparison of AI-ready characteristics between the Power11 platform and previous generations.

Table 1-1 Technology comparison of the Power11 platform to previous generations

Characteristics	Power11	Power 10	POWER9
Processor Technology	7 nm	7 nm	14 nm
Maximum cores	16	15	12
Maximum SMT threads per core	SMT8	SMT8	SMT8
Maximum frequency	3.8 – 4.4 GHz	3.75 – 4.15 GHz	3.9 – 4.0 GHz
AI Precision Support	FP64/32/16, Bf16, INT8 (FP16/8 with Spyre)	FP64/32/16, Bf16, INT8	FP64/32 (with GPU), FP16
Memory Support	DDR5 and DDR4	DDR5 and DDR4	DDR4 and DDR3
Peak Memory Bandwidth	Up to 512 GBps	Up to 409 GBps	170 GBps
L2 Cache	2048 KB per core	2048 KB per core	512 KB per core
L3 Cache	8 MB of L3 cache per core with each core having access to the full 128 MB of L3 cache, on-chip high-efficiency SRAM	8 MB of L3 cache per core with each core having access to the full 120 MB of L3 cache, on-chip high-efficiency SRAM	10 MB of L3 cache per core with each core having access to the full 120 MB of L3 cache, on-chip high-efficiency eDRAM
I/O Interconnect	PCIe Gen 5	PCIe Gen 5	PCIe Gen 4, NVLink 2.0
On-Chip Acceleration	Improved MMA engine and SIMD vector instructions	Matrix Math Accelerator (MMA) engine	SIMD, Compression, Encryption (AES, SHA), GZIP engine
Off-Chip Acceleration	Up to 8 Spyre accelerator cards in an expansion drawer	N/A	Up to 6 Nvidia GPUs in a server (NVLink 2.0)
Python/ML Ecosystem	Improved Python performance, AI/ML stack fully optimized (IBM Open-Source AI Foundation for Power)	Optimized Python, PyTorch/TensorFlow for Power10 MMA (rocketce)	Standard Python, scientific libraries and opensource frameworks (opence)
Target AI workloads	Generative AI, real-time inference, Retrieval-Augmented Generation (RAG)	Hybrid cloud AI, mission-critical AI/ML inference.	GPU-assisted deep learning (DL), high-performance computing (HPC).

## 1.5.1 POWER9

POWER9 served as a foundational platform for enterprise AI, offering strong support for high-performance computing and data analytics. Its integration with NVIDIA GPUs via NVLink 2.0 enabled fast data exchange, which was essential for deep learning workloads. With high memory bandwidth and multi-threading capabilities, POWER9 could efficiently handle parallel AI tasks, although it relied heavily on external accelerators for peak performance.

In contrast, POWER11 represents a more advanced and AI-centric architecture. It expands native support for a wider range of AI data types, such as FP64, BF16, and INT8, allowing for more efficient training and inference across diverse models. POWER11 also incorporates on-chip AI acceleration, reducing dependence on external GPUs and improving overall performance and energy efficiency. Additionally, it benefits from a more mature software ecosystem, including optimized AI frameworks and enhanced security features, making it better suited for modern enterprise AI deployments.

### GPUs

The primary accelerator for the POWER9 platform is GPU support. The IBM POWER AC922 was the first server to leverage NVLink 2.0 technology to advance deep learning. Deep learning (DL) models are trained using GPUs such as the NVIDIA Tesla V100, available with either 16 GB or 32 GB of memory. Each GPU delivers up to 7.8 TFLOPs of double-precision performance, 15.7 TFLOPs for single precision, and up to 125 TFLOPs for deep learning workloads when the additional Tensor Cores are used. A water-cooled AC922 system (GTX model) can accommodate up to six GPUs.

### NVLink 2.0

Developed jointly by IBM, NVIDIA, and the OpenPOWER Foundation, NVLink 2.0 offers up to 5x the communication bandwidth between the CPUs and GPUs compared to traditional PCIe Gen 3 connections. This higher bandwidth significantly accelerates data transfer between system memory and GPUs, as well as between the GPUs themselves.

Each POWER9 processor connected via NVLink 2.0 provides up to 300 GB/s of bidirectional bandwidth with attached GPUs, compared to 32 GB/s with conventional PCIe Gen3.

Figure 1-1 on page 13 shows the comparison of POWER9 NVLink 2.0 with traditional processor interconnects that use PCIe and first-generation NVLink. For more information about GPUs and NVLink in POWER9 see *IBM Power System AC922 Technical Overview and Introduction*, REDP-5494.

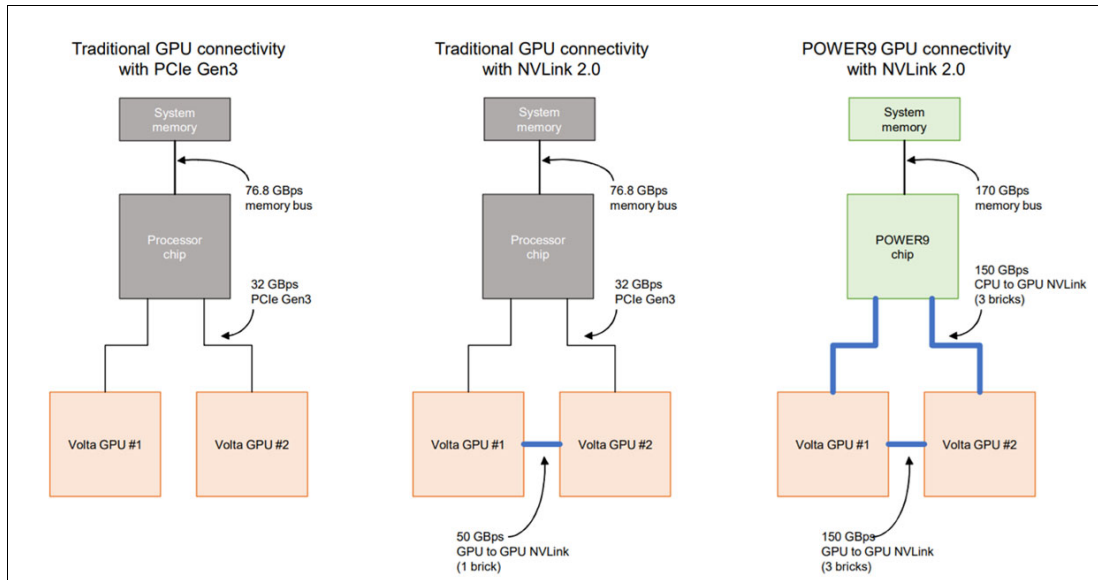


Figure 1-1 Comparison between traditional CPU-GPU interconnects and POWER9 using NVLink 2.0

## Target workloads

The POWER9 platform, when paired with NVIDIA GPUs and NVLink 2.0 interfaces, significantly boosts processing capabilities to meet the demands of high-performance computing (HPC), high-performance data analytics (HPDA), and artificial intelligence.

## 1.5.2 Power10

Power10 introduced significant advancements over POWER9, particularly in AI acceleration and architectural efficiency. It featured on-chip matrix math engines designed to speed up AI inference and training without relying solely on external GPUs. Power10 also supported a wider range of AI data types, including BF16 and INT8, which improved performance and reduced power consumption for deep learning workloads. Its memory bandwidth and security enhancements made it well-suited for enterprise AI deployments, especially in regulated industries.

Power11 builds on these capabilities with further refinements in AI performance and ecosystem maturity. It offers improved support for mixed-precision AI operations and delivers higher throughput for both training and inference tasks. Power11 also enhances the in-chip acceleration introduced in Power10, making it more efficient and scalable. Additionally, the platform benefits from a more optimized software stack and broader framework support, allowing developers to more easily deploy and manage AI workloads. Compared to Power10, Power11 provides better performance per watt, deeper integration with AI tools, and a more robust foundation for next-generation enterprise AI applications.

## Matrix Math Accelerator

The Matrix Math Accelerator (MMA) in Power10 is designed to accelerate computation-intensive operations such as matrix multiplication and convolutions. It uses a dense math engine (DME) microarchitecture to provide efficient acceleration for cognitive computing, machine learning, and AI inferencing workloads. For detailed coverage of MMA design and performance, refer to *Matrix-Multiply Assist Best Practices Guide*, REDP-5612.

## SIMD

Power10 delivers a substantial advancement in SIMD capabilities through a redesigned vector subsystem that doubles general-purpose SIMD throughput compared to POWER9. Each Power10 core includes eight independent fixed- and floating-point SIMD engines, providing higher parallelism for vectorized operations and improved efficiency in workloads involving dense numerical computation. In addition, Power10 introduces “2× general, 4× matrix SIMD relative to POWER9,” reflecting IBM’s expansion of SIMD bandwidth and execution width to better support AI, HPC, and analytics tasks. These enhancements operate alongside the Vector-Scalar Extension (VSX) instruction set of the Power ISA, enabling robust compiler auto-vectorization and optimized execution of scalar, vector, and mixed-precision operations. Combined, these features establish SIMD as a core component of Power10’s acceleration strategy, complementing the in-core MMA engine to deliver end-to-end performance gains for modern enterprise workloads.

## 1.6 Summary

Artificial Intelligence (AI) is transforming enterprise computing by enabling automation, advanced analytics, and personalized experiences. IBM Power11 emerges as a strategic platform for AI, offering enhanced performance, security, and scalability compared to previous generations. Key innovations include on-chip Matrix Multiply Assist (MMA), expanded SIMD (Single Instruction, Multiple Data) vector acceleration, the IBM Spyre accelerator, and support for mixed-precision operations, which optimize AI workloads for both training and inference.

The chapter highlighted AI fundamentals such as machine learning, deep learning, and natural language processing, along with common misconceptions and current enterprise applications. It also addressed industry trends, noting the initial excitement around large language models (LLMs) and the subsequent focus on overcoming their limitations, particularly in scalability and performance.

Power11 delivers enterprise-grade reliability, quantum-safe security, and hybrid cloud flexibility, integrating seamlessly with Red Hat OpenShift AI and IBM watsonx® for lifecycle management and governance. Compared to POWER9 and Power10, Power11 offers superior throughput, energy efficiency, and a mature ecosystem, making it an ideal platform for mission-critical AI deployments.

# Use Cases and Industry Applications

This chapter focuses on how AI can be integrated into enterprise environments using IBM Power Systems. It introduces deployment patterns that combine hardware acceleration, middleware, and application layers to embed AI into mission-critical workloads while maintaining security and compliance. These patterns support both traditional AI/ML and generative AI, enabling organizations to modernize operations and improve productivity.

The chapter also highlights practical applications and cross-industry use cases such as predictive analytics, natural language processing, computer vision, process automation, and generative AI. It emphasizes emerging trends like AIOps for automating IT operations and AI-driven DevOps for accelerating software delivery. Finally, it discusses real-time inference and edge computing, showing how Power11 systems deliver low-latency AI processing for scenarios like fraud detection, IoT, and autonomous systems.

The following topics are covered:

- ▶ AI Deployment Patterns and Use Cases
- ▶ AI for industries
- ▶ Cross industry AI

## 2.1 AI Deployment Patterns and Use Cases

This section showcases enterprise AI-ready deployment patterns and use cases built on IBM Power Systems. The patterns define the infrastructure and integration strategy — how AI is deployed across Power Systems and cloud environments.

They seamlessly integrate purpose-built accelerated hardware with traditional AI/ML and generative AI capabilities, delivered as an optimized full-stack solution. These patterns drive business growth and enhance productivity by embedding AI into mission-critical enterprise processes.

By keeping data where it resides, these patterns ensure data privacy and regulatory compliance.

The framework supports off-the-shelf GenAI capabilities such as entity extraction, RAG-based digital assistants. These are hardened and optimized for enterprise use. The framework also enables pre-built, use-case-driven AI services from the IBM-supported catalog, focusing on turnkey AI for enterprise workloads.

### 2.1.1 Enterprise AI-ready deployment pattern on Power systems

This is a generalized blueprint for integrating AI into mission-critical workloads using IBM Power infrastructure. It emphasizes hardware acceleration, AI readiness, and enterprise integration, which are core elements of a deployment pattern.

The deployment pattern is ideal for organizations running mission-critical workloads on IBM Power Systems that seek to modernize with AI integration — driving productivity improvements and ROI while maintaining on-premises control and ensuring data security.

Figure 2-1 presents a structured, layered view of the constituents grouping in the deployment pattern.

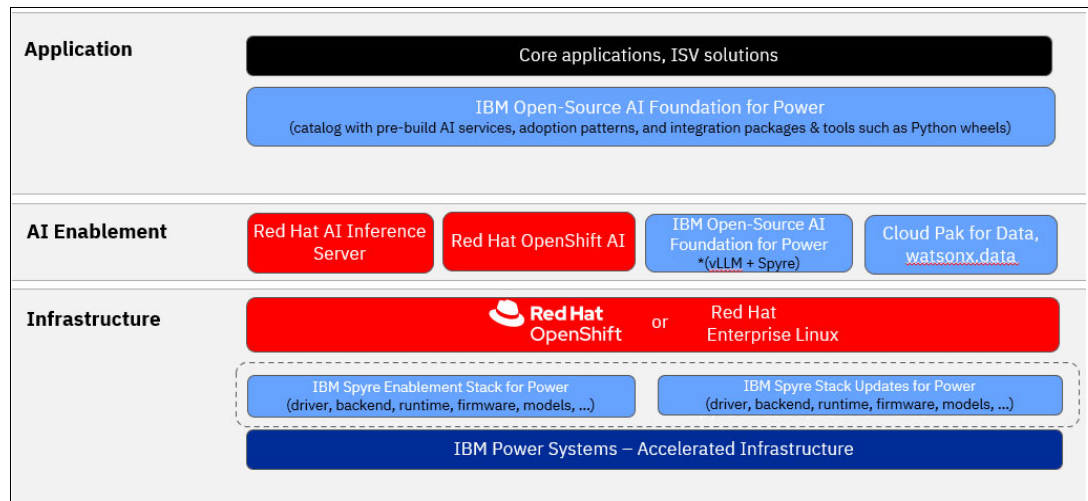


Figure 2-1 Enterprise AI-ready deployment pattern

The AI stack is organized into three logical layers, infrastructure, AI enablement, and application.

## Infrastructure Layer

This layer provides the foundational computing power and operating system services required to support AI workloads.

The key components of this layer are:

► IBM Power based servers – Accelerated AI Infrastructure

IBM Power offers a robust foundation for AI acceleration through specialized hardware:

- Matrix Math Accelerator (MMA): Power11’s integrated on-chip matrix engines designed to accelerate high-density tensor and matrix operations, delivering significantly higher throughput for deep learning inference and generative AI workloads.
- SIMD Acceleration: Expanded, high-throughput vector processing integrated into each Power11 core, accelerating parallel numeric operations essential for AI preprocessing, feature extraction, and real-time analytics.
- Off-chip Spyre: A high-performance architecture that enhances generative AI inferencing and large model execution beyond the chip.
- Middleware

The Enterprise AI stack includes middleware components that enable higher-level software to leverage the underlying infrastructure.

- IBM Spyre Enablement Stack for Power

Required for making use of Spyre from higher level software stacks. Enables integration of Spyre capabilities into AI frameworks and applications.

Includes:

- PyTorch backend
- Runtimes
- Device driver
- Card firmware
- Pre-compiled models with Spyre support

- IBM Spyre Stack Updates for Power

Provides regular updates for the IBM Spyre Enablement Stack for Power, including drivers, firmware, AI models, and other runtime components as needed.

► Operating Systems

- Red Hat Enterprise Linux (RHEL) on LPAR

Enables execution of open-source AI foundations, including Python-based inference frameworks like **vLLM**, delivering high-performance inferencing on Power Systems.

- Red Hat OpenShift (with Red Hat CoreOS nodes)

Hosts containerized enterprise AI platforms such as Cloud Pak for Data (CP4D), watsonx.data, and Red Hat OpenShift AI (RHOAI), providing orchestration and lifecycle management across Power infrastructure.

## AI Enablement Layer: Inference Server Components

This layer empowers the enterprise stack with AI capabilities, enabling model deployment, inference, and data-driven decision-making. It integrates both IBM-native and open-source components for flexibility and performance.

### The key components of this layer are:

- ▶ RHAIS (Red Hat AI Inference Server)  
Purpose-built for efficient generative AI inferencing, particularly optimized for IBM's Spyre architecture. RHAIS offers a lightweight and easy-to-deploy starting point, with minimal overhead—ideal for initial deployments on a single LPAR (Logical Partition).
- ▶ RHOAI (Red Hat OpenShift AI)  
A core platform for building, deploying, and managing AI applications using Kubernetes-native tools. RHOAI accelerates time-to-value through fit-for-purpose hardware acceleration and supports enterprise-grade features such as high availability (HA), user management, multi-tenancy, and scalable infrastructure, making it a strong choice for production AI workloads.
- ▶ IBM Open-Source AI Foundation for Power – Native integration  
This constituent represents native integration of vLLM with Spyre within the IBM Open-Source AI Foundation for Power. This is an open-source enablement layer with pre-built AI services and packages; supports native vLLM + Spyre integration for pure open-source inferencing.
- ▶ Cloud Pak for Data  
IBM Cloud Pak for Data (CP4D) runs on Red Hat OpenShift and serves as a unified platform that integrates multiple AI and data services to streamline the entire data-to-AI lifecycle.
- ▶ watsonx.data  
A scalable data lakehouse designed for analytics and AI workloads. It is deployed alongside CP4D to support data-intensive inferencing use cases.

## Application Layer

This layer enables seamless integration of AI into real-world enterprise applications, delivering business and industry-specific solutions powered by the underlying AI and infrastructure layers.

This layer consists of:

- ▶ IBM Open-Source AI Foundation for Power (Application Integration)  
Provides a catalog of pre-built AI services, adoption patterns, and integration packages to accelerate AI adoption.  
Includes tools such as:
  - Python wheels for object detection and image classification
  - PDF2Text and OCR (Optical Character Recognition) for document processing
- ▶ ISV Solutions (Industry-specific or AI use-case specific)  
Offers prepackaged AI solutions tailored to enhance core enterprise applications.  
Targeted domains include:
  - Core Banking Systems
  - Enterprise Resource Planning (ERP)
  - Enterprise Content Management (ECM)

These solutions are AI-ready and designed to address specific business use cases, enabling faster deployment

## Data Sources and AI Readiness Flow:

Enterprise data originates from a variety of transactional systems and databases, including:

- IBM Db2® (on AIX/IBM i)
- Oracle
- PostgreSQL
- SAP HANA

Depending on the use case the data can exist in several formats.

- For structured analytics, the data flows from source databases to data warehouses for reporting and BI.
- When using cold or archival data the data can be offloaded directly to watsonx.data to optimize storage and access.
- For unstructured data, the data is loaded into watsonx.data to support Retrieval-Augmented Generation (RAG) solutions.

Data transformation engines in watsonx.data provides multiple fit-for-purpose engines to prepare data for AI:

- Spark & Presto: Modify and transform structured data for predictive AI use cases.
- OpenSearch: Vectorize unstructured data to enable RAG workflows.

The workflow supports both traditional and generative AI models, enabling diverse use cases across the enterprise.

## 2.1.2 Traditional AI/ML on Power Systems – on-premises AI/ML deployment pattern

This deployment pattern is ideal for enterprises running mission-critical workloads on IBM Power Systems that seek to integrate traditional **AI/ML** capabilities while maintaining on-premises control, ensuring data security, and enabling real-time decision-making.

Figure 2-2 presents a structured, layered view of the deployment pattern where the top layer represents the AI/ML pipeline and inference flow, while the bottom layer shows infrastructure constituents that host the implemented functions.

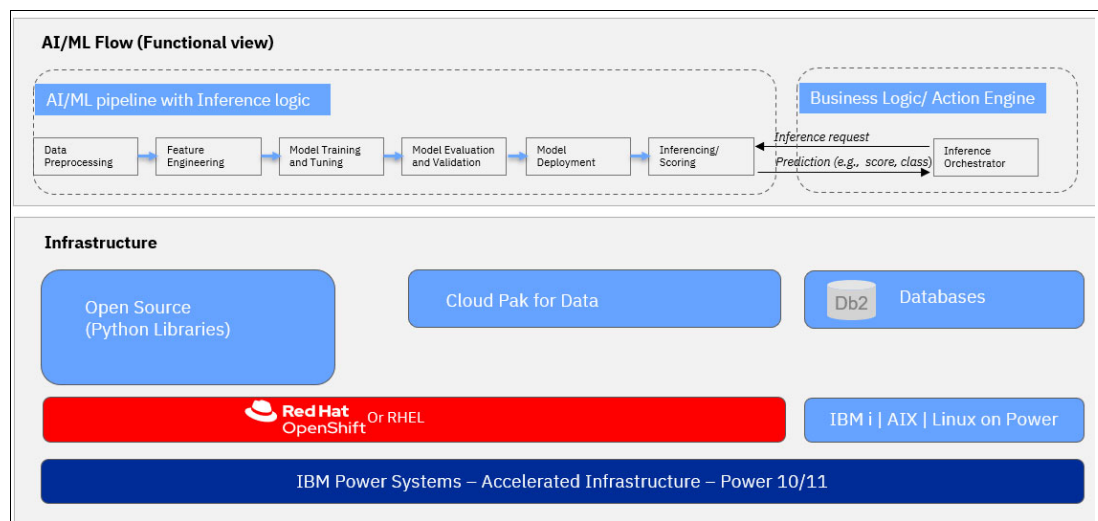


Figure 2-2 On-premises AI/ML deployment pattern

In this section, we focus on real-time AI/ML applications that tightly integrate with source data. While the emphasis is on real-time use cases, the underlying logic also applies to non-real-time scenarios such as batch inference.

The transaction control logic typically resides on the AIX or IBM i LPAR. When a new transaction record is inserted (or updated) in the database, a trigger or application logic initiates a synchronous API call to a classification inference endpoint — exposed by the AI/ML running pipeline on Linux LPAR(s). The model evaluates the transaction in real time and returns a risk score or classification, which is then used to determine downstream actions (e.g., approve, flag, or hold the transaction). The response guides action—approve, flag, or hold the transaction

AI/ML model training and inference are performed locally on IBM Power Systems. Enterprise data remains on-premises, ensuring robust security for sensitive information and compliance with data residency and regulatory requirements.

Real-time AI/ML applications, such as fraud detection, operate directly on source data to meet low-latency requirements. Other applications rely on curated data from warehouses or lakehouses (e.g., watsonx.data) for model training and batch inference

## Infrastructure Layer

The infrastructure layer comprises the following constituents:

- ▶ IBM Power Systems

IBM Power10 and Power11 systems offer a robust infrastructure for AI acceleration, leveraging math libraries optimized for on-chip Matrix Math Accelerators (MMAs).

- ▶ AIX, IBM i LPARs

These operating system platforms host mission-critical workloads such as:

- Core banking applications
- ERP systems (e.g., SAP, JD Edwards)
- Logistics and supply chain platforms

- ▶ Databases (e.g., Db2) on AIX/IBM i

Databases to store structured transactional data such as:

- Customer activity logs
- Payment records
- ERP and core banking transactions

- ▶ RHEL LPAR or Red Hat OpenShift

Provide environments to build/ run AI/ML pipelines and expose inference endpoints

RHEL LPARs offer lightweight Linux environments on IBM Power Systems. These partitions are ideal for running Python-based machine learning models and hosting popular frameworks such as TensorFlow and Scikit-learn.

Red Hat OpenShift offers a Kubernetes-based platform for container orchestration, supporting AI/ML workflows through containerized deployments. It integrates with IBM Cloud Pak® for Data to enable model development, deployment, and management.

- ▶ Open-Source Python libraries on RHEL

The Python open-source ecosystem offers a comprehensive suite of libraries—such as Scikit-learn for machine learning algorithms, Pandas for data manipulation and analysis, and TensorFlow for deep learning—that collectively support every stage of the AI and machine learning pipeline. These tools enable developers to efficiently handle tasks

ranging from data preprocessing and feature engineering to model training, evaluation, and deployment.

► **Cloud Pak for Data on Red Hat OpenShift**

IBM Cloud Pak for Data (CP4D) offers a suite of integrated modules that streamline the AI/ML lifecycle. The **AutoAI** module automates key stages of model development, including data preparation, algorithm selection, and pipeline generation.

- The Watson Studio Learning (WSL) module provides a collaborative environment for data scientists to develop notebooks, visualize data, and build models efficiently.
- The Watson Machine Learning (WML) module serves as the runtime engine for training, deploying, and managing machine learning models.
- Additionally, the IBM Knowledge Catalog (IKC) can be optionally integrated during the data preparation phase to support data governance, cataloging, and metadata management, ensuring trusted and well-documented data assets throughout the pipeline.

### **AI/ML flow (functional view)**

This layer represents the **logical and functional architecture** of how tasks are executed across systems. A typical Machine Learning pipeline includes the following steps:

1. **Data Preprocessing** - Clean raw data to make it usable
  - Handle missing values
  - Convert data types (strings to numerics)
  - Encode categorical variables (oneshot, label encoding)
2. **Feature Engineering** - Create or select the most informative features
  - Combine or split columns
  - Extract time/date features
  - Apply domain logic (flag anomalies)
  - Drop irrelevant or highly correlated features
3. **Model Training + Tuning** - Train model(s) on preprocessed data
  - Choose algorithm: [RandomForest](#), [XGBoost](#), [LightGBM](#) as examples
  - Tune hyperparameters (e.g., depth, learning rate)
  - Optionally do cross-validation
  - Model Evaluation & validation
  - Use metrics like [accuracy](#), [precision](#), [recall](#), [F1-score](#), [ROC AUC](#)
  - Check for overfitting or underfitting
4. **Model Deployment** - Make the model accessible for inference
  - Export as .pkl, .pmml, etc.
  - Deploy to RESTful endpoint (e.g., Flask, FastAPI, or Watson ML)
5. **Inference**: Generate predictions from new data
  - Accepts raw input, preprocesses (if pipeline is included), and returns output
  - Real-time (RESTful API) or batch inference (CSV, database, etc.)

### **Business Logic/ Action Trigger**

This component represents enterprise business applications running on AIX and IBM i, integrated with AI/ML inference orchestration for traditional workloads. Enterprise systems such as ERP, core banking, and logistics on AIX/IBM i platforms connect with AI capabilities using APIs and event bridges (e.g., RESTful APIs, message queues, Kafka).

An Inference Orchestrator serves as the bridge between AI models and business logic. It transforms legacy data formats into model-ready inputs and routes AI predictions back to enterprise systems for actionable outcomes.

- ▶ Key use cases include:
- ▶ Fraud detection in banking
- ▶ Demand forecasting in logistics
- ▶ Customer churn prediction
- ▶ Credit scoring
- ▶ Inventory optimization

### **Traditional AI/ML pipeline build approach on Power**

A traditional AI/ML pipeline is implemented on IBM Power infrastructure using the following commonly used approaches:

#### ▶ **Open-Source**

Open-source Python library-based solution offers full control over pipeline design and development, making it ideal for early-stage prototyping and experimentation by data science teams. However, this approach lacks built-in governance and automation and requires strong in-house expertise in ML and DevOps to scale and operationalize.

#### ▶ **IBM Cloud Pak for Data**

IBM Cloud Pak for Data (CP4D) based solutions are designed for enterprise-grade ML pipelines with robust governance, compliance, and production-readiness. They integrate seamlessly with enterprise data systems and core business processes, supporting auditability, explainability, and scalability. CP4D is well-suited for organizations prioritizing security, collaboration, and lifecycle management.

#### ▶ **Hybrid Approach**

A commonly adopted strategy combines the agility of open-source tools (e.g., Python, Scikit-learn, Jupyter) with the enterprise capabilities of CP4D. Data scientists often develop custom scripts within CP4D's notebook environment, leveraging both flexibility and governance in a unified platform.

**Note:** To facilitate experimentation, data science teams often use synthetic, sample, or public datasets external to the enterprise databases for initial model training and validation. The model is later fine tuned and validated with real production data from the database (e.g. Db2)

## **2.1.3 Hybrid Agentic AI with IBM watsonx Orchestrate and On-Premises Power**

This scenario-driven pattern introduces a hybrid architecture combining cloud-native orchestration (watsonx) with on-premises compute. It combines structured ML pipelines with GenAI inference for richer insights and automation.

This deployment pattern is ideal for enterprises running mission-critical workloads on IBM Power Systems seeking to integrate Generative Agentic AI capabilities while maintaining on-premises control, ensuring data security, and enabling real-time decision-making.

### **Architecture Overview**

watsonx Orchestrate® on IBM Cloud brings agentic AI capabilities to legacy backend systems running on IBM Power (IBM i, AIX). Hybrid Infrastructure connectivity between cloud and on-premises environments is achieved via secure, managed connectivity solutions such as IBM Cloud Satellite®.

Generative AI-Driven Modernization enhances traditional enterprise workloads by infusing generative AI capabilities focused on workflow automation and intelligent decision support.

For customer-initiated flows, an AI agent can present itself as a chat bot interface embedded in a web page for customers to interact with. Orchestrate® agentic workflow is triggered to retrieve data from on-premises Power system backend for contextual enrichment and for next action.

For system-initiated flow, an on-premises event (e.g., a transaction) can push the event to Orchestrate via API. Orchestrate initiates workflows, performs GenAI-driven tasks, sends messages as well as other functions.

The pattern securely exposes subsets of on-premises enterprise data to AI agents running in IBM Cloud. Industry-specific examples include Fraud detection and prevention, predictive maintenance, intelligent customer support with context aware communication in banking, finance, healthcare. Figure 2-3 presents a structured, layered view of the deployment that reflects how watsonx Orchestrate operates in hybrid enterprise environments.

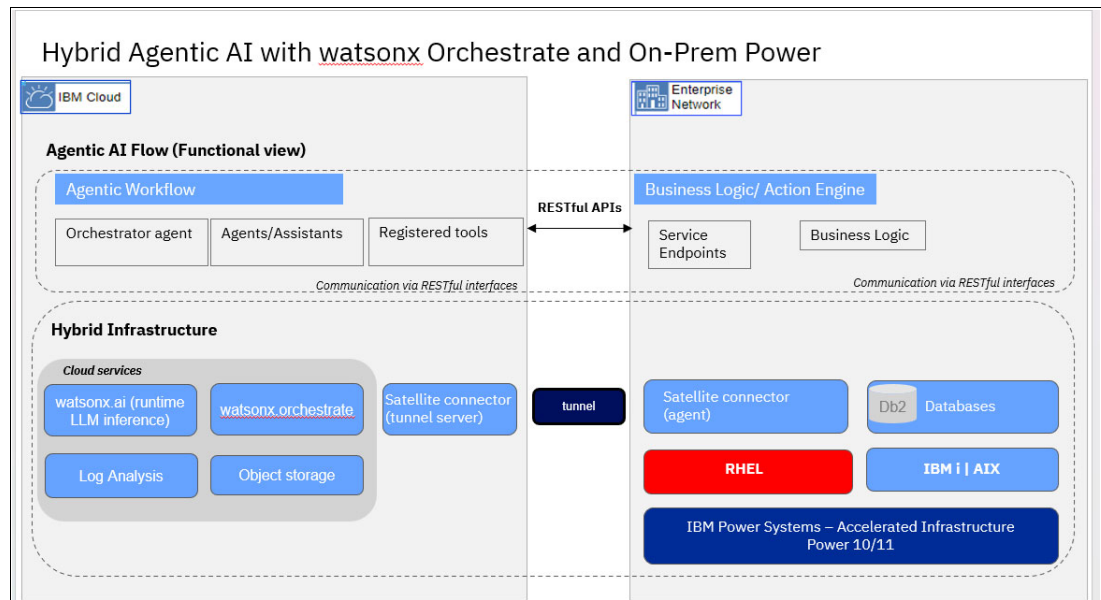


Figure 2-3 Hybrid Agentic AI with watsonx Orchestrate and On-premises Power

### Hybrid Infrastructure layer

This layer defines the **physical and network topology** that enables secure communication across Cloud and on-premises environments

- ▶ The cloud side consists of:
  - IBM watsonx Orchestrate: The orchestration engine that coordinates agentic workflows.
  - IBM watsonx.ai®: Hosts the LLM runtime for generative AI tasks (e.g., summarization, classification, decision support).
  - Satellite Connector (Tunnel Service): Secure bridge that enables cloud agents to interact with on-premises services without exposing internal networks.
- ▶ The on-premises enterprise side consists of:
  - IBM Power10 and Power11 systems offer a robust infrastructure for AI acceleration, leveraging math libraries optimized for on-chip Matrix Math Accelerators (MMAs) and off-chip Spyre cards.

- RHEL: Hosts the **Satellite Connector Agent**, which establishes outbound tunnels to the cloud.
- AIX / IBM i with DB2: Core enterprise systems running mission-critical workloads and exposing APIs or services for orchestration.

### **Agentic AI Flow Layer (Functional view)**

This layer represents the **logical and functional architecture** of how tasks are executed across systems.

- ▶ Cloud Side:
  - Orchestrator Agent: Central agent that manages workflow execution and coordination.
  - Agents / Assistants: Specialized AI agents that perform tasks like data extraction, decision-making, or user interaction.
  - Registered Tools: Catalog of callable service endpoints (APIs, connectors) that agents use to perform actions.
- ▶ On-premises Enterprise Side:
  - ▶ Business Logic: Core application logic (e.g., RPG programs, COBOL routines, Java services) that drives enterprise processes.
  - ▶ Service Endpoints (Tool Implementations): RESTful APIs or service wrappers that expose business logic to the orchestration layer.

## **2.1.4 Pre-built AI use cases with IBM Spyre for Power**

IBM Spyre for Power is an integrated full stack solution that unlocks turnkey AI to enterprises, including capabilities like modern digital assistants.

The offering provides out-of-the-box AI solutions tailored to specific enterprise use cases enabling faster adoption.

At the foundation is the AI services layer, which includes modular, reusable capabilities such as document digitization, entity extraction, and vector search. These services act as technology enablers, powering higher-level adoption patterns.

The adoption pattern layer provides composable solutions that combine AI services into enterprise ready workflow.

At the top of the stack are real-world applications that deliver measurable business value. These use cases are powered by adoption patterns and tailored to specific enterprise functions such as IT service management, HR automation, and supply chain optimization.

Spyre accelerates AI adoption with pre-built, one-click-install AI services. These services can be integrated into enterprise processes following proven adoption patterns. Figure 2-4 on page 25 shows some example use cases.

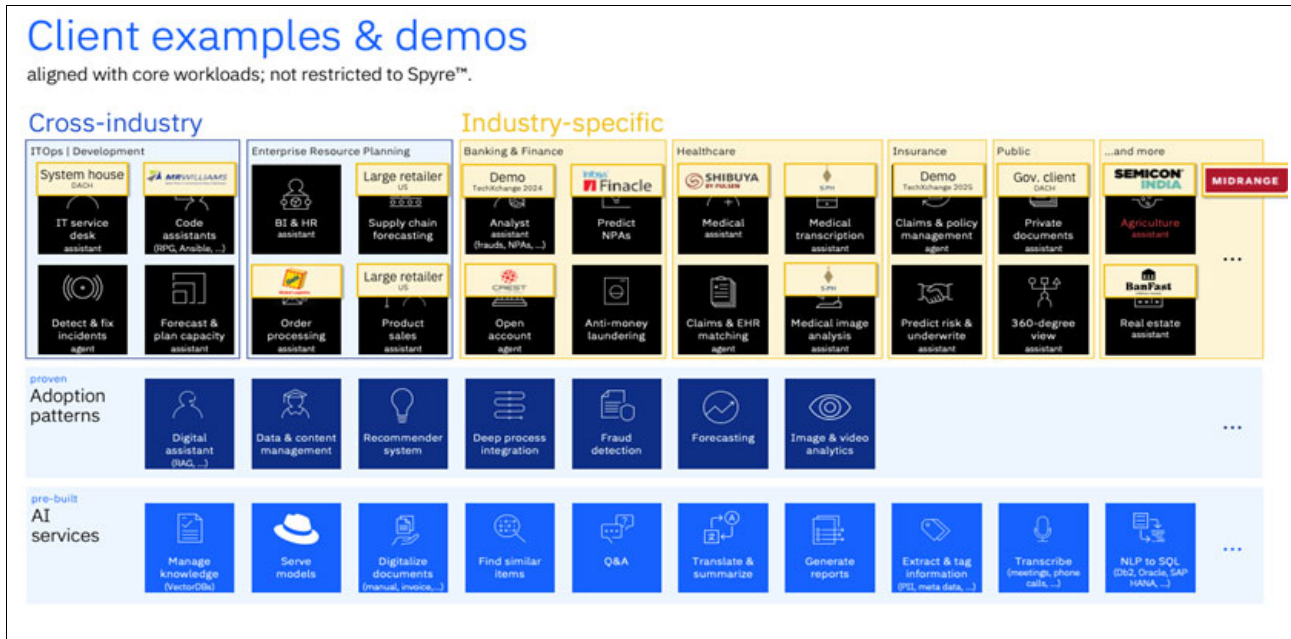


Figure 2-4 Example use cases

As an example, a traditionally manual ERP order process can be automated. AI services read incoming order emails, extract details, and fill out ERP fields many times faster than experienced human operators.

For GA 1, the focus is on Digital Assistants (“RAG Pipeline”) which has different AI services to enable the service (like Digitalize documents from PDF to markdown via Python libraries, or build up a VectorDB). This is shown in Figure 2-5.



Figure 2-5 Digital assistant use cases

Another AI service is Entity Extraction with a simple integration into existing workflows. These use cases are shown in Figure 2-6.

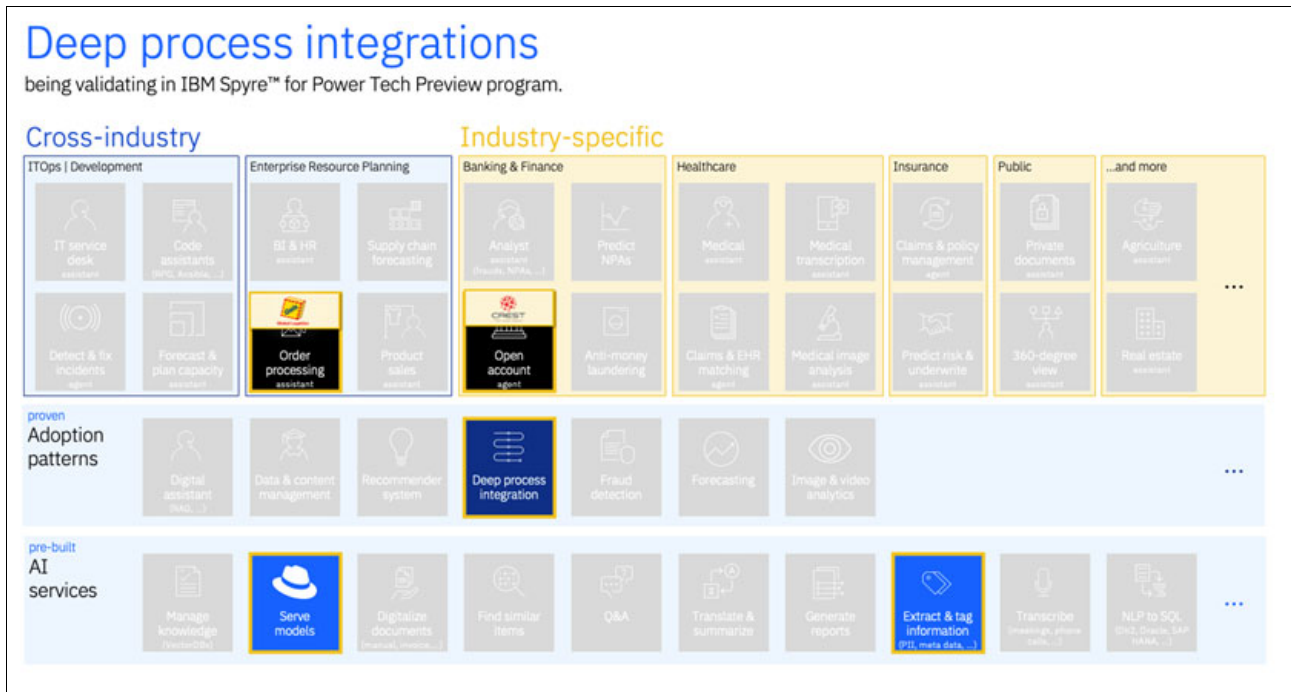


Figure 2-6 Process integration use cases.

## 2.2 AI for industries

Artificial Intelligence (AI) is reshaping industries by enabling smarter, faster, and more efficient processes tailored to sector-specific needs. From banking and finance to healthcare, manufacturing, and government, AI delivers transformative capabilities that go beyond generic applications. In each domain, AI addresses unique challenges—such as fraud detection in financial services, predictive maintenance in manufacturing, personalized treatment in healthcare, and citizen service automation in government. By leveraging advanced analytics, machine learning, and generative AI, organizations can unlock new levels of innovation, improve decision-making, and deliver enhanced experiences for customers and stakeholders. This industry-focused approach ensures that AI solutions are not only powerful but also aligned with regulatory, operational, and business requirements.

### 2.2.1 Banking and Finance

Artificial Intelligence (AI) and Machine Learning (ML) have fundamentally transformed the banking and finance industry, driving efficiency, powering new customer experiences, enhancing security, and transforming risk management. Banks have moved from early experiments to deploying AI at scale across almost every core business function.

AI adoption began in the 1980s with expert systems for fraud detection and credit scoring. Early ML algorithms supported risk assessment, trading, and compliance monitoring as data sources expanded. The surge in digital banking and cloud computing after 2010 unlocked vast, accessible datasets, accelerating AI innovation.

Traditional ML has evolved into deep learning and generative AI, powering conversational chatbots, predictive analytics, and real-time fraud detection. The emergence of regulatory technology (RegTech), robo-advisors, and algorithmic trading has moved AI from back-office tasks to customer-facing and mission-critical roles.

### **Customer Service**

Chatbots and virtual assistants are widely used throughout the industry to service requests for account queries and offer personalized financial advice. They improve operational efficiencies and customer satisfaction.

### **Fraud Detection**

Advanced AI systems monitor transactions in real time to detect unusual behaviors that may signal fraud or money laundering. Trained on historical data, these models establish normal user patterns and combine them with identifiers such as email, phone number, and IP address to spot anomalies quickly. This approach reduces both fraud losses and false positives.

### **Credit Scoring**

Major banks have adopted AI that analyze hundreds of data points, including alternative and real-time data to deliver more accurate and inclusive credit decisions. This deeper insight enables lenders to tailor loan terms, interest rates, and repayment plans to a borrower's actual financial situation.

### **Compliance & Regulatory Reporting (RegTech)**

Financial institutions leverage AI to automate suspicious activity reports, verify KYC documents, and perform compliance checks. Natural language processing (NLP) systems interpret new regulations and proactively adjust workflows, reducing risk and minimizing manual effort during audits.

### **Investment Management**

AI systems analyze market trends, execute trades, and update portfolio strategies in real time, often outperforming traditional quantitative models. Generative AI simulates thousands of market scenarios, enabling advisors and clients to make data-driven decisions for portfolios, hedging, and regulatory stress tests. Robo-advisors deliver personalized wealth management by tailoring strategies, automating rebalancing, and synthesizing research instantly, boosting both client returns and advisor productivity.

### **Operations**

AI also reconciles transactions, monitors operational risk, and reduces errors in reconciliations and settlements across large institutions.

### **Summary**

AI has become a cornerstone of modern banking and finance, enabling institutions to deliver faster, smarter, and more personalized services while strengthening security and compliance. From fraud detection and credit scoring to investment management and operational efficiency, AI-driven solutions are redefining how financial organizations operate.

Looking ahead, generative AI will play a growing role in financial advisory, enabling hyper-personalized investment strategies and real-time scenario modeling for clients. Quantum computing promises breakthroughs in risk modeling and portfolio optimization, allowing institutions to process complex simulations at unprecedented speed. Meanwhile,

AI-driven RegTech will continue to evolve, automating compliance in response to increasingly dynamic global regulations.

However, these advancements bring new challenges. Financial institutions must address algorithmic transparency, bias mitigation, and data privacy to maintain trust and meet regulatory expectations. Collaboration between banks, technology providers, and regulators will be critical to establish ethical frameworks and governance standards. Ultimately, the future of AI in banking and finance lies in balancing innovation with accountability, ensuring that technology drives sustainable growth while safeguarding customer confidence and systemic stability.

## **2.2.2 Government**

AI is now embedded across many government functions, from citizen services and tax compliance to policing and infrastructure management. Local, regional, and national agencies leverage AI across a wide range of applications.

### **Citizen Services**

Similar to corporate call centers, government agencies handle large volumes of routine inquiries. AI-powered chatbots now respond to questions about taxes, permits, policies, and procedures, reducing staff workload and improving response times.

### **Transportation and Infrastructure**

Governments use AI to monitor traffic conditions, analyze flow patterns, and adjust signals in real time, reducing congestion and improving safety. The same data supports predictive maintenance for infrastructure, helping prevent costly failures. AI also analyzes sensor data from public vehicles such as buses, garbage trucks, and mail carriers to schedule maintenance proactively, minimizing downtime. Local governments apply AI to optimize energy use in facilities, adjust street lighting, and manage water supply driving sustainability and cost savings. Planning agencies leverage AI to evaluate demand, demographics, and environmental data, enabling smarter project prioritization and funding allocation for roads, utilities, and public buildings.

### **Law Enforcement & Public Safety**

Metropolitan police departments employ predictive analytics to forecast crime hotspots and allocate patrols more effectively, enhancing public safety and resource efficiency. Machine learning models analyze historical incident data to identify trends and guide deployment decisions, complementing traditional crime analysis. Public agencies also use computer vision and pattern recognition to monitor sensitive infrastructure and public spaces, improving threat detection, though these applications raise important governance and privacy concerns.

### **Border Security & Immigration**

The European Union's iBorderCtrl system combines automated questioning, facial analysis, and risk scoring to prescreen travelers, accelerating low-risk crossings and focusing human attention on higher-risk cases. U.S. Customs and Border Protection applies AI to screen cargo at ports of entry, validate identities via the CBP One app, and enhance threat awareness at the border. AI-powered baggage screening uses object detection and image classification to identify prohibited items in carry-on luggage. Immigration agencies also leverage AI to streamline services, eliminating redundant paperwork by consolidating customer information from multiple systems.

## **Tax Administration & Fraud Detection**

AI models assist tax authorities in selecting representative samples for audits and identifying returns likely to contain errors or unpaid taxes. Canada's tax agency uses machine learning to cross-check filings against financial data, uncovering patterns of evasion and recovering significant sums while focusing audits on high-risk cases. Similarly, the Australian Taxation Office applies AI to analyze returns at scale, flag anomalies, and guide compliance strategies, improving accuracy and efficiency. Governments also use AI to detect suspicious benefit claims by spotting patterns such as repeated contact details or inconsistent information, reducing welfare fraud. Real-time monitoring of transactions in taxation, procurement, and social programs helps prevent and investigate fraud before funds are lost.

## **Environmental Regulation**

The U.S. Environmental Protection Agency applies AI to analyze air quality data in real time, informing pollution control policies and enabling cities to respond more effectively to environmental risks. Municipalities use AI to integrate environmental and demographic data into urban planning, green space allocation, and resilience projects, aligning infrastructure with climate and population trends. Regulators also leverage AI to process large volumes of filings, comments, and technical documents, helping staff identify key issues and patterns when developing or enforcing regulations.

## **Patent Examination & Intellectual Property**

DesignVision, an AI-powered tool, searches U.S. and international industrial design collections using images as input queries. It provides centralized access and federated searching across design patents, registrations, trademarks, and industrial designs from more than 80 global registers. The U.S. Patent and Trademark Office also uses AI to assist examiners in locating relevant documents, streamlining the search and adjudication of new patent applications.

## **Summary**

AI is transforming government operations by improving efficiency, reducing costs, and enhancing service delivery across diverse domains, from citizen engagement and infrastructure management to law enforcement and environmental regulation. These technologies enable data-driven decision-making and proactive risk management, but they also introduce challenges around privacy, transparency, and ethical governance. As adoption accelerates, balancing innovation with accountability will be critical to ensuring that AI serves the public interest responsibly.

## **2.2.3 Manufacturing**

Artificial Intelligence (AI) and Machine Learning (ML) have revolutionized manufacturing by improving efficiency, enabling predictive maintenance, optimizing supply chains, and driving innovation in product design. Manufacturers have progressed from early automation to deploying AI at scale across production lines, quality control, and logistics.

AI adoption in manufacturing began with rule-based systems for process automation and quality checks in the 1980s. As sensor technology and industrial IoT matured, ML algorithms enabled predictive maintenance and real-time monitoring of equipment health. The rise of smart factories and Industry 4.0 initiatives after 2010 accelerated AI integration, leveraging vast datasets from connected machines and enterprise systems.

Traditional ML has evolved into deep learning and generative AI, powering advanced applications such as defect detection through computer vision, dynamic scheduling, and

autonomous robotics. These technologies have moved AI from isolated tasks to mission-critical roles in production and operations.

### **Predictive Maintenance**

AI models analyze sensor data from machines to predict failures before they occur, reducing downtime and maintenance costs. By identifying patterns in vibration, temperature, and pressure readings, these systems enable proactive interventions and extend equipment life.

### **Quality Control**

Computer vision systems powered by deep learning inspect products in real time, detecting defects with higher accuracy than manual checks. This approach improves consistency, reduces waste, and ensures compliance with stringent quality standards.

### **Supply Chain Optimization**

AI-driven analytics forecast demand, optimize inventory, and streamline logistics. By integrating data from suppliers, production schedules, and market trends, these systems minimize delays and reduce operational costs.

### **Robotics and Automation**

Collaborative robots (cobots) equipped with AI adapt to changing tasks on the factory floor, improving flexibility and productivity. Generative AI enhances robotic programming by simulating thousands of scenarios for optimal performance.

### **Product Design and Innovation**

AI accelerates design cycles by analyzing customer feedback, material properties, and performance data. Generative design algorithms propose innovative solutions that balance cost, durability, and sustainability.

### **Summary**

AI has become a cornerstone of modern manufacturing, enabling smarter production, predictive insights, and agile operations. From predictive maintenance and quality control to supply chain optimization and robotics, AI-driven solutions are reshaping how factories operate. Looking ahead, generative AI will play a growing role in design and simulation, while edge AI will enable real-time decision-making on the shop floor. Manufacturers must address challenges such as data security, workforce reskilling, and ethical AI deployment to ensure sustainable growth and competitiveness in an increasingly digital industrial landscape.

## **2.2.4 Healthcare**

The use of AI in healthcare dates back several decades to the early expert systems of the 1960s, computer-aided diagnoses systems of the 1970s, image recognition systems in the 1980s, and natural language processing systems of the 1990s that could “understand” medical texts. Of course, progress did not stop there as the use of AI in healthcare has continued to increase, improving patient care, discovering new treatments, and increasing efficiency.

### **Drug development**

Drug discovery is a complex, costly, and time-consuming process that on average takes 10-15 years to bring a single drug to market. And success rates are notoriously low with only about 10% of candidates every making it to market.

However, machine learning and generative AI are helping researchers to identify and optimize drug candidates to accelerate the process and improve efficiency. AI can analyze massive datasets of scientific literature, genetic information, and chemical compounds to identify potential new drug candidates. Large language models are used to engineer proteins, predicting how changes in an amino acid sequence affect its function.

AI is also used to make clinical trials more efficient. By analyzing electronic health records, AI can identify trial participants based on factors such as age, medical history, and genetic makeup. Further, AI can optimize patient stratification and identify medical “twins” for inclusion in control groups.

Another interesting use case is the repurposing of existing medicines. AI is used to review health records and scientific journals to discover potential new applications for existing drugs.

### **Cancer drug research**

Artificial Intelligence (AI) is accelerating cancer drug discovery by analyzing vast datasets of genomic information, clinical trial results, and chemical compound libraries. Machine learning models identify promising drug candidates by predicting molecular interactions and assessing toxicity profiles, significantly reducing the time and cost of traditional research. Deep learning techniques enable virtual screening of millions of compounds, while generative AI designs novel molecules optimized for efficacy and safety. AI also supports personalized oncology by matching therapies to individual genetic profiles, improving treatment outcomes. By integrating real-world evidence and simulation models, AI-driven platforms are transforming cancer research from a slow, trial-and-error process into a data-driven, precision-focused approach.

### **Patient Care**

AI excels at analyzing medical images, such as X-rays and CT scans, to detect abnormalities, and can perform these tasks quicker and with fewer errors than humans. In emergency situations the reduced time to a diagnosis can be the difference between survival and death for a patient. In more routine scans, AI can look for early signs of medical conditions unrelated to the current situation and thereby catch issues that otherwise would have gone unnoticed.

Patients are frequently monitored, either in a facility or with remote devices, to collect data such as heart rate, oxygen levels, glucose levels. AI can continuously analyze that data to catch early signs of a problem.

AI-enhanced robotic systems enable surgeons to perform complex surgeries with greater precision and control. As a result, procedures are less invasive with fewer complications and shorter patient recovery times.

AI can analyze a patient’s medical history, lifestyle, and genetic information to create tailored treatment plans. This can include matching medications to a patient that are more likely to be effective and/or cause fewer side effects.

### **Administration**

Natural language processors have been used for a long time to transcribe spoken words into notes. Over the years, this technology has improved and is now capable of analyzing patient-provider conversations in real time, extracting the relevant information for clinical notes.

Tedious administrative tasks such as scheduling, coding, room assignments, and billing can be largely automated. This reduces overhead, enhances efficiency, and lowers costs.

AI can analyze historical data, news reports, and social media to predict demand. This can allow facility administrators to more accurately forecast admission rates and staffing requirements.

### **Key Challenges**

The adoption of AI in healthcare raises ethical concerns given the sensitivity of much of the data and the potential high stakes of decisions.

AI models may show biases due to being trained on data that is not representative of the entire population. This has historically been an issue in medicine with women and minorities frequently being underrepresented in medical research and studies. Therefore, care needs to be taken in choosing the data used for AI training.

Questions of security and ownership of medical data also need to be considered. Patient information can often be sensitive and protected by laws like HIPPA and GDPR. It is imperative then that the data be properly protected during both training and use with AI.

Over reliance of AI in diagnosis in patient care can potentially erode human-to-human empathy and trust between the medical provider and the patient. Over the longer term, some are concerned that it could lead to a decline in healthcare professionals' clinical skills and reasoning.

## **2.2.5 Airlines**

Artificial Intelligence (AI) is transforming the airline industry by improving operational efficiency, enhancing passenger experience, and optimizing safety and maintenance. Airlines have evolved from using basic automation for scheduling to deploying AI-driven solutions across nearly every aspect of their operations.

### **Evolution of AI in Airlines**

Early AI adoption focused on revenue management and flight scheduling using rule-based systems. With the rise of big data and cloud computing, machine learning enabled predictive analytics for demand forecasting and dynamic pricing. Today, deep learning and generative AI power advanced applications such as real-time flight optimization, personalized travel experiences, and predictive maintenance.

Key Applications of AI in Airlines

#### **Predictive Maintenance**

AI models analyze sensor data from aircraft engines and systems to predict component failures before they occur. This reduces unscheduled downtime, improves safety, and lowers maintenance costs.

#### **Flight Operations Optimization**

AI systems optimize flight paths, fuel consumption, and crew scheduling by analyzing weather patterns, air traffic, and operational constraints. These solutions help airlines reduce delays and improve on-time performance.

#### **Customer Experience**

Chatbots and virtual assistants provide real-time support for booking, check-in, and flight status inquiries. Generative AI enables hyper-personalized offers, loyalty program recommendations, and tailored travel itineraries.

### **Revenue Management and Dynamic Pricing**

AI algorithms forecast demand and adjust ticket prices dynamically based on factors such as seasonality, competitor pricing, and booking trends. This maximizes revenue while maintaining competitive pricing.

### **Safety and Compliance**

AI-driven systems monitor flight data and operational logs to detect anomalies and ensure compliance with aviation regulations. Natural language processing (NLP) assists in analyzing incident reports and regulatory updates.

### **Crew and Resource Scheduling**

AI optimizes crew assignments and resource allocation, reducing operational bottlenecks and improving workforce efficiency.

### **Summary**

AI has become a critical enabler for modern airlines, driving improvements in safety, efficiency, and customer satisfaction. From predictive maintenance and flight optimization to personalized passenger services and dynamic pricing, AI solutions are reshaping the industry. Looking ahead, generative AI will enhance operational planning and customer engagement, while edge AI will enable real-time decision-making during flights. Airlines must address challenges such as data privacy, algorithmic transparency, and cybersecurity to ensure responsible and sustainable AI adoption.

## **2.2.6 Retail**

Artificial Intelligence (AI) is reshaping the retail industry by enabling personalized shopping experiences, optimizing supply chains, and improving operational efficiency. Retailers have moved beyond basic recommendation engines to deploying AI-driven solutions across merchandising, customer engagement, and inventory management.

### **Evolution of AI in Retail**

Early AI adoption focused on product recommendations and demand forecasting using rule-based systems. With the rise of e-commerce and omnichannel retail, machine learning enabled dynamic pricing, personalized marketing, and real-time inventory optimization. Today, deep learning and generative AI power advanced applications such as virtual shopping assistants, automated store operations, and immersive customer experiences.

### **Personalized Recommendations**

AI analyzes customer behavior, purchase history, and preferences to deliver tailored product suggestions across online and in-store channels, increasing conversion rates and customer loyalty.

### **Dynamic Pricing and Promotions**

AI algorithms adjust prices and promotions in real time based on demand, competitor pricing, and inventory levels, helping retailers maximize revenue and maintain competitiveness.

### **Inventory and Supply Chain Optimization**

AI-driven forecasting predicts demand accurately, reducing stockouts and overstock situations. Intelligent systems optimize logistics and warehouse operations for faster delivery and lower costs.

## **Customer Service and Engagement**

Chatbots and virtual assistants provide instant support for inquiries, returns, and product searches. Generative AI enhances engagement by creating personalized marketing content and interactive shopping experiences.

## **Fraud Detection and Security**

AI monitors transactions and user behavior to detect anomalies, preventing payment fraud and safeguarding customer data.

## **Visual Search and Augmented Reality**

Computer vision enables customers to search for products using images, while AR applications allow virtual try-ons, improving the shopping experience.

## **Summary**

AI has become a cornerstone of modern retail, driving personalization, efficiency, and innovation. From dynamic pricing and inventory optimization to immersive shopping experiences, AI solutions are redefining how retailers operate and engage with customers. Looking ahead, generative AI will enable hyper-personalized marketing and virtual store environments, while edge AI will support real-time analytics in physical stores. Retailers must address challenges such as data privacy, ethical AI use, and integration with legacy systems to ensure sustainable and responsible adoption.

## **2.2.7 Agriculture**

Artificial Intelligence (AI) is transforming agriculture by enabling precision farming, improving crop yields, and optimizing resource usage. Farmers and agribusinesses have moved from traditional practices to deploying AI-driven solutions for soil analysis, irrigation management, and supply chain efficiency.

### **Evolution of AI in Agriculture**

Early AI adoption focused on basic automation for irrigation and pest control. With the rise of IoT sensors and satellite imagery, machine learning enabled predictive analytics for crop health and weather forecasting. Today, deep learning and computer vision power advanced applications such as automated harvesting, disease detection, and yield prediction.

### **Precision Farming**

AI systems analyze soil conditions, moisture levels, and nutrient requirements to optimize planting and fertilization schedules. This reduces waste and improves crop productivity.

### **Crop Monitoring and Disease Detection**

Computer vision and drone-based imaging detect early signs of disease, pests, and nutrient deficiencies, enabling timely interventions and minimizing crop loss.

### **Weather Forecasting and Risk Management**

AI models predict weather patterns and climate risks, helping farmers plan planting and harvesting schedules more effectively.

### **Automated Irrigation and Resource Management**

AI-driven irrigation systems adjust water distribution based on real-time soil and weather data, conserving water and reducing costs.

## Supply Chain Optimization

AI forecasts demand and optimizes logistics for transporting produce, reducing spoilage and improving market efficiency.

## Robotics and Autonomous Equipment

AI-powered robots perform tasks such as planting, weeding, and harvesting, reducing labor costs and increasing operational efficiency.

## Summary

AI is revolutionizing agriculture by enabling data-driven decisions, improving sustainability, and increasing profitability. From precision farming and crop monitoring to automated irrigation and supply chain optimization, AI solutions are redefining how food is grown and delivered. Looking ahead, generative AI will enhance crop modeling and breeding strategies, while edge AI will enable real-time decision-making in the field. Farmers must address challenges such as data privacy, infrastructure limitations, and workforce training to ensure responsible and effective AI adoption.

## 2.3 Cross industry AI

Artificial Intelligence (AI) is transforming business operations across all sectors by addressing universal challenges through adaptable, horizontal solutions. Key cross-industry applications include:

- ▶ Predictive Analytics for trend forecasting and risk management.
- ▶ Natural Language Processing (NLP) powering chatbots, virtual assistants, and automated document handling.
- ▶ Computer Vision for image/video analysis in security, quality control, and content moderation.
- ▶ Recommendation Systems delivering personalized experiences for products, services, and content.
- ▶ Process Automation (RPA) streamlining repetitive tasks like data entry and reporting.
- ▶ Anomaly Detection for fraud prevention, cybersecurity, and operational monitoring.
- ▶ Generative AI creating text, images, and code for marketing, design, and development.
- ▶ Optimization Algorithms improving supply chains, resource allocation, and scheduling.
- ▶ AIOps (AI for IT Operations) automating incident detection, root cause analysis, and system performance optimization.
- ▶ AI-driven DevOps accelerating software delivery through intelligent testing, deployment automation, and predictive failure analysis.

These horizontal use cases enable organizations to enhance efficiency, reduce costs, and unlock new opportunities regardless of industry, making AI a strategic imperative for competitive advantage.

### 2.3.1 AI in IT Operations

Enterprise IT environments have grown increasingly complex, making traditional operations difficult to manage. The adoption of hybrid cloud architectures, containerized applications, and microservices has dramatically increased the volume and velocity of operational data. Conventional monitoring tools often struggle to keep pace with these demands.

To address these challenges, organizations are turning to AIOps (Artificial Intelligence for IT Operations), AI-driven solutions designed to improve efficiency and responsiveness in IT operations.

### **What is AIOps?**

AIOps platforms analyze massive volumes of telemetry data, including logs, metrics, events, traces, and configuration details. Using machine learning and statistical techniques, they:

- ▶ Detect anomalies in real time
- ▶ Correlate related events
- ▶ Predict potential service disruptions

This proactive approach enables IT teams to reduce downtime, accelerate issue resolution, and manage systems more effectively.

### **Key Benefits**

AIOps delivers a range of capabilities that transform IT operations from reactive to proactive, enabling greater efficiency, resilience, and automation across complex environments

- ▶ **Alert Reduction:** AIOps groups similar notifications and filters out non-actionable alerts, allowing engineers to focus on critical incidents.
- ▶ **Root Cause Analysis:** AI maps system dependencies and identifies underlying issues, which is essential in dynamic, distributed environments.
- ▶ **Automation:** Common tasks—such as restarting services, scaling resources, or applying updates—can be executed automatically, reducing manual effort and minimizing human error.
- ▶ **Capacity Planning & Optimization:** By analyzing historical data and usage trends, AIOps helps teams allocate resources efficiently and control costs.
- ▶ **Security & Compliance:** Continuous monitoring and automated responses strengthen security posture and ensure policy adherence.
- ▶ **Enhanced IT Service Management:** AI enables automated ticket handling, sentiment analysis, and chatbot interfaces, improving user experience and accelerating resolution.

### **Why AIOps Now?**

Modern IT ecosystems—spanning hybrid clouds, microservices, and edge computing—generate continuous streams of logs, metrics, and events across multiple layers (network, compute, storage, and application). Traditional monitoring systems, built for static infrastructures, cannot keep up. They produce excessive alerts, lack contextual correlation, and require significant manual intervention.

AIOps overcomes these limitations by applying advanced analytics, machine learning, and automation to IT operations data. Instead of relying on static thresholds or manual triage, AIOps platforms ingest diverse data sources, normalize them, and use AI models to detect anomalies, correlate events, and predict failures before they impact business services

This shift from reactive firefighting to intelligent, automated operations reduces Mean Time to Repair (MTTR), minimizes service disruptions, and delivers predictive insights for capacity planning and performance optimization.

## **Core Capabilities of AIOps**

AIOps delivers a comprehensive set of capabilities that enable IT organizations to move from reactive troubleshooting to proactive, predictive, and automated operations, ensuring resilience and efficiency in today's complex digital environments

### ***Anomaly Detection***

Anomaly detection is one of the foundational capabilities of AIOps. Instead of relying on static thresholds, which often fail in dynamic environments, AIOps uses machine learning models to learn normal system behavior and identify deviations in real time. These models analyze historical and streaming data—such as logs, metrics, and traces—to establish baselines for expected performance. When patterns deviate, such as sudden CPU spikes, memory leaks, or unusual network latency, the system flags these anomalies immediately. This early detection prevents cascading failures and enables IT teams to intervene proactively before service degradation occurs, significantly reducing downtime and improving overall system reliability.

### ***Event Correlation***

Modern IT environments generate thousands of alerts daily, creating overwhelming noise and alert fatigue for operations teams. AIOps addresses this challenge through intelligent event correlation. By applying AI-driven algorithms, AIOps groups related alerts and identifies the underlying root cause, transforming what appears to be multiple unrelated issues into a single actionable incident. For example, instead of treating 50 alerts from different microservices as separate problems, AIOps can correlate them to a single failing database node. This capability reduces operational noise, accelerates troubleshooting, and improves Mean Time to Resolution (MTTR), allowing teams to focus on resolving critical issues rather than sifting through redundant alerts.

### ***Intelligent Alert Management***

Intelligent alert management builds on event correlation by further reducing noise and prioritizing alerts based on business impact. AIOps platforms automatically cluster similar alerts, suppress non-actionable notifications, and rank incidents according to severity and potential disruption. This ensures that engineers spend their time addressing the most critical issues rather than being overwhelmed by thousands of low-priority alerts. By streamlining alert workflows, AIOps improves operational efficiency, reduces response times, and helps organizations maintain service continuity even in highly complex environments.

### ***Root Cause Analysis***

Identifying the root cause of an issue in a dynamic, distributed infrastructure can be extremely challenging. AIOps simplifies this process by leveraging AI to map system dependencies and analyze event patterns across multiple layers of the IT stack. This capability eliminates guesswork by pinpointing the exact source of a problem—whether it's a misconfigured load balancer, a failing microservice, or a network bottleneck. By accelerating root cause identification, AIOps reduces downtime, minimizes the risk of recurrence, and enables teams to implement long-term fixes rather than temporary workarounds.

### ***Predictive Analytics***

AIOps goes beyond reactive monitoring by incorporating predictive analytics to forecast future states of the infrastructure. Using historical data and advanced machine learning models, AIOps can predict potential issues such as disk capacity exhaustion weeks in advance or anticipate network congestion during peak loads. These insights enable IT teams to plan capacity effectively, optimize resource allocation, and avoid unplanned outages. Predictive analytics also supports cost optimization by ensuring that resources are scaled appropriately based on anticipated demand, helping organizations maintain performance while controlling expenses.

### Automated Remediation

Automation is a key differentiator of AIOps, closing the loop between detection and resolution. When anomalies or failures are identified, AIOps can trigger predefined workflows or scripts to resolve issues automatically. Common examples include restarting failed services, scaling resources during traffic surges, or applying configuration updates without human intervention. Automated remediation minimizes manual effort, reduces downtime, and supports self-healing systems, significantly lowering the risk of human error. This capability allows IT teams to focus on strategic initiatives rather than repetitive operational tasks.

### Architecture Overview of AIOps

The architecture of AIOps is designed to handle massive volumes of heterogeneous IT operations data, apply AI-driven analytics, and automate remediation. It consists of three primary layers: Data Sources, the AI/ML Pipeline, and the Integration Layer, all working together to deliver intelligent, automated IT operations. Figure 2-7 provides an overview of the AIOps architecture.

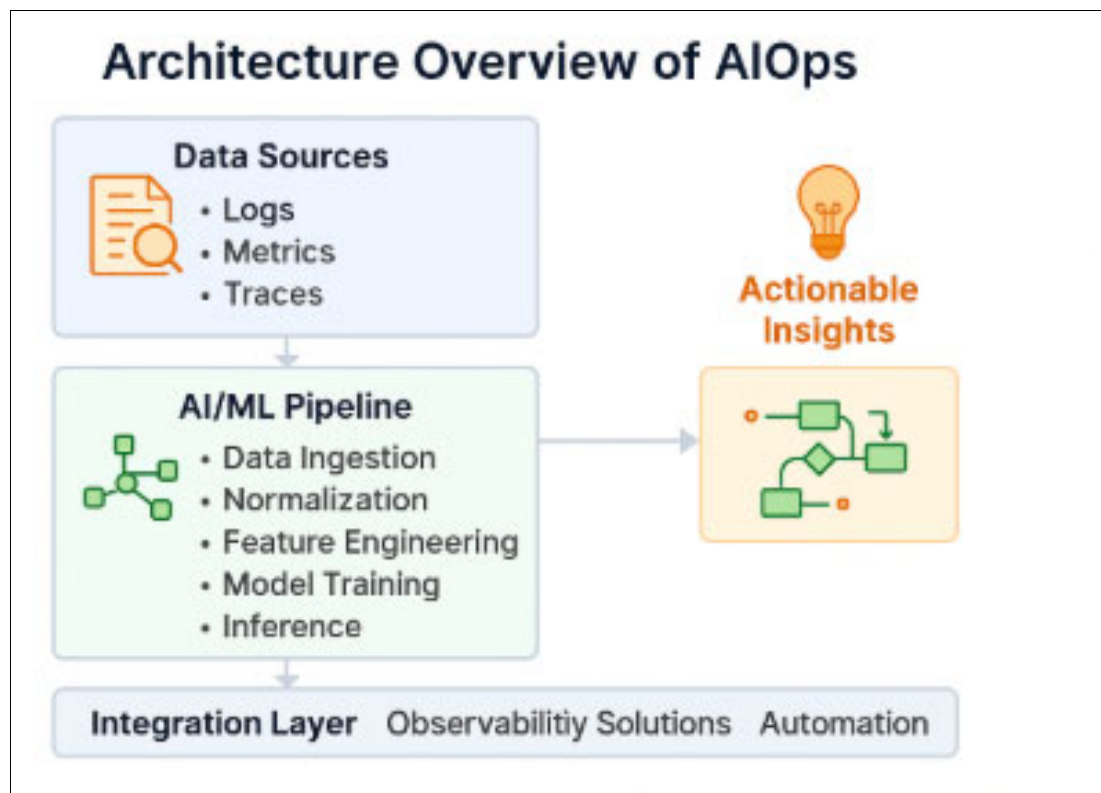


Figure 2-7 AIOps architecture

### Data Sources

Modern IT environments generate data from multiple layers and components, including logs, metrics, traces, and alerts. Logs—such as application, system, and security logs—provide granular event details, while metrics capture resource utilization like CPU, memory, network throughput, and storage performance. Distributed tracing data from microservices and containerized workloads adds visibility into complex transactions, and alerts from monitoring tools and observability platforms highlight potential issues. These data streams are often unstructured or semi-structured and originate from diverse sources such as cloud platforms, on-premises systems, and edge devices. AIOps platforms must ingest and normalize this data to ensure consistency and enable effective analysis.

### ***AI/ML Pipeline***

The intelligence of AIOps lies in its data processing and machine learning pipeline. This pipeline typically includes several stages:

- ▶ **Data Ingestion:** Collecting data from multiple sources using connectors and APIs.
- ▶ **Normalization:** Converting heterogeneous data into a unified format for consistency.
- ▶ **Feature Engineering:** Extracting relevant features for model training.
- ▶ **Model Training:** Building machine learning models for anomaly detection, event correlation, and predictive analytics.
- ▶ **Inference:** Applying trained models to real-time data streams to generate actionable insights.

To achieve this, AIOps platforms often leverage technologies such as Apache Kafka for streaming, Apache Spark for distributed processing, and ML frameworks like TensorFlow or PyTorch for model development and deployment.

### ***Integration Layer***

AIOps does not operate in isolation; it integrates seamlessly with existing IT management ecosystems to close the loop between detection and resolution. This includes IT Service Management (ITSM) tools like ServiceNow or Remedy for incident and change management, observability solutions such as Prometheus, Grafana, or OpenTelemetry for metrics and traces, and automation engines like Ansible or Chef to execute remediation workflows. These integrations ensure that insights generated by AIOps translate into automated actions, improving operational efficiency and reducing downtime.

### ***End-to-End Flow***

The AIOps workflow follows a structured process:

1. **Data Collection:** Logs, metrics, and traces flow into the ingestion layer.
2. **Processing & Analysis:** AI/ML models detect anomalies, correlate events, and predict failures.
3. **Actionable Insights:** Alerts are enriched with context and sent to ITSM tools for prioritization.
4. **Automation:** Remediation scripts or workflows are triggered automatically, enabling self-healing and minimizing manual intervention.

### **Best Practices**

Successfully adopting AIOps requires a strategic approach that balances technology, data, and business alignment. The following best practices help organizations maximize the value of AIOps initiatives:

- ▶ **Start with High-Impact Use Cases**  
Begin by focusing on use cases that deliver immediate and measurable benefits, such as reducing alert noise or accelerating incident resolution. These quick wins build confidence, demonstrate ROI, and create momentum for broader adoption.
- ▶ **Ensure Data Readiness and Governance**  
AIOps relies on high-quality, comprehensive data from diverse sources. Establish strong data governance practices to ensure accuracy, consistency, and security. This includes normalizing data formats, managing access controls, and maintaining compliance with regulatory requirements.
- ▶ **Continuously Retrain Models with Feedback Loops**

Machine learning models improve over time when they incorporate real-world feedback. Implement continuous learning processes that retrain models based on operational outcomes and evolving system behaviors. This ensures that anomaly detection, event correlation, and predictive analytics remain accurate and relevant.

- ▶ **Align AIOps Initiatives with Business Objectives**

AIOps should not be treated as a purely technical project. Align initiatives with broader business goals such as reducing downtime, improving customer experience, and optimizing costs. Define clear KPIs and success metrics to measure impact and communicate value to stakeholders.

## 2.3.2 AI DevOps

AI DevOps is the practice of applying DevOps principles to the development, deployment, and maintenance of Artificial Intelligence (AI) models and applications. It bridges the gap between data science and IT operations, ensuring that AI solutions are delivered reliably, securely, and at scale.

### **Evolution of AI DevOps**

Traditional DevOps focuses on automating software delivery pipelines, enabling continuous integration and continuous deployment (CI/CD). AI introduces new complexities such as model training, data versioning, and performance monitoring. AI DevOps extends these principles to include:

- ▶ **Continuous Training (CT):** Updating models as new data becomes available.
- ▶ **Model Versioning:** Tracking changes in model architecture and parameters.
- ▶ **Data Governance:** Managing datasets for compliance, quality, and reproducibility.

### **Key Components of AI DevOps**

Implementing AI DevOps requires more than simply adapting traditional DevOps practices. AI introduces unique challenges such as managing large and evolving datasets, monitoring model performance over time, and ensuring compliance with ethical and regulatory standards. To address these complexities, AI DevOps incorporates specialized components that streamline the entire lifecycle—from data preparation and model training to deployment and ongoing governance. These components enable organizations to automate workflows, maintain transparency, and scale AI solutions effectively across hybrid environments.

#### ***Automated Pipelines for AI***

AI DevOps pipelines integrate data ingestion, feature engineering, model training, validation, and deployment. Tools like Kubeflow, MLflow, and Red Hat OpenShift AI enable automation across these stages.

#### ***Monitoring and Observability***

AI systems require monitoring beyond uptime and latency. Metrics include model accuracy, drift detection, and fairness indicators. Continuous monitoring ensures models remain reliable in production.

#### ***Scalable Infrastructure***

AI workloads demand high-performance compute resources. Platforms like IBM Power11 provide integrated accelerators and hybrid cloud flexibility, enabling scalable training and inference.

### ***Security and Compliance***

AI DevOps incorporates security at every stage, from encrypted data pipelines to quantum-safe cryptography for model storage. Compliance frameworks ensure adherence to regulations like GDPR and industry-specific standards.

### ***Collaboration and Governance***

AI DevOps fosters collaboration between data scientists, developers, and operations teams. Governance tools manage model lifecycle, audit trails, and explainability to maintain trust and accountability.

### **Benefits of AI DevOps**

The major benefits of AI DevOps are:

- ▶ Faster deployment of AI models into production.
- ▶ Improved reliability and scalability for mission-critical applications.
- ▶ Enhanced security and compliance for sensitive data.
- ▶ Continuous improvement through automated retraining and monitoring.

### **Summary**

AI DevOps transforms AI from experimental projects into enterprise-grade solutions by embedding automation, governance, and scalability into the AI lifecycle. As organizations adopt AI at scale, AI DevOps becomes essential for delivering consistent, secure, and high-performing AI services across hybrid environments.

## **2.3.3 Real-time inference**

Real-time inference refers to the process of running an AI or machine learning model and generating predictions or insights instantly as new data is received. Instead of processing data in batches or offline, real-time inference delivers outputs with minimal latency, enabling immediate decision-making for applications like fraud detection, autonomous vehicles, and personalized recommendations.

Real-time inference occurs when you apply Artificial Intelligence (AI) models instantly, as data is generated. This contrasts with the training process, which requires significant resources to build models. With inference, organizations can deploy pre-trained models, such as LLMs and machine learning models, in distributed environments close to the data source.

Most enterprises face the challenge of processing massive data volumes in real time, and real-time inference addresses this need by enabling immediate insights and decisions.

In technical terms, real-time inference means that:

- ▶ AI models run with low latency.
- ▶ Systems can handle high volumes of concurrent requests.
- ▶ The infrastructure must be resilient, as it is not always possible to rely on cloud connectivity.

Here are some strong examples of real-time inference in action:

- ▶ **Fraud Detection in Financial Transactions**  
Banks and payment processors use real-time inference to analyze transactions as they occur, flagging suspicious activity instantly to prevent fraud.
- ▶ **Autonomous Vehicles**

Self-driving cars rely on real-time inference to process sensor data and make split-second decisions for navigation and safety.

- ▶ **Personalized Recommendations**

Streaming platforms and e-commerce sites deliver product or content recommendations in real time based on user behavior and context.

- ▶ **Predictive Maintenance in Manufacturing**

IoT sensors on equipment feed data to AI models that predict failures or maintenance needs immediately, reducing downtime.

- ▶ **Healthcare Monitoring**

Wearable devices and hospital systems use real-time inference to detect anomalies in vital signs and alert medical staff instantly.

- ▶ **Cybersecurity Threat Detection**

AI models monitor network traffic and identify potential breaches or malware in real time to protect enterprise systems.

## **Edge computing**

Edge computing is a distributed IT architecture that brings computation and data storage closer to where data is generated at the edge of the network. Instead of sending all data to centralized cloud or data centers, edge computing processes it locally. This reduces latency, improves real-time decision-making, and optimizes bandwidth. Edge computing is ideal for scenarios such as IoT, autonomous systems, and remote operations where speed, security, and reliability are critical.

Edge computing is used to:

- ▶ Reduce latency and avoiding the transfer of data to data centers or the cloud.
- ▶ Reduce bandwidth costs and storage capabilities.
- ▶ Provide compliance with data governance and privacy which require that data be processed locally.

In all these scenarios, Power11 provides the computational density, memory bandwidth, and MMA accelerator support required to sustain real-time inference workloads. Its architecture is designed to minimize latency under high concurrency, maintain throughput even in data-intensive edge environments, and ensure operational continuity where network connectivity to central data centers or cloud resources cannot be guaranteed.

### ***Power11 Capabilities for edge computing***

The Scale-Out Power11 models S1124, S1122, and S1112 are engineered for real-time inference at the edge, where critical workloads must operate with minimal resources. These systems deliver optimal performance for distributed environments through the following capabilities:

- ▶ **Low Latency and High Throughput**

Inference pipelines require extremely low latency to avoid bottlenecks and maintain performance. Power11 addresses this with a memory subsystem that includes DDR5 and OMI interfaces, increasing bandwidth by up to 50 percent compared to the previous generation and reducing data access latency. Each core is enhanced with higher IPC and SMT-8, enabling concurrent execution of multiple threads with stable response times. This combination of memory bandwidth and core efficiency allows inference models to run small batch sizes in real time.

- ▶ **Integrated AI Acceleration: MMA and Spyre**

Power11 integrates Matrix Math Assist (MMA) directly on the CPU, providing deterministic, cache-friendly acceleration for dense linear algebra operations such as matrix multiplication, convolution, and attention kernels. This reduces latency per token or frame without relying on external accelerators, which is critical for edge computing.

For workloads requiring higher throughput, such as multiple video or image streams, Power11 supports the Spyre adapter to extend compute capacity beyond a single socket. This approach maintains low latency for small batch inference using MMA while leveraging Spyre for parallel or large batch inference to maximize overall throughput.

- ▶ **Security, Resiliency, and Reliability**

Edge environments often operate without continuous connectivity to cloud or data centers, yet require the same level of security and availability. Power11 provides secure boot, memory encryption, and LPAR isolation, ensuring workload protection even if a partition fails. The platform also supports Zero Planned Downtime, allowing firmware and OS updates without interrupting workloads. This ensures edge nodes can process data, safeguard information, and run AI models independently of WAN connectivity.

**Examples of architecture edge computing**

The architecture of edge computing could be organized on three layer: far edge (end devices, micro-CPU, sensors closed to source data) where capture and filter data, near edge micro-data centers with computing and storage for inference in real-time and use models of ML/LLM with cache in data and resiliency with connection between core data centers and Data center/core (on-premises or cloud) is a central orchestration with high computing and storage, solution for DR and can recollect info for the end of cycle for the edge computing.

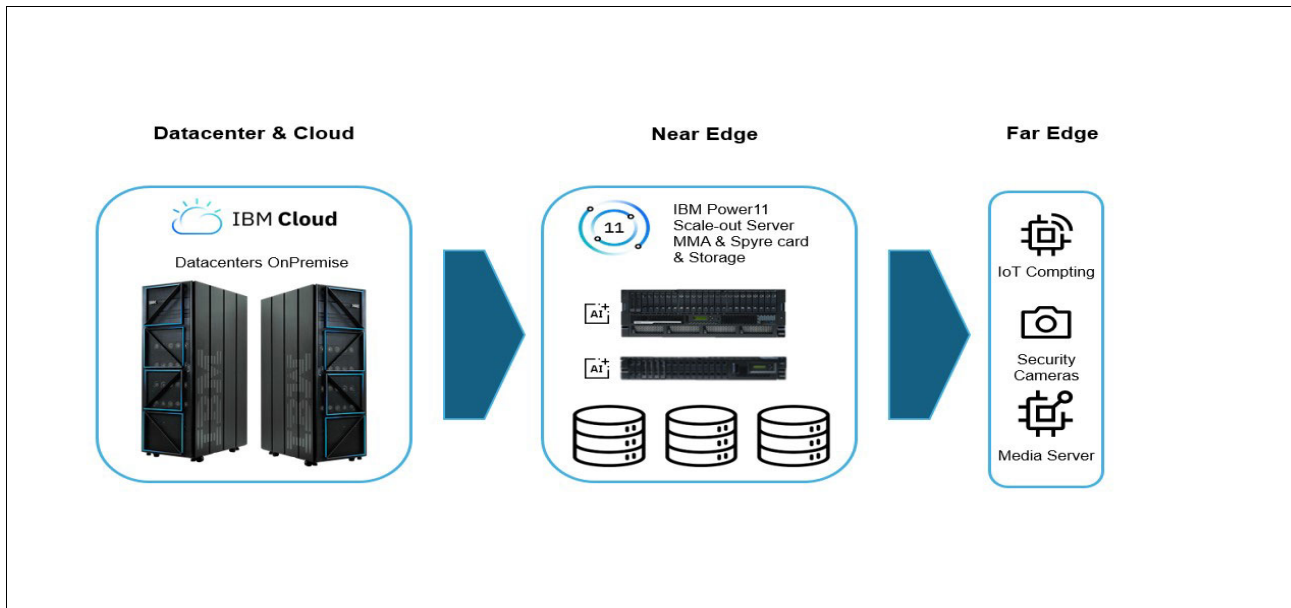


Figure 2-8 Example Architecture Edge Computing

In this architecture, the flow begins close to the physical source of the data: signals and events are captured at the edge by devices and micro-computers/controllers make the pre-filtering to contain volume without losing context (timestamps, asset identity, location). This first segment doesn't make the real high compute to favor determinism, low power, and continuous operation even under intermittent connectivity. Data is then securely routed to a nearby compute node with greater compute and I/O density, a site micro-datacenter, where real-time inference executes, and multiple streams are correlated. This tier balances proximity

with compute density, reduces WAN traffic, and preserves data privacy, while keeping operations during outages via persistent queues and observability.

Data gravity drives the data, the higher the volume and more compliance requirements, the more it makes sense to bring compute closer to the source and to transport only summaries, embeddings, and events that preserve value at lower cost and risk. In practice, the where to process decision becomes nearly deterministic once four variables are quantified (target latency, volume data, local retention, and compliance). The inference and correlation live in the nearby domain with local compute and storage, and training, historical analytics, and orchestration are reserved for the core. This decouples temporal criticality from analytical criticality, aligning network and compute costs with business SLOs and ensuring continuity in environments with non-deterministic connectivity.

### ***Use Cases or Examples***

This section discusses some example use cases for edge computing in different industries.

#### ► Banking

Some examples for the use of the edge computing in banking with the utilization of the technology IBM Power11, could be the next one:

In pre-authorization and anti-fraud at POS & ATM, the decision must be made as quickly as possible so as not to cause latency in authorizing the transaction. Terminals operate as far edge for capture and pre-filtering; inference lives at the near edge of the branch or in a regional hub to correlate transactions and risk signals, while the core consolidates history, recalibrates models, and manages global rules. Power11 provides low-latency inference on CPU with MMA for high concurrency, LPAR isolation for compliance, and non-disruptive updates to maintain continuous operation.

The near edge runs document classification pipelines, PII detection, field validation, risk rules, and eventually a compact LLM for intent/summary; the core consolidates metrics and retrains models with anonymized data. Power11 provides concurrency for multiple simultaneous streams with stable latency, enabling LLM inference and real-time validations; LPAR isolation protects sensitive data by branch, and zero-downtime resilience prevents workstation interruptions during maintenance.

#### ► Retail

In shelf availability, cameras continuously capture images and the store runs image analysis to check for stock shortages and planogram deviations, while the core adds multi-store KPIs and centrally retrains the models for inference in each store. Power11 supports deterministic inference with MMA to perform real-time analysis of camera footage in each store.

In stores where the self-checkout model is implemented, sensors and cameras in the store deliver signals that can be processed in the shortest possible time in Power11 with MMA to decide in real time if there is a risk (concealment, non-payment). The value of Power11 lies in its low latency, which does not slow down the payment experience, and in maintaining the store's maximum availability through LPAR to isolate failures and data.

The focus of real-time inference is to give the point where born data complies with the SLA, latency, reduce costs of the networking, and keep control of sensitive data. Edge computing is the way to make it viable; deploy computing close to the end point (sensors, cameras, IoT) for filtering and take the decision without losing time to send data to the data center or cloud.

The result is a deterministic and predictable one, where the network ceases to be a bottleneck and privacy is respected by processing what is appropriate locally.

Power11 fits naturally into these models for these reasons: low latency on the CPU with on-chip MMA for SML (Small Language Models) or some Machine Learning Models, and

enterprise-class resilience to operate remote sites without interrupting business. The Power11 scale-out configurations prioritize density computing per watt and high parallelism with SMT8 to maintain SLAs without oversizing.



## On-Chip Acceleration

This chapter explores the evolution and current state of on-chip acceleration technologies in IBM Power processors, culminating in the advanced capabilities introduced with IBM Power11. It provides a technical overview of how IBM has progressively integrated specialized hardware accelerators to optimize performance for AI, analytics, and high-throughput workloads.

The chapter also discusses how these capabilities are exposed through open software stacks, including ONNX, PyTorch, and TensorFlow, and how they integrate with IBM's broader AI ecosystem, including Watsonx and Red Hat OpenShift.

The following topics are covered in this chapter:

- ▶ Overview of AI acceleration features
- ▶ AI Acceleration on IBM Power11: MMA and SIMD Optimization
- ▶ Training of Machine Learning Models

### 3.1 Overview of AI acceleration features

IBM Power11 introduces a new level of AI acceleration designed to meet the demands of modern enterprise workloads. Traditional AI deployments have relied heavily on external GPUs for training and inference, which can introduce latency, increase costs, and complicate data governance. Power11 addresses these challenges by integrating advanced acceleration technologies directly into its architecture, enabling high-throughput AI processing with lower latency and improved energy efficiency.

At the core of this capability is the Matrix Multiply Assist (MMA) feature, which accelerates matrix operations—the foundation of machine learning and deep learning—directly in hardware. MMA reduces computational overhead for tasks such as neural network training and inference, delivering exceptional performance for complex AI models.

These innovations enable Power11 to run AI workloads closer to where data resides, reducing dependency on external GPU clusters and supporting hybrid deployments across edge, on-premises, and cloud environments. With support for mixed-precision operations such as INT8 and FP16, Power11 delivers optimized performance for both training and inference, making it a trusted platform for mission-critical AI applications.

### 3.2 AI Acceleration on IBM Power11: MMA and SIMD Optimization

IBM Power11 builds on a decade of architectural innovation to deliver advanced AI acceleration capabilities. At the heart of this design is the Matrix Multiply Assist (MMA) feature, which accelerates matrix operations—the foundation of machine learning and deep learning—directly in hardware. MMA enables high-performance execution of matrix multiplication and related operations, reducing computational overhead for neural network training and inference.

Starting with IBM POWER7, the Vector Scalar Extension (VSX) capabilities were introduced to accelerate computation through vectorization. VSX provides facilities for vector and scalar binary floating-point operations, extending and unifying the 32 floating-point registers (FPRs) of the Power ISA to 128 bits. These are combined with the existing 32 128-bit Vector Multimedia Extension (VMX) registers to create a single register file known as the Vector-Scalar Register (VSR). Figure 3-1 shows the construction of the VSR to enable SIMD instructions.

.qword															
.dword[0]								.dword[1]							
.word[0]				.word[1]				.word[2]				.word[3]			
.hword[0]		.hword[1]		.hword[2]		.hword[3]		.hword[4]		.hword[5]		.hword[6]		.hword[7]	
.byte[0]	.byte[1]	.byte[2]	.byte[3]	.byte[4]	.byte[5]	.byte[6]	.byte[7]	.byte[8]	.byte[9]	.byte[10]	.byte[11]	.byte[12]	.byte[13]	.byte[14]	.byte[15]
.nibble	.nibble	.nibble	.nibble	.nibble	.nibble	.nibble	.nibble	.nibble	.nibble	.nibble	.nibble	.nibble	.nibble	.nibble	.nibble
0	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60
64	68	72	76	80	84	88	92	96	100	104	108	112	116	120	124

Figure 3-1 Vector-Scalar Register

Operations are executed in parallel across all elements in a VSR, supporting Single Instruction, Multiple Data (SIMD) computations. Depending on the data type—byte, halfword, or word—elements are interpreted as signed, unsigned, single-precision, or double-precision.

VSX supports numerous operations such as Vector Splat, Vector Permute, and Vector Arithmetic instructions, which are critical for AI workloads.

For AI-related use cases, computations like vector sum and vector multiply-add operations are significant for accelerating linear algebra routines used in deep learning frameworks. These optimizations allow Power11 to handle large-scale tensor operations efficiently, reducing latency and improving throughput for inference and training.

Example 3-1 demonstrates how VSX instructions accelerate a common AI computation—calculating the squared Euclidean distance between two vectors.

*Example 3-1 Example code*

---

```
for (i = 0; i < d; i++) {  
    const float tmp = x[i] - y[i];  
    res += tmp * tmp;  
}
```

---

When compiled for Power11 with VSX optimization, this loop translates into instructions like those shown in Example 3-2.

*Example 3-2*

---

```
lvs      v2, r24, r29      // Load vector  
xvsubsp  vs46, vs28, vs51 // Subtract vectors (single precision)  
xvmaddasp vs1, vs13, vs13 // Multiply-add for accumulation
```

---

These SIMD instructions execute multiple operations in parallel, dramatically improving performance compared to scalar execution.

## 3.3 Training of Machine Learning Models

Training is the core phase of any machine learning (ML) workflow, where models learn patterns, structures, or relationships within data. The goal of training is to adjust model parameters such that the model can generalize from the training data to unseen examples. This process typically involves iteratively minimizing a loss function through optimization techniques such as stochastic gradient descent (SGD) or its variants.

Depending on the model architecture, dataset size, and computational resources, training can range from seconds on a personal computer to weeks on large-scale distributed clusters with specialized hardware accelerators. Modern media portrays large GPU farms which are needed for model training, but model training can even start at small models which can even be trained on personal computers to large LLMs, which need those kinds of resources.

### 3.3.1 The Training Process

The training process generally follows these key stages:

1. Data Preparation:

Raw data must be collected, cleaned, and transformed into a numerical representation suitable for model input. This may include normalization, feature extraction, or tokenization (in the case of text).

2. Model Initialization

The model's parameters (weights and biases) are initialized—often randomly—to begin the learning process. Proper initialization is critical for efficient convergence.

### 3. Forward Pass

Input data is propagated through the model to produce an output (prediction). This output is compared to the true label using a predefined **loss function** (e.g., mean squared error for regression, cross-entropy for classification).

### 4. Backward Pass (Backpropagation)

The model computes gradients of the loss function with respect to its parameters. These gradients indicate how the parameters should be updated to minimize the loss.

### 5. Parameter Update

Optimization algorithms such as **SGD**, **Adam**, or **RMSProp** update model parameters iteratively. The process repeats across many epochs until convergence.

## 3.3.2 Computational Complexity and Hardware Requirements

The computational demands of training machine learning models vary greatly depending on the type of model, its size, and the volume of data being processed. While simple models can be trained quickly on standard CPUs, more complex architectures—especially deep learning models—often require specialized hardware such as Graphical Processing Units (GPUs), or Tensor Processing Units (TPUs) to achieve reasonable training times.

Below is an overview of common model types, their relative complexity, and the hardware resources typically needed for efficient training.

### Linear and Logistic Regression

Linear and logistic regression models are among the simplest forms of machine learning. They involve learning a small number of parameters to fit straight-line (or logistic curve) relationships between input features and the target output.

- ▶ **Training Effort:** Low — typically completes in seconds to minutes.
- ▶ **Hardware Requirements:** Standard CPUs are sufficient, even for relatively large datasets. Training can be easily performed on a laptop or a single server.
- ▶ **Scalability:** These models scale well to large datasets, but they struggle with complex nonlinear relationships.

### Decision Trees and Ensemble Models

Decision trees divide data into branches based on feature values, while ensemble methods like Random Forests or Gradient Boosted Trees build many trees to improve accuracy. These models can handle more complex patterns than linear models but require more computation and memory.

- ▶ **Training Effort:** Moderate — seconds to hours depending on the number and depth of trees.
- ▶ **Hardware Requirements:** Multi-core CPUs are generally sufficient. Training benefits from parallel processing, so systems with multiple cores or clusters can speed things up. GPUs are rarely necessary.
- ▶ **Scalability:** Ensemble models handle moderate to large datasets efficiently but can become memory-intensive when hundreds or thousands of trees are built.

## Support Vector Machines (SVMs)

SVMs are powerful for smaller datasets and can model complex decision boundaries, especially when using nonlinear kernels. However, they can become computationally expensive as data size grows.

- ▶ **Training Effort:** High for large datasets — can take hours or even days.
- ▶ **Hardware Requirements:** High-performance CPUs with large memory are preferred. GPUs are typically not used unless special implementations are employed.
- ▶ **Scalability:** Best suited for smaller datasets (up to tens or hundreds of thousands of samples).

## Neural Networks and Deep Learning Models

Neural networks—especially deep architectures such as Convolutional Neural Networks (CNNs) and Transformers (used in Large Language Models)—are significantly more computationally demanding. They can contain millions or even billions of parameters and often require large datasets to perform well.

- ▶ **Training Effort:** Very high — can take hours to weeks depending on model size and data volume.
- ▶ **Hardware Requirements:**
  - GPUs (Graphics Processing Units) are the standard hardware for deep learning, as they can perform many matrix operations in parallel.
  - Distributed training across multiple GPUs or nodes is common for very large models.
- ▶ **Scalability:** Excellent, but requires significant investment in hardware and optimization techniques (e.g., mixed-precision training, data parallelism).

### 3.3.3 Hardware and Energy Considerations

The choice of hardware significantly impacts both training speed and energy consumption:

- ▶ CPUs are flexible and cost-effective for small to medium-scale models.
- ▶ GPUs accelerate training of large neural networks by orders of magnitude, especially in computer vision and natural language processing.

Energy efficiency is becoming an important factor in model training, as large models can consume megawatt-hours of electricity. Efficient use of hardware resources, model compression, and optimization techniques are key to sustainable AI development.

### 3.3.4 Summary

The computational complexity of training machine learning models varies widely across model classes. Simpler models such as linear regressors can be trained in seconds on modest hardware, while deep neural networks may require thousands of GPU hours and vast energy consumption.

Understanding these computational requirements is crucial for system architects and data scientists when designing scalable AI infrastructure. Efficient training not only depends on algorithmic optimization but also on leveraging appropriate hardware and parallelization strategies to meet performance and cost objectives.





# IBM Spyre

The IBM Spyre Accelerator is a purpose-built, enterprise-grade solution designed to deliver scalable performance for complex AI workloads, including generative AI use cases. Each Spyre module is implemented as a PCIe card and integrates 32 dedicated accelerator cores, enabling high-throughput, low-latency AI computation. Co-developed by IBM Research and IBM Infrastructure, the Spyre architecture emphasizes efficiency by allowing data to flow directly between compute engines, minimizing unnecessary memory transfers and reducing energy consumption.

Spyre supports a range of low-precision numeric formats which significantly reduce memory footprint and power usage while maintaining accuracy for AI inference tasks. This makes it especially well suited for deploying large-scale AI models in enterprise environments.

When paired with IBM Power systems, the Spyre Accelerator enhances AI performance by offloading intensive computations from general-purpose CPUs. Unlike traditional architectures where data and instructions are frequently shuttled between processor and memory, Spyre's tightly integrated design streamlines execution, resulting in greater efficiency, scalability, and security for AI-driven enterprise transformation.

The following topics are covered in this chapter:

- ▶ Introduction to IBM Spyre
- ▶ Turnkey AI with Spyre
- ▶ Spyre adapter design
- ▶ Power Implementation
- ▶ Examples and supported workloads
- ▶ Understanding LLM Performance
- ▶ Hybrid Compute - Spyre and CPU
- ▶ Futures

## 4.1 Introduction to IBM Spyre

The IBM Spyre Accelerator for Power is designed to deliver exceptional performance for AI workloads, significantly outperforming traditional CPU-based architectures in both speed and efficiency. In conventional computing models, data and instructions are frequently transferred between the processor and memory. Spyre takes a different approach by being optimized for matrix and vector computations, which are fundamental to modern AI.

This architecture enables direct data transfer between compute engines, reducing memory overhead and latency. As a result, AI models execute with lower energy consumption and improved throughput. These advantages make Spyre particularly effective for large-scale inference and training tasks where memory bandwidth and power efficiency are critical.

GPUs have been the dominant choice for accelerating AI workloads, but they introduce challenges that impact both performance and cost.

- ▶ **Latency Issues:** GPU-based systems often rely on external interconnects and discrete memory pools, which can lead to higher latency when moving data between compute units and system memory. This becomes a bottleneck for real-time inference and large-scale training.
- ▶ **Cost and Complexity:** Deploying GPUs at scale requires additional infrastructure, including specialized cooling, power delivery systems, and high-bandwidth networking to support fast data movement between accelerator nodes. These factors increase total cost of ownership and operational complexity, especially in enterprise environments.

Spyre addresses these limitations by integrating acceleration directly into the Power architecture, eliminating the need for external interconnects and reducing latency while maintaining a compact, energy-efficient footprint.

Figure 4-1 shows the IBM Spyre Accelerator PCIe Attached Card.


<p>IBM Spyre™ Accelerator PCIe attached card</p> <p>SoC implements IBM's leadership innovations in low-precision AI arithmetic and algorithms</p> 	<p>System on a chip architecture <a href="#">optimized for enterprise AI</a> workloads.</p> <p><a href="#">32 low-power AI</a> cores.</p>	<p>Supports <a href="#">multi-precision for inference &amp; training: FP16/8, INT8</a>.</p> <p>Enabled for <a href="#">Foundation Models</a>.</p>
	<p>Enabled in the <a href="#">Red Hat software stack (OCP AI)</a></p> <p>Supports popular AI Framework and libraries (<a href="#">PyTorch</a>, <a href="#">vLLM</a>)</p>	<p>Implemented in 5nm technology.</p> <p>High performance and <a href="#">low-power design</a> with <a href="#">75W</a> consumption.</p>

Figure 4-1 IBM Spyre Accelerator PCIe Attached Card.

## 4.1.1 Role in AI workload orchestration and optimization

Off-chip accelerators like IBM's Spyre card play a pivotal role in scaling and optimizing AI workloads, especially in heterogeneous computing environments. Their integration into system architecture enables several key advantages:

### 1. Specialized Compute for AI Tasks

Off-chip accelerators are designed to handle specific operations, like matrix multiplications, convolutions, and data movement, far more efficiently than general-purpose CPUs. This specialization allows for:

- Faster inference and training cycles
- Lower power consumption per operation
- Higher throughput for parallelized tasks

### 2. Workload Offloading and Balancing

By offloading compute-intensive portions of AI workloads (e.g., model inference, preprocessing, or feature extraction) to accelerators, systems can:

- Free up CPU resources for orchestration, control logic, or other services
- Achieve better workload distribution across compute nodes
- Reduce contention and bottlenecks in shared memory or I/O paths

### 3. Orchestration Across Heterogeneous Resources

Modern AI orchestration frameworks like OpenShift AI can dynamically route workloads to the most suitable compute resource:

- Route workloads based on model size, latency requirements, and data locality
- Using policies that optimize for energy efficiency, throughput, or cost
- Enabling co-location of AI inference with transactional or database workloads on the same server

### 4. Scalability and Modularity

Off-chip accelerators can be clustered and scaled independently of the host CPU:

- For example, adding multiple Spyre cards in an I/O drawer can scale AI compute linearly
- This modularity supports flexible deployment models, from edge to cloud

## 4.2 Turnkey AI with Spyre

[AI Services for IBM Power](#) provide a turnkey, production-ready framework for deploying enterprise AI workloads on IBM Power11 systems. As part of the IBM Open-Source AI Foundation for Power, AI Services supply pre-built, containerized components that simplify the integration of model inferencing, retrieval-augmented generation (RAG), embedding pipelines, and re-ranking tasks into existing Power-based environments. These services are engineered to run efficiently on the Power11 platform and are fully optimized for the IBM Spyre Accelerator for Power, ensuring high throughput and low latency for modern AI workloads.

### 4.2.1 Architecture and Design Principles

AI Services are designed around a modular, open-source-aligned architecture that enables fast deployment and consistent operational behavior across Power-based data centers. Each service operates as a self-contained deployment unit that bundles an AI model, runtime configuration, and API-driven serving interface. This architecture allows enterprises to adopt

AI incrementally, integrate new model capabilities with minimal customization, and maintain consistent performance as workloads scale.

Key design principles include:

- ▶ Turnkey operation: AI Services require minimal setup and provide out-of-the-box integrations with the Red Hat AI Inference Server and the Spyre Accelerator.
- ▶ Open and extensible: Built on open-source foundations and delivered with support for Granite® models, embedding models, and third-party inferencing frameworks.
- ▶ Optimized for Power11: Each service is tuned to take advantage of Power11's vector processing, high memory bandwidth, and hardware-assisted AI features.
- ▶ Accelerator-aware: Seamless integration with the Spyre Accelerator allows services to dynamically route inference workloads and leverage FP16 execution, continuous batching, and multi-card scaling.

## 4.2.2 Integration with IBM Spyre Accelerator for Power

AI Services are tightly integrated with the Spyre Enablement Stack for Power, including drivers, vLLM-based backends, and optimized inferencing runtimes. This integration provides:

- ▶ Direct accelerator access through VFIO and zero-copy data movement
- ▶ Continuous batching support for increased throughput on LLM workloads
- ▶ Multi-card scaling for RAG pipelines and large-context inferencing
- ▶ Model caching and runtime optimizations that reduce start-up and inference latency

This synergy ensures that AI workloads benefit fully from the Spyre hardware capabilities while maintaining predictable performance across diverse use cases.

## 4.2.3 Supported AI Use Cases

AI Services provide pre-configured implementations for several high-value enterprise AI tasks. These services use IBM Granite and other supported open-source models to deliver reliable, production-grade functionality:

- ▶ Large Language Model (LLM) Inference: High-speed text generation, summarization, and conversational tasks using Granite instruct models.
- ▶ Embedding Generation: Vectorization of documents for semantic search, knowledge-base indexing, and retrieval-augmented generation pipelines.
- ▶ RAG Pipelines: End-to-end retrieval-augmented generation utilizing embedding services, vector databases, and LLM-based answer synthesis.
- ▶ Re-ranking: Contextual ranking of retrieved results using advanced re-ranking models to improve accuracy in search and retrieval systems.
- ▶ Entity Extraction: Structured data extraction from unstructured sources using transformer-based NLP models.

Table 4-1 provides details on the use cases currently supported.

Table 4-1 Use cases supported at initial GA.

Use case	Model name	Batch size	Maximum input context size	Maximum output context size	Number of cards per container
Entity extraction	Granite3.3-8b-instruct	16	3K	3K	1
RAG embedding	Granite-Embedding-125m-English Granite-Embedding-278m-multilingual	Up to 256	512	Vector of size 768	1
RAG embedding	Granite-Embedding-30m-English Granite-embedding-107m-multilingual	Up to 256	512	Vector of size 384	1
RAG inferencing	Granite3.3-8b-instruct	32	32K (batch * context equals less than 128K)	32K (batch * context equals less than 128K)	4
Reranker	beg-re-ranker-v2-m3	Up to 4	8K	8K	1

Each AI Service provides APIs, configuration examples, and deployment patterns aligned with the workloads supported by Spyre and Red Hat AI Inference Server.

#### 4.2.4 Deployment and Operations

AI Services are delivered as containerized artifacts that can be deployed in several ways, including through standalone Pod-man containers for development, testing, and lightweight environments, or through Pod-man Quad lets for production scenarios that require automated lifecycle management. They also integrate directly with the Red Hat AI Inference Server, which supports model downloads, environment preparation, and resource delegation. Each service includes all necessary runtime logic to streamline operations, making it easier to scale workloads across multiple Spyre accelerators, monitor inference performance through the Spyre metrics infrastructure, manage logs and diagnostics using `systemd`, `journald`, and Podman tools, and apply SELinux and system-level security controls consistent with Red Hat best practices.

AI Services provide a turnkey approach that significantly reduces the complexity of adopting enterprise-grade AI. Organizations can deploy fully functional AI pipelines using minimal configuration, accelerating time-to-value and lowering operational barriers. This turnkey capability includes:

- ▶ Pre-integrated models and serving runtimes
- ▶ Pre-defined API interfaces for inferencing and data processing
- ▶ Compatibility with enterprise governance, security, and observability frameworks
- ▶ Automated scaling and consistent performance on Power11 infrastructure

Enterprises can embed AI directly into existing business processes, leverage their Power11 infrastructure for high-volume inferencing, and adopt generative AI solutions without extensive re-architecture, while the combined use of open-source software and high-performance hardware enables them to securely run advanced workloads such as Retrieval-Augmented Generation (RAG) directly where their business-critical data resides. For more information on AI

Services provided by the IBM Open-Source AI foundation for Power consult the documentation at <https://www.ibm.com/docs/en/aishervices>.

### 4.3 Spyre adapter design

The IBM Spyre PCIe Gen5 x16 Artificial Intelligence (AI) Accelerator Adapter is a high performance PCI Express (PCIe) Generation 5 x16 adapter purpose-built for AI workloads. At its core is a Samsung 5LPE chip that integrates 32 dedicated accelerator cores and 128 GB of LPDDR5 memory, delivering the compute power and bandwidth required for advanced AI processing. The Spyre adapter is supported in IBM Z17 and IBM Power11 systems.

Spyre delivers powerful compute capabilities within a compact, power-efficient 75W envelope, making it ideal for data center environments where energy efficiency and performance density are critical. Each Spyre card consists of 34 AI accelerator cores, of which 32 are active cores. Together, they collectively deliver over 300 TOPS (trillions of operations per second) of compute power<sup>1</sup>. It supports a wide range of numeric formats making it highly optimized for modern AI inference workloads.

Each AI accelerator core consists of two corelets coupled to a L3 scratchpad memory of 2MB. Each corelet has an 8x8 PT (Processing Tile) array of highly optimized floating point processors tuned for AI/DL workloads in various precisions such as FP8, and FP16.

Matrices are divided into smaller chunks, called tiles, which are stored in the PT array and loaded into memory slices within the core. The PT array serves as the primary compute structure, executing high-throughput Multiply and Accumulate (MAC) operations on these tiles. The Processing Elements (PE) and the SFP (Special Function PE) array handle auxiliary math functions required by AI models—such as activations, normalization layers, and other non matrix-multiply operations. A 128 byte data ring interconnects all AI accelerator cores with LPDDR5 memory, providing high bandwidth movement of tile data across the subsystem<sup>2</sup>. Figure 4-2 shows this architecture.

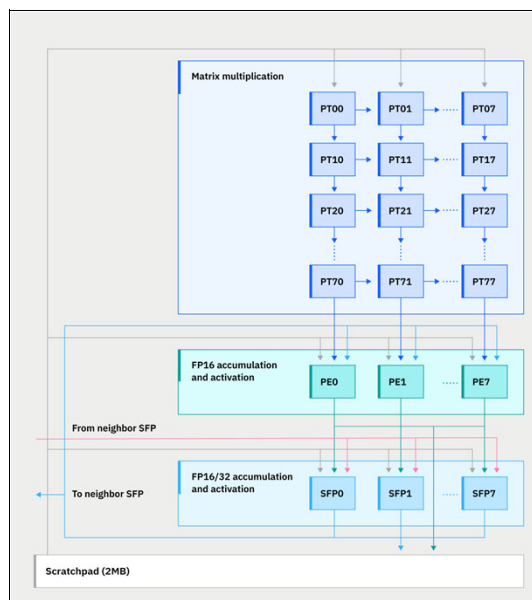


Figure 4-2 Spyre accelerator architecture

<sup>1</sup> <https://www.nextplatform.com/2024/08/27/ibm-shows-off-next-gen-ai-acceleration-on-chip-dpu-for-big-iron/>

<sup>2</sup> [https://hc2024.hotchips.org/assets/program/conference/day1/04\\_HC2024.IBM.CBerry.final.pdf](https://hc2024.hotchips.org/assets/program/conference/day1/04_HC2024.IBM.CBerry.final.pdf)

With 128GB of high-bandwidth LPDDR5 memory, Spyre is well-suited for memory-intensive tasks such as large language model (LLM) inference. The software stack is fully compatible with leading AI frameworks like PyTorch and vLLM, enabling seamless integration into existing AI pipelines.

Spyre is designed for scalability. Up to eight cards can be clustered within a single I/O drawer, offering a combined 256 accelerator cores and 1 TB of memory. This configuration provides 1.6 TB/s aggregate memory bandwidth, significantly boosting capacity for demanding AI applications. The software stack supports model sharding across multiple cards, enabling efficient parallelism, reduced latency, and support for larger model sizes. For example, smaller encoder models can be executed on a single Spyre card, while larger decoder models, such as Granite-3.3-8B-Instruct, scale effectively across four cards, ensuring optimal performance and resource utilization.

## 4.4 Power Implementation

The Spyre card enhances the capabilities of Power11 based systems by providing a low-power, high-efficiency acceleration path tailored for memory-intensive and compute-heavy workloads. Leveraging the advanced virtualization and workload consolidation features of IBM Power systems, Spyre-powered AI applications can be seamlessly collocated with mission-critical services—such as databases and transactional systems—on the same physical server. This architectural proximity reduces latency, boosts throughput, and eliminates the overhead typically associated with cross-node or cross-platform communication. Figure 4-3 illustrates the overall architecture and integration of the Spyre cards within the IBM Power System environment. A total of eight Spyre cards are attached within a dedicated I/O drawer, which is physically and logically connected to the IBM Power System. Communication between the IBM Power System and the Spyre cards is facilitated through a dedicated Linux LPAR.

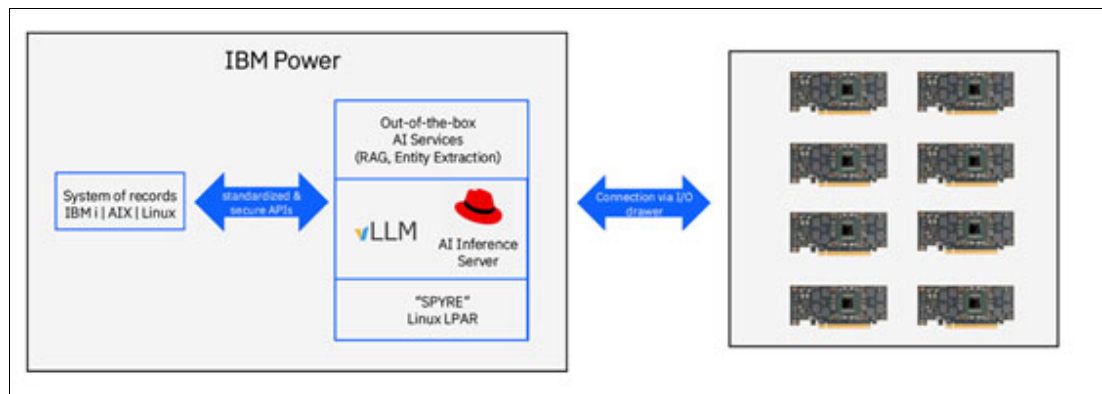


Figure 4-3 Spyre implementation

The Red Hat AI Inference Server is a comprehensive software enablement package required to fully leverage the capabilities of the Spyre Accelerator within higher-level AI software stacks on IBM Power systems. It provides all the essential components needed to integrate Spyre into enterprise AI workflows, ensuring seamless operation and optimized performance. Built on technologies like vLLM and model compression, Red Hat AI Inference Server streamlines deployment of large language models for high throughput, low latency, and cost-efficient performance across hybrid cloud environments. Running it on Power processor-based servers amplifies these benefits with high availability, advanced security features, and hardware-accelerated inferencing, built to helping organizations reduce latency, increase efficiency, and scale AI workloads with confidence. For clients using AI on Power, this means

a consistent, enterprise-ready foundation for bringing generative AI into core business processes.

The stack includes:

- ▶ a vLLM Inference server with a PyTorch backend tailored for Spyre, enabling efficient execution of AI models within familiar development environments.
- ▶ Optimized runtime libraries that manage execution flow and resource utilization across the accelerator cores.
- ▶ A robust device driver that facilitates communication between the host system and the Spyre hardware.
- ▶ Up-to-date firmware for the Spyre card, ensuring stability, security, and performance enhancements.
- ▶ A collection of precompiled AI models with native Spyre support, allowing users to quickly deploy and test workloads without starting from scratch.

A key advantage of this architecture is that vLLM is OpenAI backend compatible. This means that applications and workflows built around the OpenAI API can run seamlessly on vLLM with minimal adjustments. Because vLLM implements an OpenAI-compatible API, developers can take applications built for the OpenAI cloud and run them locally on a Spyre LPAR or other on-premises hardware with minimal changes.

When moving an application from OpenAI cloud to the vLLM server running on-premise, you only need to change one line – the client initialization with the new *base\_url*. Everything else, including your prompts, message handling, and model calls, remains exactly the same. This demonstrates how effortless it is to switch between cloud and on-premise deployments while leveraging the same AI workflows. Figure 4-4 shows the original code and Figure 4-5 shows the changed code pointing to the Spyre interface.

```
python

from openai import OpenAI

# Create client for OpenAI cloud API
client = OpenAI(api_key="YOUR_OPENAI_API_KEY")

# Example: Chat completion
response = client.chat.completions.create(
    model="gpt-4",
    messages=[
        {"role": "user", "content": "Summarize the key benefits of vLLM."}
    ],
)

print(response.choices[0].message.content)
```

Figure 4-4 Original API call

```
python

from openai import OpenAI

# Create client pointing to vLLM server running on-prem
client = OpenAI(base_url="http://ip-of-spyre-lpar:8000/v1") # vLLM endpoint

# Example: Chat completion
response = client.chat.completions.create(
    model="gpt-4",
    messages=[
        {"role": "user", "content": "Summarize the key benefits of vLLM."}
    ],
)

print(response.choices[0].message.content)
```

Figure 4-5 API call to Spyre interface

Together, these components form the foundation for deploying scalable, high-performance AI inference on IBM Power platforms. The Inference server is continuously updated on a regular cadence, ensuring that users receive the latest enhancements, including updated drivers, firmware, optimized AI models, and other critical components. This ongoing support ensures that the platform remains current, secure, and performance-optimized for evolving AI workloads.

### Prerequisites and support

A typical Spyre inference environment consists of a Power11 host system equipped with eight or more Spyre accelerator cards, connected via an ENZO PCIe4 Expansion I/O drawer. The host system must run Red Hat Enterprise Linux (RHEL) version 9.6 or later or Red Hat OpenShift and include the following software components:

- ▶ Red Hat AI Inference Server
- ▶ IBM Open-Source AI Foundation for Power

Once this configuration is in place, users can begin running inference workloads supported by the Spyre platform.

## 4.4.1 Physical implementation

For the IBM Power environment, this adapter is supported exclusively in ENZO PCIe Gen4 expansion drawers configured with ENZA fanout modules, ensuring optimal connectivity and system integration for enterprise-scale deployments.

Some key design concepts for the Spyre adapter are:

- ▶ Advanced Process Technology
- ▶ Built using 5nm fabrication, Spyre combines high performance with a low-power design, operating at only 75W.
- ▶ Optimized AI Compute:
  - Includes 32 low-power AI cores for parallel processing and scalability.
  - Provides multi-precision support for inference offering flexibility across workloads.
- ▶ Foundation Model Enablement
 

Fully integrated with the Red Hat OpenShift AI (OCP AI) software stack, enabling enterprise deployment of foundation models.
- ▶ Broad Framework Compatibility
 

Supports widely used AI frameworks and libraries such as PyTorch and vLLM, allowing developers to leverage existing tools without additional complexity.

Table 4-2 shows the Technical Specifications of the IBM Spyre AI Accelerator Adapter.

*Table 4-2 Technical Specifications of the IBM Spyre AI Accelerator Adapter.*

Description	Item
Adapter FRU Number	03PN188
I/O Bus Architecture	PCIe Gen5 x16
Operating Voltage	3.3 V, 12 V.
Form Factor	Full-height, half-length.

Description	Item
Maximum Supported Adapters	Up to 4 per ENZA fanout module; up to 8 total per ENZO PCIe Gen 4 expansion drawer.

This dedicated, enterprise-grade AI acceleration chip sits on a 75W PCIe adapter, surrounded by 128 GB of LPDDR5 memory to hold a wide variety of LLMs in support of the heterogeneous workloads that are typically seen on IBM Power servers. Figure 4-6 shows the IBM Power AI Acceleration Chip on 75W PCIe Adapter with 128 GB LPDDR5 Memory.



Figure 4-6 IBM Power AI Acceleration Chip on 75W PCIe Adapter with 128 GB LPDDR5 Memory.

A single Spyre adapter does not provide sufficient compute capacity for most enterprise AI workloads, so IBM has designed a solution that leverages existing I/O expansion technology to scale performance. This approach uses an ENZO PCIe Gen4 I/O Expansion Drawer populated with eight Spyre adapters, forming a tightly integrated logical cluster. The firmware on these eight adapters manages workload distribution and coordinates data transfers across the cluster, ensuring efficient parallel processing. From a software perspective, this cluster appears as a single high-performance compute engine, delivering an aggregate of 1 TB of on-card memory and 1.6 TB/s of memory bandwidth. This architecture enables organizations to run large-scale AI models and foundation workloads with minimal latency and maximum throughput, while maintaining compatibility with the Power11 platform and its enterprise software stack. This is shown in Figure 4-7.

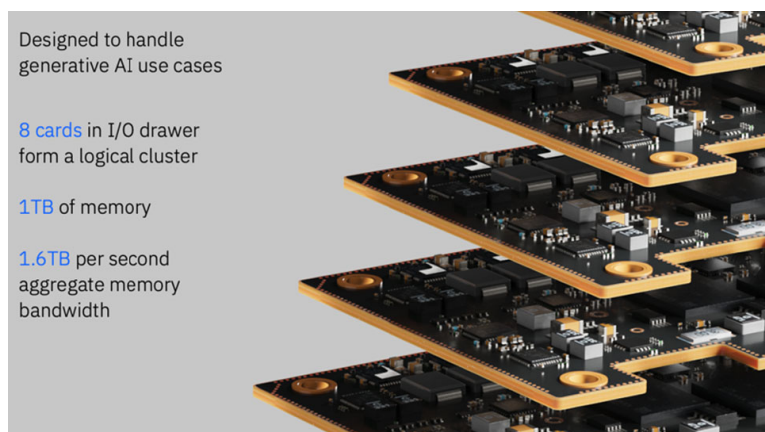


Figure 4-7 Spyre cluster in an expansion drawer.

## 4.4.2 Expansion Drawer support

The Spyre-enabled expansion drawer can be attached to any of the announced IBM Power11 servers, with the number of supported drawers varying by system model. At launch, Spyre will be offered in a fixed configuration that includes eight Spyre adapters installed in a PCIe Gen4 I/O Expansion Drawer. This configuration creates a high-density accelerator solution designed for AI workloads.

The supported Power11 servers are shown in Table 4-3

Table 4-3 Supported Power11 servers

Server Model	Number of drawer supported	Number of adapters supported
L1122	1	8
L1124	1	8
S1122	1	8
S1124	1	8
E1150	1	8
E1180	1 or 2	8 or 16

To meet the additional power and cooling requirements of this configuration, new components such as upgraded power supplies and specialized fanout modules are integrated into the expansion drawer. These enhancements ensure reliable operation and optimal performance when hosting multiple high-power Spyre adapters.

The initial supported software and hardware include:

- Firmware: FW1110.10
- Hardware Management Console (HMC): Version HMC1111
- Operating System: Red Hat Enterprise Linux (RHEL) 9.6
- Spyre Software Stack Container
- OpenShift AI (Tech Preview: Q4 2025, General Availability: Q1 2026)

### ENZO PCIe Gen 4 I/O Expansion Drawer

The ENZO PCIe Gen 4 I/O Expansion Drawer is designed to support high-performance PCIe adapters in IBM Power systems. It accommodates up to eight adapters using two 6-slot fanout modules, each identified by feature codes ENZA or ENZF.

### Adapter Placement and Slot Priorities

Figure 4-8 illustrates the rear view of the ENZO PCIe Gen4 expansion drawer, highlighting the location codes for adapter slots within the PCIe Gen4 6-slot fanout module.

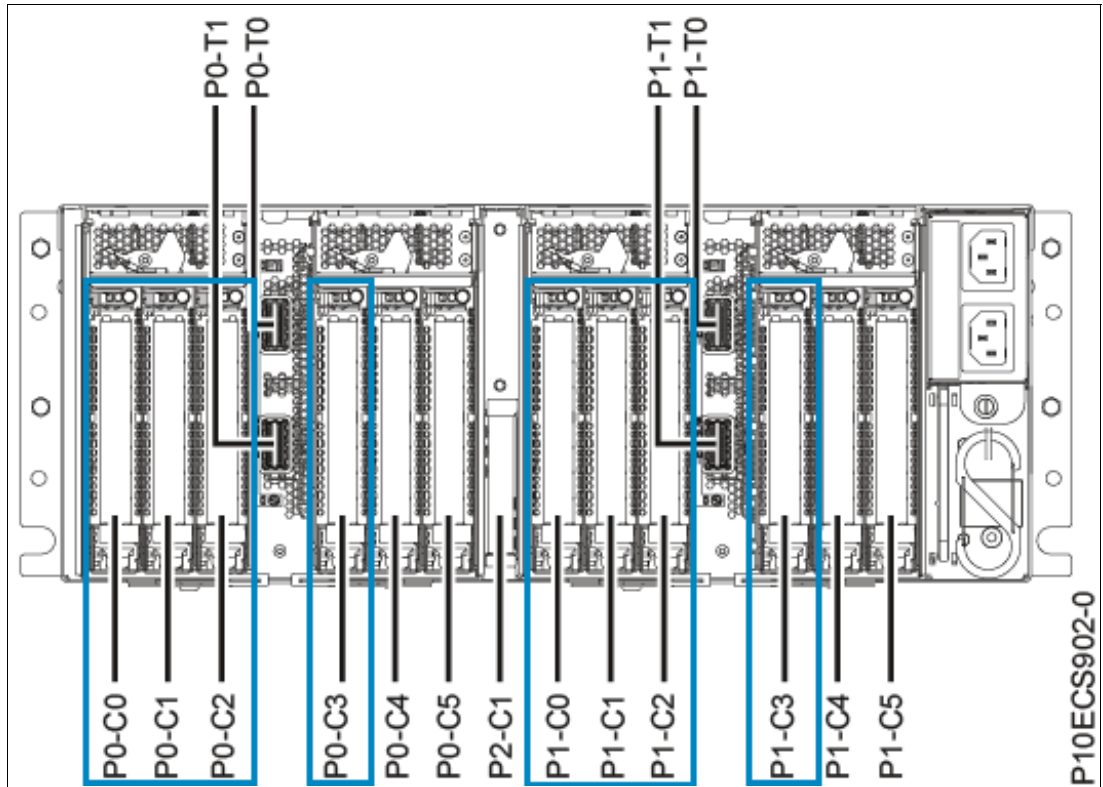


Figure 4-8 The rear view of the ENZO PCIe Gen 4 expansion drawer.

For optimal performance and compatibility with the IBM Spyre AI Accelerator Adapter, adhere to the following slot priority rules:

► Slot Priorities:

- P0-C0 through P0-C3
- P1-C0 through P1-C3

These slots must be fully populated when installing IBM Spyre adapters.

► Fanout Module Compatibility:

- ENZF fanout modules support all adapter types except the Feature Code ECSE adapter.
- ENZA fanout modules support only the Feature Code ECSE adapter.

Table 4-4 on page 65 illustrates the slot priorities and maximum supported IBM Spyre cards in ENZO PCIe Gen4 expansion drawer.

Table 4-4 Slot Priorities and Maximum Supported IBM Spyre cards in ENZO PCIe Gen4 Expansion Drawer

Feature code <sup>a</sup>	Description	ENZO PCIe4 expansion drawer	
		Slot priorities <sup>b</sup>	Maximum number of adapters supported <sup>c</sup>
ECSE	PCIe5 x16 AI Accelerator adapter (FC ECSE; CCIN 2E1F); Adapter part number: 03PN188	P0-C0 through P0-C3 and P1-C0 through P1-C3 must be fully populated	

- When configured with ENZA fan-out modules, the ENZO PCIe Gen 4 expansion drawer supports only the FC ECSE adapter
- The slot priority sequence is based on the ENZO PCIe4 expansion drawer configured with two PCIe4 6-slot fanout modules
- The maximum number of adapters supported per PCIe4 6-slot fanout module

**Note:** When the PCIe Gen 5 x16 AI Accelerator Adapter (Feature Code: ECSE; CCIN: 2E1F) is installed in the ENZO PCIe Gen 4 I/O Expansion Drawer, the adapter will operate at PCIe Gen4 speeds, resulting in a downgrade from PCIe Gen5 to Gen4 performance.

### Limitations and Requirements

Consider these limitations and requirements for the Spyre Accelerator and the ENZA fanout module.

- ▶ Feature ENZA (PCIe4 FOM) cannot be mixed with Feature ENZF in the Expansion Drawer Feature ENZO.
- ▶ Feature ENZA must be included in the order to enable the IBM Spyre Accelerator for Power (Feature ECSE).
- ▶ Feature ENZA supports only IBM Spyre Accelerator for Power (Feature ECSE); no other I/O adapters are supported.
- ▶ Each Feature ENZA supports up to four IBM Spyre Accelerator for Power (Feature ECSE) adapters.
- ▶ The Expansion Drawer Feature ENZO is identified in manufacturing with a distinct CCIN when configured for accelerator functionality.
- ▶ Cable Card attachment for Feature ECSE in ENZO requires installation in x16 system slots.
- ▶ Upgrades to enable accelerator functionality in the field for ENZO are not supported.
- ▶ The ENZO Accelerator Configuration is not supported with eSCMs processor modules featuring 4-core or 10-core configurations on the S1122 server.
- ▶ The ENZO Accelerator Configuration is not supported on systems with a single 16-core processor module on the S1124 server.
- ▶ One TB of memory is required in when ordering the Spyre Accelerator.

### Installing the IBM Spyre adapter in ENZA Fanout Module

To install the PCIe Gen5 x16 AI Accelerator Adapter into an ENZA fanout module within an ENZO PCIe Gen4 expansion drawer:

- Insert the Adapter:** With the tailstock clamp open, firmly insert the adapter into the tailstock retaining channel.
- Secure the Adapter:** Rotate the adapter upward into position and close the tailstock clamp as shown in Figure 4-9 on page 66.

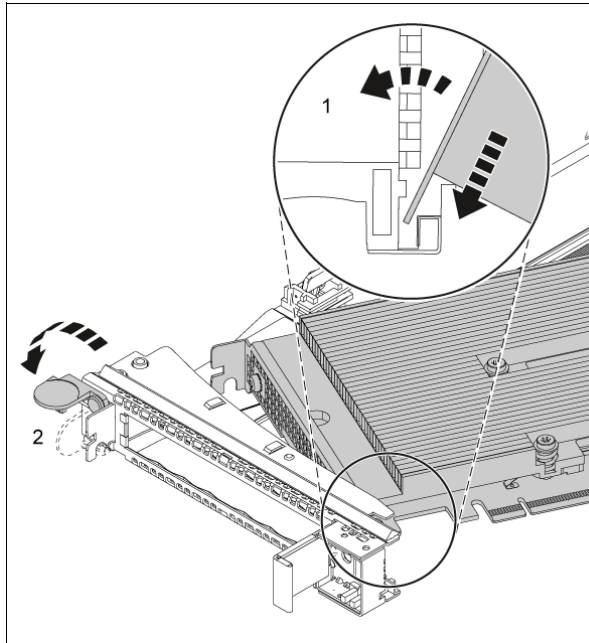


Figure 4-9 Placing the ECSE Adapter in the Cassette

3. **Lock the Retainers:** Slide the adapter retainers toward the adapter, ensuring it remains parallel to the cassette. Lock the top retainers by rotating them to a vertical position.
4. **Final Clip Adjustment:** Slide the double clip assembly until it aligns flush with the card's tabbed feature, then rotate the tailstock clamp to the closed position.
5. **Verify Retainer Clips:** Ensure all retainer clips are in the locked (vertical) position. When in the horizontal position, clips are unlocked and can be adjusted. This is shown in Figure 4-10.

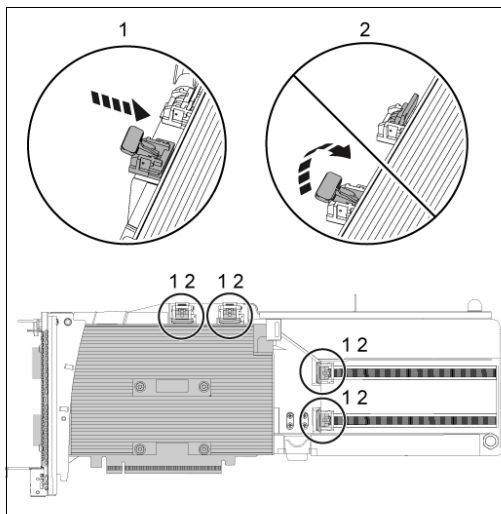


Figure 4-10 Locking the Retainer Clips

## Verifying the link speed

To confirm the operational PCIe link speed and device details, use the command shown in Example 4-1.

### Example 4-1 Validating the configuration of the Spyre adapter

---

```
# lspci -vvv -d 1014:06a7
182:60:00.0 Processing accelerators [1200]: IBM Spyre Accelerator [1014:06a7] (rev 02)
  Physical Slot: U5B61.001.WZS09NJ-P1-C1
  Device tree node: /sys/firmware/devicetree/base/pci@800000020000182/pci1014,6a7@0
  Control: I/O- Mem+ BusMaster- SpecCycle- MemWINV- VGASnoop- ParErr+ Stepping- SERR+ FastB2B-
DisINTx-
  Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR-
<PERR- INTx-
  Interrupt: pin A routed to IRQ 0
  NUMA node: 0
  IOMMU group: 0
  Region 0: Memory at 4108200000 (64-bit, prefetchable) [size=4M]
  Region 2: Memory at 4100000000 (64-bit, prefetchable) [size=2G]
  Region 4: Memory at 4108000000 (64-bit, prefetchable) [size=32M]
  Capabilities: [40] Power Management version 3
    Flags: PMEClk- DSI- D1- D2- AuxCurrent=375mA PME(D0-,D1-,D2-,D3hot-,D3cold-)
    Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=0 PME-
  Capabilities: [50] MSI: Enable- Count=1/8 Maskable+ 64bit+
    Address: 0000000000000000 Data: 0000
    Masking: 00000000 Pending: 00000000
  Capabilities: [70] Express (v2) Endpoint, MSI 00
    DevCap:MaxPayload 512 bytes, PhantFunc 0, Latency L0s unlimited, L1 unlimited
      ExtTag+ AttnBtn- AttnInd- PwrInd- RBE+ FLReset+ SlotPowerLimit 0.000W
    DevCtl:CorrErr- NonFatalErr+ FatalErr+ UnsupReq+
      RlxdOrd+ ExtTag+ PhantFunc- AuxPwr- NoSnoop- FLReset-
      MaxPayload 512 bytes, MaxReadReq 4096 bytes
    DevSta:CorrErr- NonFatalErr- FatalErr- UnsupReq- AuxPwr- TransPend-
    LnkCap:Port #0, Speed 32GT/s, Width x16, ASPM L1, Exit Latency L1 <64us
      ClockPM- Surprise- LLActRep- BwNot- ASPMOptComp+
    LnkCtl:ASPM Disabled; RCB 64 bytes, Disabled- CommClk-
      ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
    LnkSta:Speed 16GT/s (downgraded), Width x16 (ok)
      TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-
    DevCap2: Completion Timeout: Range ABCD, TimeoutDis+ NROPrPrP- LTR-
      10BitTagComp+ 10BitTagReq- OBFF Not Supported, ExtFmt- EETLPPrefix-
      EmergencyPowerReduction Not Supported, EmergencyPowerReductionInit-
      FRS- TPHComp- ExtTPHComp-
      AtomicOpsCap: 32bit- 64bit- 128bitCAS-
    DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis+ LTR- OBFF Disabled,
      AtomicOpsCtl: ReqEn-
    LnkCap2: Supported Link Speeds: 2.5-32GT/s, Crosslink- Retimer+ 2Retimers+ DRS-
    LnkCtl2: Target Link Speed: 32GT/s, EnterCompliance- SpeedDis-
      Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-
      Compliance De-emphasis: -6dB
    LnkSta2: Current De-emphasis Level: -6dB, EqualizationComplete+ EqualizationPhase1+
      EqualizationPhase2+ EqualizationPhase3+ LinkEqualizationRequest+
      Retimer- 2Retimers- CrosslinkRes: Upstream Port
  Capabilities: [100 v2] Advanced Error Reporting
    UESta:DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalftTLP- ECRC- UnsupReq-
ACSViol-
```

```

UEmsk:DLP- SDES- TLP- FCP- CmplTtO- CmpltAbrt- UnxCmplt- RxOF- MalftTLP- ECRC- UnsupReq-
ACSViol-
UESvrt:DLP+ SDES+ TLP- FCP+ CmplTtO- CmpltAbrt- UnxCmplt- RxOF+ MalftTLP+ ECRC- UnsupReq-
ACSViol-
CESta:RxErr- BadTLP- BadDLLP- Rollover- Timeout- AdvNonFatalErr-
CEmsk:RxErr- BadTLP- BadDLLP- Rollover- Timeout- AdvNonFatalErr+
AERCap:First Error Pointer: 00, ECRCGenCap+ ECRCGenEn+ ECRCChkCap+ ECRCChkEn+
  MultHdrRecCap- MultHdrRecEn- TLPPfxPres- HdrLogCap-
  HeaderLog: 00000000 00000000 00000000 00000000
Capabilities: [148 v1] Alternative Routing-ID Interpretation (ARI)
  ARICap:MFVC- ACS-, Next Function: 0
  ARICtl:MFVC- ACS-, Function Group: 0
Capabilities: [158 v1] Secondary PCI Express
  LnkCtl3: LnkEquIntrruptEn- PerformEqu-
  LaneErrStat: 0
Capabilities: [188 v1] Physical Layer 16.0 GT/s <?>
Capabilities: [1b8 v1] Lane Margining at the Receiver <?>
Capabilities: [200 v1] Extended Capability ID 0x2a
Capabilities: [230 v1] Single Root I/O Virtualization (SR-IOV)
  IOVCap:Migration-, Interrupt Message Number: 000
  IOVctl:Enable- Migration- Interrupt- MSE- ARIHierarchy-
  IOVSta:Migration-
  Initial VFs: 255, Total VFs: 255, Number of VFs: 0, Function Dependency Link: 00
  VF offset: 1, stride: 1, Device ID: 06a8
  Supported Page Size: 00000553, System Page Size: 00000010
  Region 0: Memory at 0000000000000000 (64-bit, prefetchable)
  VF Migration: offset: 00000000, BIR: 0
Capabilities: [270 v1] Vendor Specific Information: ID=0002 Rev=4 Len=100 <?>
Capabilities: [370 v1] Vendor Specific Information: ID=0001 Rev=1 Len=038 <?>
Capabilities: [3a8 v1] Data Link Feature <?>
Kernel driver in use: vfio-pci

```

---

In this example the `lspci` command Lists all PCI devices.

- The `vvv` flag provides verbose output with detailed device capabilities and status.
- The `d 1014:06a7` flag filters for devices with Vendor ID 1014 (IBM) and Device ID 06a7 (IBM Spyre Accelerator). Spyre accelerator cards can use either 1014:06a7 or 1014:06a

In the output, `182:60:00.0` specifies the exact PCI domain, bus, device, and function.

From the command output, look for the following lines to verify link speed:

- `LnkCap:Port #0, Speed 32GT/s, Width x16, ASPM L1, Exit Latency L1 <64us`
- `LnkSta:Speed 16GT/s (downgraded), Width x16 (ok)`

`LnkCap` shows the maximum supported speed (PCIe Gen5: 32 GT/s). `LnkSta` shows the actual operational speed (PCIe Gen4: 16 GT/s), confirming the downgrade.

**Note:** This downgrade is expected behavior when using the ECSE adapter in a PCIe Gen4 drawer. While the adapter supports PCIe Gen5 speeds, it will operate at Gen4 speeds due to the drawer's hardware limitations.

## 4.5 Examples and supported workloads

The following steps detail how to run inferencing with vLLM. Examples for three supported workloads are shown, RAG (Retrieval-Augmented Generation), entity extraction, and embedding. Note that details from the following examples are subject to change.

### Running RAG and entity extraction

1. In order to run RAG or entity extraction workloads, first set the following required environment variables:

AIU_IDS	Space separated list of AIU IDs for the container to use
AIU_WORLD_SIZE	Number of AIU's for the container to use
HOST_MODELS_DIR	Path to model directory on host
VLLM_MODEL_PATH	Path to desired model within container environment
MAX_MODEL_LEN	Max model length/context size, refers to the max sum of prompt length and output tokens
MAX_BATCH_SIZE	Max Batch size of vLLM server

Example 4-2 shows setting the variables for a RAG environment.

#### *Example 4-2 Variables for RAG*

---

```
export AIU_IDS="0301:50:00.0 0302:60:00.0 0303:70:00.0 0304:80:00.0"
export AIU_WORLD_SIZE=4
export HOST_MODELS_DIR=/models
export VLLM_MODEL_PATH=/models/granite-3.3-8b-instruct
export MAX_MODEL_LEN=32768
export MAX_BATCH_SIZE=32
```

---

Example 4-3 shows setting variables for an entity extraction environment.

#### *Example 4-3 Entity extraction settings*

---

```
export AIU_IDS="0301:50:00.0"
export AIU_WORLD_SIZE=1
export HOST_MODELS_DIR=/models
export VLLM_MODEL_PATH=/models/granite-3.3-8b-instruct
export MAX_MODEL_LEN=3072
export MAX_BATCH_SIZE=16
```

---

2. Start the container with the command shown in Example 4-4.

#### *Example 4-4 Starting the container*

---

```
podman run -d \
  --device=/dev/vfio \
  -v ${HOST_MODELS_DIR}:/models \
  -e AIU_PCIE_IDS="${AIU_IDS}" \
  -e VLLM_SPYRE_USE_CB=1 \
  --pids-limit 0 \
  --users=keep-id \
  --group-add=keep-groups \
  --memory 200G \
  --shm-size 64G \
  -p 8000:8000 \
  <container url>:<container tag> \
  --model "${VLLM_MODEL_PATH}" \
  -tp "${AIU_WORLD_SIZE}" \
```

```
--max-model-len "${MAX_MODEL_LEN}" \  
--max-num-seqs ${MAX_BATCH_SIZE}
```

---

3. View the newly created container with the **podman ps** command. The container will begin warming up a vLLM server with the specified shape. Progress can be monitored using the **podman logs <cid>** command, replacing `<cid>` with the correct container ID.
4. Once the vLLM server is up and running, use the **curl** command shown in Example 4-5 to interact with the model.

#### *Example 4-5*

---

```
curl -X POST -H "Content-Type: application/json" -d '{  
  "model": "/models/granite-3.3-8b-instruct",  
  "prompt": "What is the capital of France?",  
  "max_tokens": 50  
}' http://<your_server_ip>:8000/v1/completions | jq
```

---

## Running embedded workloads

The following steps detail how to run the embedded workloads.

1. First, set the following required environment variables shown in Example 4-6.

#### *Example 4-6 Environment variables to be set*

---

AIU\_IDS -> Space separated list of AIU IDs for the container to use  
AIU\_WORLD\_SIZE -> Number of AIU's for the container to use  
HOST\_MODELS\_DIR -> Path to model directory on host  
VLLM\_MODEL\_PATH -> Path to desired model within container environment  
PROMPT\_LENS -> Max length of input prompts in tokens  
BATCH\_SIZES -> Batch size of vLLM server

---

See Example 4-7 for an example.

#### *Example 4-7 Set environment variables*

---

```
export AIU_IDS="0301:50:00.0"  
export AIU_WORLD_SIZE=1  
export HOST_MODELS_DIR=$HOME/models  
export VLLM_MODEL_PATH=/models/granite-embedding-125m-english/  
export PROMPT_LENS=512  
export BATCH_SIZES=4
```

---

2. Start the container with the command shown in Example 4-8.

#### *Example 4-8 Starting the container*

---

```
podman run -d \  
  --device=/dev/vfio \  
  -v ${HOST_MODELS_DIR}:/models \  
  -e AIU_PCIE_IDS="${AIU_IDS}" \  
  -e VLLM_SPYRE_WARMUP_BATCH_SIZES="${BATCH_SIZES}" \  
  -e VLLM_SPYRE_WARMUP_PROMPT_LENS="${PROMPT_LENS}" \  
  --userns=keep-id \  
  --group-add=keep-groups \  
  --pids-limit 0 \  
  --memory 200G \  
  --shm-size 64G \  
  --
```

```
-p 8000:8000 \  
<container url>:<container tag> \  
--model "${VLLM_MODEL_PATH}" \  
-tp "${AIU_WORLD_SIZE}"
```

---

3. View the newly created container with the **podman ps** command. The container will begin warming up a vLLM server with the specified shape. Progress can be monitored using the **podman logs <cid>** command, replacing `<cid>` with the correct container ID.
4. Once the vLLM server is up and running, run the **curl** command shown in Example 4-9 to interact with the model.

*Example 4-9 Interact with the model*

---

```
curl http://<your_server_ip>:8000/v1/embeddings -H "Content-Type:  
application/json" -d '{ "model": "/models/granite-embedding-125m-english/",  
"input": ["What is the capital of Paris?"] }'
```

---

## 4.6 Understanding LLM Performance

Large Language Models operate through two distinct phases during inference. The first phase, prompt intake (also called prefill or prompt processing), occurs when the model receives and processes the entire input prompt simultaneously. During this phase, the model computes attention scores across all input tokens in parallel, building the initial context representation through matrix multiplications across the full prompt length. The second phase, token generation (also called decode or autoregressive generation), happens sequentially where the model generates one token at a time. Each newly generated token is fed back into the model to produce the next token, creating a chain of predictions until the response is complete. This sequential nature fundamentally differentiates token generation from the parallel processing of prompt intake.

The two phases have dramatically different performance bottlenecks due to their operational characteristics. Prompt intake is compute-intensive because it processes many tokens simultaneously, performing large matrix multiplications (batch matrix operations) that fully utilize compute cores and arithmetic units. The key metrics here are FLOPS (floating-point operations per second), card utilization, and throughput measured in tokens processed per second. In contrast, token generation is memory bandwidth-bound because it generates only one token at a time, requiring the model to repeatedly load large weight matrices from memory for each single token prediction. This creates a bottleneck where the accelerator spends more time waiting for data transfer from memory than performing actual computations. The critical metrics for token generation are memory bandwidth utilization (GB/s), latency per token, and the arithmetic intensity ratio, which remains low due to the small batch size of sequential generation.

Leveraging these architectural advantages, Spyre has been specifically optimized for initial use cases such as Retrieval-Augmented Generation (RAG) and entity extraction. A critical differentiator is Spyre's native integration with POWER systems. Since enterprise databases already reside on POWER infrastructure, organizations can execute AI workloads directly at the data source. This approach eliminates the need to migrate data to cloud environments, enabling secure, on-premises AI processing that maintains data sovereignty while delivering low-latency performance.

Figure 4-11 provides some performance guidance comparing Power10 MMA performance to the Spyre adapter.

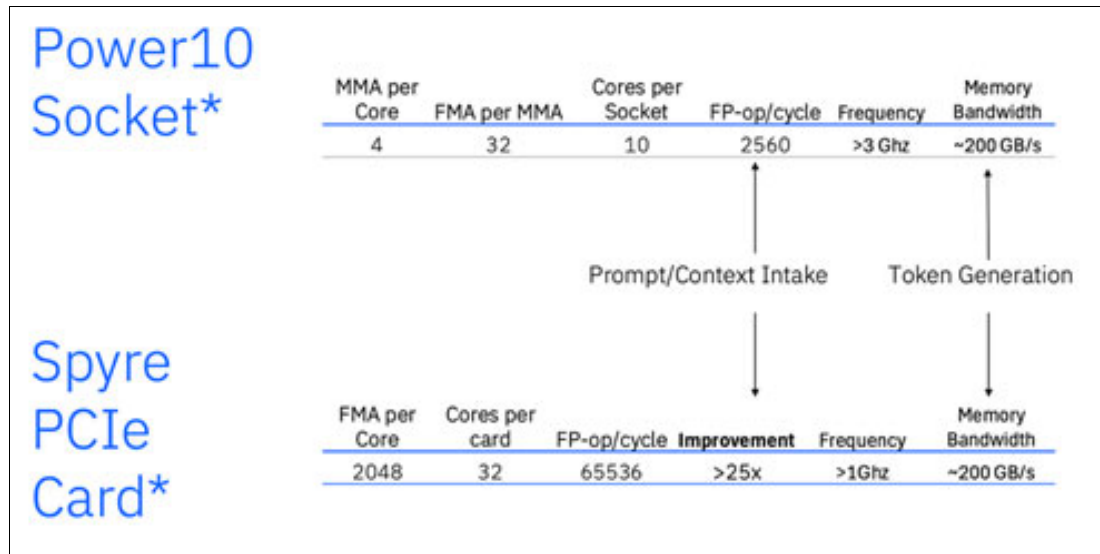


Figure 4-11 Performance guidance

## 4.7 Hybrid Compute - Spyre and CPU

AI systems benefit from a hybrid compute architecture that balances **on-chip** and **off-chip** resources to optimize performance, power efficiency, and scalability. Enhancements in the IBM Power11 processor, particularly in core performance and system capacity, further boost the efficiency of the integrated Matrix Math Accelerator (MMA) and SIMD for inferencing tasks. When combined with the IBM Spyre accelerator, Power11 systems gain additional AI inferencing capabilities, forming a tightly integrated platform for scalable, high-throughput AI workloads.

On-chip workloads are executed directly within the processor, leveraging tightly integrated components such as:

- ▶ Matrix Math Accelerator (MMA) in Power11
- ▶ SIMD units and vector engines
- ▶ On-chip caches and memory

Off-chip workloads are offloaded to external accelerators such as Spyre cards

These workloads typically involve:

- ▶ Large-scale model inference and training
- ▶ Agentic workloads requiring multi-modal reasoning or long-context processing

In Retrieval-Augmented Generation (RAG) pipelines, efficient workload distribution across heterogeneous compute resources is key to achieving optimal performance. Many preprocessing tasks—such as document ingestion, and embedding generation—can be effectively executed on CPUs, freeing up specialized accelerators for more demanding operations. The inference phase, particularly when using large language models (LLMs), is computationally intensive and well-suited for offloading to high-performance accelerators like the Spyre card.

By profiling the end-to-end pipeline, time-consuming stages can be identified and prioritized for off-chip acceleration. This enables intelligent scheduling and dynamic resource allocation, ensuring that each component of the workflow runs on the most appropriate hardware. Such an approach not only balances the computational load but also maximizes throughput, minimizes latency, and improves overall system efficiency. This strategy is especially valuable in production environments where responsiveness and cost-efficiency are critical for scaling AI-powered applications

## 4.8 Futures

IBM's AI acceleration roadmap continues to evolve, with several key enhancements planned to expand capabilities, improve performance, and support emerging AI paradigms:

1. Integration with AI Platforms
  - Watsonx.ai integration will enable seamless deployment, orchestration, and lifecycle management of AI workloads across hybrid cloud environments.
  - Enhanced compatibility with Red Hat and IBM AI stacks will simplify model serving, monitoring, and scaling.
2. Expanded Support for Agentic Workloads
  - New use cases will be added to support agentic AI workloads, including autonomous decision-making systems, multi-agent collaboration, and real-time reasoning engines.
  - Optimizations will target low-latency inference and dynamic task routing.
3. Broader Model Compatibility

Support for a wider range of AI models, including transformer-based architectures, vision-language models, finetuned models and emerging lightweight LLMs.
4. Performance and Precision Enhancements
  - Support for new data types such as FP8 for improved compute efficiency and reduced memory footprint.
  - Continue Hardware enhancements and software-level optimizations to accelerate training and inference pipelines.

Note that IBM has announced a Spyre on POWER Customer Support Plan covering both hardware and software issues which will feature Expert Care tier purchased response objectives. This support package also covers Python packages released as part of the IBM Open-Source AI Foundation for Power repository.



# AI Workload Optimization on Power11

Today, a significant number of artificial intelligence (AI) applications are developed using Python, from the model-training frameworks to the user interfaces that deliver them. For this reason, enhancing the overall Python experience on the IBM Power11 platform was a key design priority from the outset. Each generation of Power technology builds upon the strengths of the previous one, refining performance, efficiency, and developer experience. Power11 continues this evolution, bringing together hardware advancements, compiler innovations, and compatibility with Python runtime enhancements to deliver a faster and more capable environment for modern machine learning and AI workloads.

The Power11 processor introduced notable architectural improvements, including an increased number of cores and expanded Matrix Math Accelerator (MMA) capacity, that can significantly accelerate workloads typically encountered in Python environments. IBM developers have already been working to create a ppc64le-optimized python ecosystem via the devpi project<sup>1</sup> (aligning with the wheels delivery system for the foreseeable future).

This chapter explores how these hardware advances, coupled with recent compiler and library-level enhancements, combine to deliver measurable performance gains for Python workloads on Power11. We will discuss support for AI/ML frameworks (which are crucial for enterprise AI), expand on compatibility with scientific libraries like OpenBLAS, and the improved Developer experience on the Power11 platform.

The following topics are covered in this chapter:

- ▶ Memory bandwidth and cache utilization
- ▶ NUMA-aware scheduling and resource allocation
- ▶ Enhancements in Python performance on Power11
- ▶ Support for AI/ML frameworks
- ▶ Compatibility with OpenBLAS, NumPy, and other scientific libraries
- ▶ Developer productivity and tooling improvements

<sup>1</sup> <https://wheels.developerfirst.ibm.com/>

## 5.1 Memory bandwidth and cache utilization

Spyre's performance on IBM Power11 is driven by a tightly integrated memory and caching strategy that combines accelerator-level LPDDR5 bandwidth with system-level advances in Power11's OMI-based DDR5 subsystem. By pairing high-bandwidth on-card memory, efficient dataflow scheduling, and scalable accelerator cores with a host architecture designed to minimize contention, Power11 ensures that both CPU orchestration and Spyre execution pipelines stay fed with data. This synergy enables consistently high throughput, low latency, and predictable behavior for generative AI and inference workloads across enterprise Power11 deployments.

Memory bandwidth and cache utilization optimizations for Spyre on Power11 come from a combination of architectural and system-level features:

- ▶ On-card high-bandwidth LPDDR5 reduces host memory dependence
- ▶ Scaling to 16 cards increases total bandwidth and memory linearly
- ▶ Continuous batching and dataflow scheduling improve cache reuse
- ▶ 32 accelerator cores reduce memory bottlenecks with localized operands
- ▶ Power11's OMI DDR5 subsystem complements Spyre by reducing CPU-side contention

Together, these innovations let Spyre deliver high throughput for generative AI and inference workloads, while maintaining low latency and predictable performance on enterprise Power11 systems.

Power11 introduces major improvements in its Open Memory Interface (OMI) DDR5 subsystem, achieving high DDR5 bandwidth and lower latency. This allows Spyre's host-side operations (command dispatch, I/O, pre/post-processing) to access memory without starving the CPU or accelerator. [servethehome.com]

Spyre benefits indirectly because:

- ▶ CPU preprocessing and model orchestration can run at higher throughput
- ▶ Memory contention is reduced for data ingestion, staging, and batching

This synergy between Power11's DDR5 memory system and Spyre's LPDDR5 memory design helps improve overall AI workload efficiency.

## 5.2 NUMA-aware scheduling and resource allocation

To get the most performance out of a Power10 or Power11 system when running AI workloads, the AI model must run on a single NUMA node. In the IBM Power context, a NUMA node is a logical representation of a single processor chip, which can be a single-chip module (SCM) or half of a dual-chip module (DCM). Figure 5-1 on page 77 shows a logical diagram of an IBM Power S1022 or S1024 server in a 2-socket configuration.

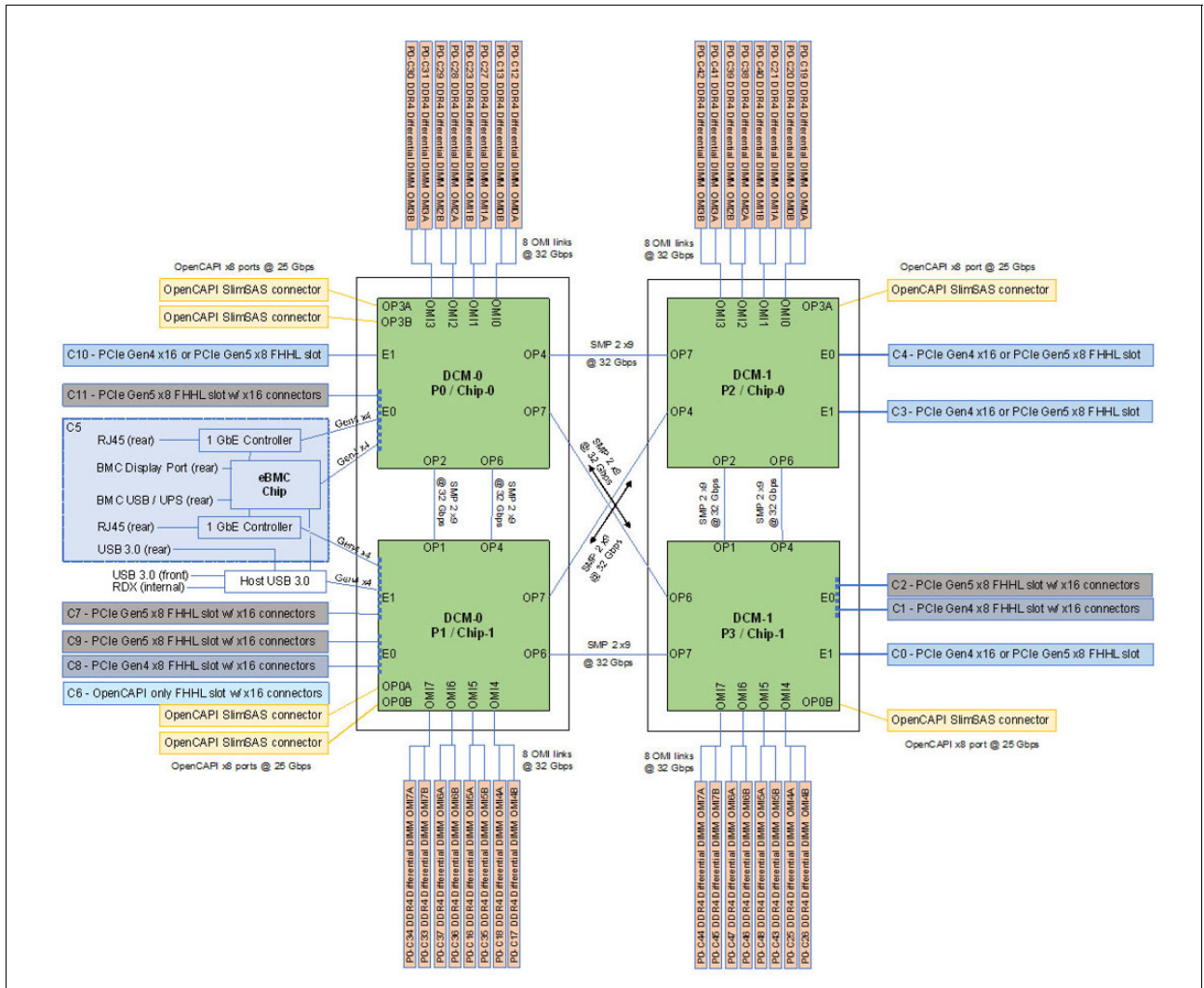


Figure 5-1 Logical diagram of Power S1022 or Power S1024 servers in 2-socket configurations

The two white boxes with black border represent the two sockets in the configuration, whereas the green boxes represent individual processor chips (NUMA nodes). Two chips form a DCM, which is placed on a socket. As you can see, there are lines between each chip representing data cables. These cables are the bottleneck if running AI workload across multiple NUMA nodes.

While the AI model processes a request, many little data packets (messages) are transferred between the physical cores involved in the process. If these packets are sent over the data lanes between individual chips, a significant performance reduction occurs. This performance drop is so severe that running AI workload on a single NUMA node with only six cores is more performant than using a setup of 12 cores across more than one NUMA node.

With this in mind, one has to make sure that an AI model is deployed on a single NUMA node. This can be achieved by sizing and scheduling the LPAR (or Red Hat OpenShift worker node designated for the AI workload) to fit on a single NUMA node. Let's assume the maximum number of cores on a single NUMA node for a given configuration is 8 cores (e.g. a 16-core DCM). The LPAR cannot get more than 8 dedicated cores assigned, otherwise it would not fit on a single NUMA node. The allocated memory should match the memory attached to the

cores of the LPAR to prevent hops. Furthermore, all memory DIMMs should be populated for maximum performance.

Power11 introduced *resource groups*. Resource groups allow you to add specific cores and memory units to a resource group and assign this resource group to an LPAR. This way, a deployment on a single NUMA node can be ensured. Please refer to [Resource Groups on Power11](#) for further information on resource groups.

Power10 does not have the luxury of resource groups. There are two ways to get an LPAR deployed on a single NUMA node:

1. Start the LPAR first on the system.
2. If option 1 is not possible, create dummy LPARs to fill up the NUMA node and start the LPAR afterwards.

## 5.3 Enhancements in Python performance on Power11

The enhancements for IBM Power11 Python performance represents a holistic approach to performance optimization, delivering improvements across three interconnected layers: the hardware architecture, the compiler and toolchain ecosystem, and the runtime environment. Each layer builds upon the other to unlock unprecedented efficiency for modern workloads, particularly in AI, HPC, and data-intensive applications.

### ► Hardware Layer – Power11 Architecture

At the foundation, Power11 introduces a significant increase in core density—up to 25% more cores per chip compared to Power10—while retaining four Matrix Multiply Assist (MMA) engines per core. This is complemented by enhanced memory bandwidth, improved cache efficiency, and faster interconnects, ensuring that compute and data movement scale seamlessly for parallel and distributed workloads.

### ► Compiler and Toolchain Layer – MMA-Aware Optimization

The second layer focuses on enabling software to exploit hardware capabilities. Starting with GCC 10.2, the toolchain includes built-in MMA functions and flags such as `-mcpu=power11`, allowing numerical libraries like OpenBLAS to compile kernels optimized for matrix acceleration. These optimizations propagate upward to Python packages such as NumPy and SciPy, ensuring that high-level APIs can leverage low-level acceleration

### ► Runtime Layer – Modernized Python Execution

At the top layer, Python itself is evolving to meet the demands of high-performance computing. Recent releases (3.13 and 3.14) introduce free-threaded execution (PEP 779) and an experimental JIT compiler, reducing interpreter overhead and enabling true parallelism. When paired with Power11's high core count and MMA acceleration, these runtime enhancements deliver substantial gains for compute-bound workloads.

Together, these three layers form a synergistic performance stack, where hardware innovations, compiler optimizations, and runtime modernization converge to maximize throughput and efficiency for next-generation applications.

### 5.3.1 Power11 Hardware Enhancement

The IBM Power11 processor introduces a series of architectural enhancements that significantly advance compute density, memory performance, and AI acceleration capabilities:

- ▶ Up to 25% higher core density compared to Power10, while retaining four MMA engines per core, providing a proportional increase in MMA throughput.
- ▶ Improved memory bandwidth and cache hierarchy efficiency, optimizing performance for data-intensive and memory-bound workloads.
- ▶ Greater energy efficiency, enabling higher sustained throughput under continuous load.
- ▶ Faster CPU-to-CPU interconnects, improving scaling for distributed, parallel, and multi-socket environments.

Together, these architectural enhancements significantly strengthen the performance of MMA-accelerated AI and HPC software stacks.

### 5.3.2 Compiler Toolchain and Library-Level Enhancements

Performance-critical Python packages such as [NumPy](#), [SciPy](#), and [scikit-learn](#) rely on compiled numerical libraries like [OpenBLAS](#), which in turn depend on the GNU Compiler Collection (GCC) toolchain.

Starting with GCC 10.2, the compiler includes built-in functions for the Matrix Multiply Assist (MMA) facility, enabling numerical kernels to leverage hardware acceleration via flags such as `-mcpu=power11`.

For more information reference:

- ▶ <https://gcc.gnu.org/pipermail/gcc-patches/2024-March/648064.html>
- ▶ <https://gcc.gnu.org/onlinedocs/gcc-14.2.0/gcc/PowerPC-Matrix-Multiply-Assist-Built-in-Functions.html>

Compiling OpenBLAS with these flags on Power11 will:

- ▶ Detect and utilize MMA units per core for accelerated matrix operations.
- ▶ Allow downstream packages (NumPy, SciPy, etc.) to inherit these optimizations.

This propagation of MMA capabilities through the Python scientific ecosystem means high-level APIs can access low-level hardware acceleration without code changes.

**Recommendation:** Verify each package's configuration to ensure MMA-optimized libraries are in use. Alternatively, build packages from source and explicitly link to optimized libraries.

### 5.3.3 Python Interpreter and Runtime Enhancements

Recent Python releases (3.13 and 3.14) continue the modernization of the CPython runtime, introducing significant performance improvements:

- ▶ PEP 779: Free-Threaded Python

Python 3.14 officially supports free-threaded execution (introduced experimentally in 3.13). By disabling the Global Interpreter Lock (GIL), Python can fully utilize available CPU cores, enabling true parallelism for multi-threaded workloads.

Note: Not all applications benefit automatically. Many numerical libraries (e.g., OpenBLAS) already bypass the GIL using OpenMP for high-performance parallel execution. For more information see

<https://docs.python.org/3.14/whatsnew/3.14.html#whatsnew314-free-threaded-now-supported>

► Experimental Just-In-Time (JIT) Compiler

Python 3.13 introduced an experimental JIT compiler aimed at reducing interpreter overhead, potentially accelerating certain workloads. For more information see <https://docs.python.org/3.13/whatsnew/3.13.html#an-experimental-just-in-time-jit-compiler>

When combined with IBM Power11’s high core count and MMA acceleration, these enhancements significantly improve baseline Python performance, especially for compute-bound workloads.

### 5.3.4 Workload Validation

A series of workload validation exercises were conducted using example programs from the [pyeco Github repository](#). The objective here being to demonstrate functional performance scaling and ecosystem compatibility when running Python-based AI/ML workloads on Power11 systems.

Each workload leveraged optimized ppc64le packages from the devpi repository to ensure efficient use of the Power11 processor capabilities. The selected examples, while not intended as benchmarks, cover a range of computational patterns from dense linear algebra to data transformation and inference tasks.

Execution was measured using the UNIX *time* utility to capture end-to-end runtime including interpreter startup, dependency resolution and computation. Dependencies were resolved prior to execution to ensure consistent and reproduceable results across all systems in the environment.

While Power11 and Power10 share the same Power ISA v3.1 architecture, the Power11 configuration demonstrated improved throughput, with observed time reductions of up to 40% in several test cases. These gains primarily reflect micro-architectural enhancements, such as increased core efficiency, improved cache hierarchy and optimized instruction scheduling within the Power11 cores.

The following section outlines the test environment and summarizes the observed performance characteristics across each workload.

Table 5-1

Characteristic	Power11	Power10
Hardware	RHEL10 LPAR running on S1124 in 1 NUMA node with 16CPUs (SMT8), 32GB memory.	RHEL10 LPAR running on S1024 in 1 NUMA node with 16CPUs (SMT8), 32GB memory.
Kernel	6.12.0-55.9.1.el10_0.ppc64le	6.12.0-55.9.1.el10_0.ppc64le
Python	3.12.9	3.12.9
GCC	14.2.1	14.2.1

**Note:** These tests emphasize CPU-only (on-chip) execution with ppc64le-optimized libraries. While the workloads include non-python components (e.g .sh scripts), they serve to illustrate broader performance scaling trends within the Power ecosystem. They are not benchmarks.

Table 5-2

Example	Metric	Power11 (s)	Power10 (s)	%speedup
ONNX	real	5.300	6.720	+21%
ONNX	user	4.118	4.337	+5%
Scikit-learn	real	7.136	12.558	+43%
Scikit-learn	user	5.044	8.873	+43%

The *user time* metric was selected as the principal performance indicator of computational efficiency, as it most accurately reflects CPU utilization during processing of CPU-bound workloads. The *real time* metric is included for completeness to demonstrate the total elapsed wall-clock time observed under normal conditions, including interpreter startup and I/O overheads.

## 5.4 Support for AI/ML frameworks

Modern enterprise workloads increasingly rely on artificial intelligence (AI) and machine learning (ML) to drive innovation, automate decision-making, and extract insights from vast datasets. IBM Power systems, particularly those running Linux, are well-positioned to support these workloads thanks to their robust architecture, high memory bandwidth, and hardware acceleration capabilities. This section explores the integration and optimization of popular AI/ML frameworks – PyTorch, TensorFlow, and scikit-learn – on IBM Power platforms, highlighting performance, compatibility, and deployment strategies.

### 5.4.1 Overview of Frameworks

This section focuses on three popular AI/ML frameworks:

- ▶ PyTorch is a dynamic, Python-based deep learning framework developed by Meta AI. It is widely used for research and production due to its flexibility and strong GPU acceleration support.
- ▶ TensorFlow, developed by Google, is a comprehensive ML framework that supports both deep learning and traditional ML models. It offers static graph execution and is optimized for large-scale deployments.
- ▶ scikit-learn is a Python library for classical machine learning algorithms, including regression, classification, clustering, and dimensionality reduction. It is built on top of NumPy, SciPy, and matplotlib.

While each framework offers distinct capabilities, they share a common need for efficient computation, parallel processing, and hardware acceleration. IBM Power systems address these requirements with advanced features such as Power10 and Power11 processors, integrated AI Accelerator Units (AIU), and high-bandwidth memory subsystems.

#### Compatibility and Installation on Linux on Power

IBM has worked closely with the open-source community to ensure that AI/ML frameworks are compatible with Linux distributions on Power architecture, including RHEL, Ubuntu, and SLES. Most packages are available via pip, conda, or native package managers, and containerized deployments using Docker or Podman are fully supported.

For example:

- ▶ pip install torch torchvision torchaudio
- ▶ pip install tensorflow
- ▶ pip install scikit-learn

Alternatively, IBM offers pre-built, Power-optimized container images through the IBM Container Registry and OpenCE (Open Cognitive Environment). These containers integrate leading AI frameworks alongside performance-tuned libraries such as oneDNN, OpenBLAS, and Eigen, ensuring efficient execution on IBM Power architecture.

Library optimizations refer to tuning mathematical and computational libraries so they can fully exploit the underlying hardware architecture. This involves:

- ▶ **Instruction-Level Optimization**

Libraries like oneDNN (for deep learning) and OpenBLAS/Eigen (for linear algebra) are compiled to use vectorized instructions (e.g., VSX on Power) and SIMD units for parallel execution. This reduces the number of CPU cycles per operation.

- ▶ **Threading and Parallelism**

Optimized libraries leverage multi-threading and SMT (Simultaneous Multithreading) capabilities of Power processors. Workloads are distributed across cores and threads efficiently to maximize throughput.

- ▶ **Memory Hierarchy Awareness**

Libraries are tuned to minimize cache misses and optimize data locality. They use techniques like blocking and prefetching to keep data close to the compute units.

- ▶ **Hardware Acceleration Integration**

For AI workloads, libraries like oneDNN can offload certain operations to AI Accelerator Units (AIUs) on Power10/Power11. This accelerates tensor and matrix computations beyond what general-purpose cores can achieve.

- ▶ **Precision and Data Types**

Libraries often support mixed-precision computing (FP32, FP16) to balance accuracy and performance. Optimizations ensure these conversions are efficient on Power hardware.

## Hardware Acceleration and Optimization

IBM Power systems offer several hardware features that accelerate AI/ML workloads:

- ▶ **AIU (AI Accelerator Unit):** Introduced in Power10 and enhanced in Power11, AIU provides matrix math acceleration for deep learning operations. PyTorch and TensorFlow can leverage AIU via optimized backends such as ONNX Runtime, XLA, or custom kernels.
- ▶ **SIMD and VSX Extensions:** These vector instructions accelerate linear algebra operations used in scikit-learn and other numerical libraries.

IBM also contributes patches upstream to ensure that Power-specific optimizations are included in major releases of PyTorch and TensorFlow. These include support for little-endian Power, AltiVec/VSX, and multi-threaded BLAS libraries.

## Performance Benchmarks

Benchmarking AI/ML workloads on Power systems has shown competitive performance compared to x86 platforms, especially for memory-intensive and parallelizable tasks. For example:

- ▶ PyTorch ResNet-50 training on Power10 with AIU shows up to 1.8x speedup compared to baseline CPU execution.

- ▶ TensorFlow BERT inference benefits from optimized matrix multiplication routines and achieves low latency on Power11 systems.
- ▶ scikit-learn random forest and SVM models scale efficiently across multiple cores, leveraging Power’s SMT (Simultaneous Multithreading) capabilities.

## 5.5 Compatibility with OpenBLAS, NumPy, and other scientific libraries

The relationship between Python and matrix multiplication optimizations on the IBM Power11 platform is most prominently realized through OpenBLAS (open-source implementation of the basic linear algebra and subprograms (BLAS)) and NumPy. These two libraries form the foundation of high-performance numerical computation in Python environments. Their relevance to this Redbook lies in how they are built, tuned and executed to take advantage of MMA capabilities available on the Power11 architecture.

The Power ISA v3.1 forms the architectural basis for both the Power10 and Power11 processors. This instruction set introduces hardware extensions that accelerate matrix operations across multiple data types, including double precision, single precision and half precision computations. The MMA unit enables these operations by providing eight 512-bit accumulator registers that can efficiently perform matrix multiply-accumulate instructions. These accumulators significantly improve throughput for dense numerical workloads, such as those found in machine learning and scientific computing.

Compiler support for these instructions is provided through the GNU Compiler Collection (GCC). As described in 5.3.2, “Compiler Toolchain and Library-Level Enhancements” on page 79, the toolchain includes built-in functions that expose MMA capabilities to developers. These functions can be enabled through compilation flags such as `-mcpu=power11` or `-mcpu=native`, allowing optimized binaries to be generated automatically for the target processor.

To validate and demonstrate compatibility:

1. OpenBLAS was compiled from source, ensuring full use of the MMA instruction set defined by ISA v3.1.
2. NumPy was subsequently built from source and explicitly linked against this OpenBLAS library.
3. Configuration was verified through NumPy to confirm correct library linkage and kernel recognition. This is shown in Figure 5-2 on page 84
4. Following this build process, several NumPy test scripts were executed to validate end-to-end functionality and performance.

These tests confirm that both OpenBLAS and NumPy correctly utilize MMA-enabled kernels when running on Power10 and Power11.

Although OpenBLAS identifies kernels as Power10 in its configuration, this is expected behavior because Power11 and Power10 share the same Power ISA v3.1 instruction set. Consequently, kernel naming reflects the ISA level rather than the processor generation.

### 5.5.1 Inspecting the OpenBLAS Configuration:

To verify OpenBLAS is using MMA (Power10) kernels for its core performance routines, you can inspect the static or shared libraries for references to those optimized functions.

## Querying the Library:

We can query the shared object to confirm references pointing to gemm (general matrix-matrix multiply – Level 3 BLAS routines) implementation files. For example in Figure 5-2, we can see strings (and the functions they correspond to) such as;

- sgemm = single precision
- dgemm = double precision
- cgemm & zgemm = complex operations

The associated \*.c files implement the MMA-based microkernels for Power10.

Use the following command to display this output:

```
strings /opt/openblas/lib/libopenblasopenmp.so | grep -E "power10" | sort
```

```
(powerai) [cecuser@pl226-pvml ~]$ strings /opt/openblas/lib/libopenblasopenmp.so | grep -E "power10" | sort
caxpy_power10.c
ccopy_power10.c
cgemm_kernel_power10.c
daxpy_power10.c
dcopy_power10.c
ddot_power10.c
dgemm_kernel_power10.c
dgemm_ncopy_8_power10.c
dgemm_small_kernel_nn_power10.c
dgemm_small_kernel_nt_power10.c
dgemm_small_kernel_tn_power10.c
dgemm_small_kernel_tt_power10.c
dgemv_n_power10.c
dgemv_t_power10.c
gemm_small_kernel_permit_power10.c
saxpy_power10.c
sbgemm_kernel_power10.c
sbgemm_ncopy_16_power10.c
sbgemm_ncopy_8_power10.c
sbgemm_tcopy_16_power10.c
sbgemm_tcopy_8_power10.c
sbgemv_n_power10.c
sbgemv_t_power10.c
scopy_power10.c
sdot_power10.c
sgemm_kernel_power10.c
sgemm_small_kernel_nn_power10.c
sgemm_small_kernel_nt_power10.c
sgemm_small_kernel_tn_power10.c
sgemm_small_kernel_tt_power10.c
trsm_kernel_LN_power10.c
trsm_kernel_LT_power10.c
trsm_kernel_RN_power10.c
trsm_kernel_RT_power10.c
zaxpy_power10.c
zcopy_power10.c
zgemm_kernel_power10.c
zgemv_n_power10.c
```

Figure 5-2 query the library for Power10 gemm kernel

## 5.5.2 Inspecting the NumPy Configuration

Inspecting NumPy’s configuration is essential to ensure that your installation is taking full advantage of optimized libraries and hardware features. Although NumPy is a Python library, its performance depends heavily on compiled C and Fortran code linked to high-performance backends such as BLAS and LAPACK. Different builds may use different implementations—such as OpenBLAS, Intel MKL, or IBM ESSL—which can significantly impact execution speed. By examining the configuration, you can verify which libraries are active, confirm

SIMD instruction support (such as AVX on x86 or VSX on IBM Power), and check whether multi-threading and compiler optimizations were applied during the build process.

NumPy provides a convenient method for this inspection through the `numpy.show_config()` function. Running this command displays critical details, including the linked BLAS/LAPACK libraries, enabled SIMD features, compiler flags, and threading layers. For example, the output may show `openblas_info` or `essl_info` under the BLAS section, indicating the backend in use. It also reveals whether advanced instruction sets and optimization flags like `-O3` or `-march=native` were included during compilation. On IBM Power systems, it is particularly important to confirm that ESSL or OpenBLAS is linked, VSX SIMD support is enabled, and compiler options are tuned for Power architecture. These checks help ensure that NumPy delivers optimal performance for AI, HPC, and data analytics workloads.

### Example source build

OpenBLAS was built specifically without dynamic architecture support and `-mcpu` flags (to force it to use only the native detected architecture during the configure phase) on each system. This is shown in Example 5-1

#### *Example 5-1 Building OpenBLAS*

---

```
export USE_THREAD=1
export NUM_THREADS=$(nproc)
export DYNAMIC_ARCH=0
```

---

Another notable configuration was enabling multithreading mode with `openmp`, allowing OpenBLAS to parallelize across multiple cores.

#### *Example 5-2*

---

```
export OPENBLAS_NUM_THREADS=$(nproc)
export OMP_NUM_THREADS=$(nproc)
make -j DYNAMIC_ARCH=0 CC=gcc FC=gfortran HOSTCC=gcc BINARY=64 INTERFACE=64 \
USE_OPENMP=1 LIBNAMESUFFIX=openmp
```

---

NumPy was built using the newly compiled OpenBLAS libraries by targeting the shared objects via `$LD_LIBRARY_PATH` (and `$PKG_CONFIG_PATH`).

Step-by-step instructions to compile OpenBLAS on Linux can be found at <https://gist.github.com/bgeneto/d01cf916f542e22b36e22aff6bcd94b>

Building NumPy from source with `venv` or system Python on Linux is documented at <https://numpy.org/devdocs/building/>

The following Python sequence was used to query the OpenBLAS and NumPy configuration:

```
python3 -c "import numpy, numpy.__config__ as c; c.show()"
```

Figure 5-3 is a screen shot of the optimized configuration on Power11:

```
(powerai) [cecuser@p1226-pvml ~]$ python3 -c "import numpy, numpy.__config__ as c; c.show()"
Build Dependencies:
blas:
  detection method: pkgconfig
  found: true
  include directory: /opt/openblas/include
  lib directory: /opt/openblas/lib
  name: openblas
  openblas configuration: USE_64BITINT= DYNAMIC_ARCH= DYNAMIC_OLDER= NO_CBLAS= NO_LAPACK=
    NO_LAPACKE= NO_AFFINITY=1 USE_OPENMP=1 POWER10 MAX_THREADS=16
  pc file directory: /opt/openblas/lib/pkgconfig
  version: 0.3.30.dev
lapack:
  detection method: pkgconfig
  found: true
  include directory: /opt/openblas/include
  lib directory: /opt/openblas/lib
  name: openblas
  openblas configuration: USE_64BITINT= DYNAMIC_ARCH= DYNAMIC_OLDER= NO_CBLAS= NO_LAPACK=
    NO_LAPACKE= NO_AFFINITY=1 USE_OPENMP=1 POWER10 MAX_THREADS=16
  pc file directory: /opt/openblas/lib/pkgconfig
  version: 0.3.30.dev
Compilers:
c:
  args: -O3, -ftree-vectorize, -fprefetch-loop-arrays, --param, prefetch-latency=300
  commands: cc
  linker: ld.bfd
  linker args: -O3, -ftree-vectorize, -fprefetch-loop-arrays, --param, prefetch-latency=300
  name: gcc
  version: 14.2.1
c++:
  commands: c++
  linker: ld.bfd
  name: gcc
  version: 14.2.1
cython:
  commands: cython
  linker: cython
  name: cython
  version: 3.1.5
Machine Information:
build:
  cpu: ppc64le
  endian: little
  family: ppc64
  system: linux
host:
  cpu: ppc64le
  endian: little
  family: ppc64
  system: linux
Python Information:
path: /home/cecuser/venvs/powerai/bin/python
version: '3.12'
SIMD Extensions:
baseline:
- VSX
- VSX2
found:
- VSX3
- VSX4
```

Figure 5-3 Power11 results for query Numpy

Looking at the output shown in Figure 5-3, you can validate that the build is correct by finding the values shown in Example 5-3.

*Example 5-3 Validate directories*

---

```
include directory: /opt/openblas/include
lib directory: /opt/openblas/lib
```

---

Our build dependencies – blas and lapack (linear algebra package) – are pointing directly to our custom build in /opt/openblas as shown in the items shown in Example 5-4.

*Example 5-4 openblas configuration*

---

```
name: openblas
openblas configuration: USE_64BITINT= DYNAMIC_ARCH=1 DYNAMIC_OLDER= NO_CBLAS= NO_LAPACK=
NO_LAPACKE= NO_AFFINITY=1 USE_OPENMP=1 POWER10 MAX_THREADS=16
```

---

The detected processor/core type at build time is POWER10, which is expected as there are no specific new kernels for Power11 as it shares the same ISA v3.1 architecture as the Power10.

Another confirmation is the output of the make command when building with dynamic architecture support on the Power11 which is shown in Example 5-5.

*Example 5-5 Build complete message*

---

```
OpenBLAS build complete. (BLAS CBLAS LAPACK LAPACKE)
OS          ... Linux
Architecture ... power
BINARY      ... 64bit
C compiler   ... GCC (cmd & version: gcc (GCC) 14.2.1 20250110 (Red Hat 14.2.1-7))
Fortran compiler ... GFORTRAN (cmd & version: GNU Fortran (GCC) 14.2.1 20250110 (Red Hat 14.2.1-7))
Library Name ... libopenblasopenmpp-r0.3.30.dev.a (Multi-threading; Max num-threads is 16)
Supporting multiple power cpu models with minimum requirement for the common code being POWER10
```

---

**Important:** The toolchain highlights the minimum requirement for the common code is Power10.

### Example pip install

Efficient package management is critical for achieving optimal performance on IBM Power systems, especially for AI, HPC, and data analytics workloads. While pip install is the standard method for installing Python packages, additional considerations apply when working on Power architecture to ensure compatibility and performance.

In our testing we used the command shown in Example 5-6.

*Example 5-6 Sample install command*

---

```
pip install --prefer-binary numpy openblas
--extra-index-url=https://wheels.developerfirst.ibm.com/ppc64le/linux
```

---

We used this as it is the more convenient method of installing python packages from the devpi project. After installing with this method, the system returned the output shown in Figure 5-4 on page 88 when queried:

```

(powerai) [cecuser@p1226-pvm1 ~]$ python3 -c "import numpy, numpy.__config__ as c; c.show()"
/home/cecuser/venvs/powerai/lib64/python3.12/site-packages/numpy/__config__.py:155: UserWarning: Install `pyyaml`
for better output
  warnings.warn("Install `pyyaml` for better output", stacklevel=1)
{
  "Compilers": {
    "c": {
      "name": "gcc",
      "linker": "ld.bfd",
      "version": "13.3.1",
      "commands": "/opt/rh/gcc-toolset-13/root/usr/bin/gcc"
    },
    "cython": {
      "name": "cython",
      "linker": "cython",
      "version": "3.1.2",
      "commands": "cython"
    },
    "c++": {
      "name": "gcc",
      "linker": "ld.bfd",
      "version": "13.3.1",
      "commands": "/opt/rh/gcc-toolset-13/root/usr/bin/g++"
    }
  },
  "Machine Information": {
    "host": {
      "cpu": "ppc64le",
      "family": "ppc64",
      "endian": "little",
      "system": "linux"
    },
    "build": {
      "cpu": "ppc64le",
      "family": "ppc64",
      "endian": "little",
      "system": "linux"
    }
  },
  "Build Dependencies": {
    "blas": {
      "name": "openblas",
      "found": true,
      "version": "0.3.29",
      "detection method": "pkgconfig",
      "include directory": "/home/tester/OpenBLAS/local/openblas/include",
      "lib directory": "/home/tester/OpenBLAS/local/openblas/lib",
      "openblas configuration": "USE_64BITINT=0 DYNAMIC_ARCH=1 DYNAMIC_OLDER= NO_CBLAS= NO_LAPACK=0 NO_LAPACKE= NO
_AFFINITY=1 USE_OPENMP=1 POWER9 MAX_THREADS=8",
      "pc file directory": "/home/tester/OpenBLAS/local/openblas/lib/pkgconfig"
    },
    "lapack": {
      "name": "openblas",
      "found": true,
      "version": "0.3.29",
      "detection method": "pkgconfig",

```

Figure 5-4 Devpi NumPy config query on RHEL10 Power11 LPAR

There are 2 points to note here.

1. First, older compiler versions (for example GCC 13.3.1) still include MMA built-in functions, even though they predate the latest Power toolchain.
2. Second, and more importantly, the OpenBlas configuration

```

"openblas configuration": "USE_64BITINT=0 DYNAMIC_ARCH=1 DYNAMIC_OLDER=
NO_CBLAS= NO_LAPACK=0 NO_LAPACKE= NO_AFFINITY=1 USE_OPENMP=1 POWER9
MAX_THREADS=8"

```

specifies IBM Power9® as the architecture at build time for both blas and lapack. Therefore, we cannot guarantee the MMA kernels were compiled.

At the time of writing, it is recommended to build both programs from source to ensure full Power10 compatibility and granular control of the configuration.

## 5.6 Developer productivity and tooling improvements

With Power11, the emphasis for Python developers is on ecosystem maturity and streamlined workflows. The platform continues to provide an enhanced foundation for Python workloads, leveraging the robust Power architecture and ppc64le-optimized libraries. At the same time, also delivering a smoother developer experience through improved package availability, tooling and integration across the wider Power ecosystem.

The ppc64le ecosystem has reached a point of consistency and maturity, meaning developers can seamlessly install, build and run most popular Python packages on Power. This includes packages built with ppc64le optimizations, often compiled against tuned math libraries like OpenBLAS that exploit Power's VSX and MMA capabilities.

- ▶ System Python

On RHEL on Power, using the system Python is simple, efficient and well-integrated with the platform's package manager and tuned libraries. With less environment setup tasks, it is fast to get started developing, improving productivity.

- ▶ Devpi and internal package mirrors:

With Devpi, organizations can continue to maintain internal mirror registries of PyPi built for ppc64le, ensuring consistent builds and reproducibility. This supports enterprise-grade Python CI/CD pipelines in Power systems environments as well as accelerated package installation and dependency resolution times when combined with Power11's improved I/O and memory latency.

- ▶ Enhanced tooling and integration:

IBM's continued contribution to the Python ecosystem and broad support for build tools like pip and wheel reinforces the platform's readiness as well as simplifies cross-compiling to ppc64le. Native support for containerized Python development (via podman or docker) with pre-built python images means developers can instantly deploy on Power11 without manual configuration.

While Python itself is platform-agnostic, IBM's commitment ensures that the Power platform remains a valued member and contributor in opensource and enterprise Python stacks. The focus is not raw platform performance, but ecosystem expansion and continuity, maintaining consistency, providing optimized packages, guaranteeing Power developers are productive and supported.

For more information see [Developing applications using Python Packages on IBM Power](#),





## AI and Security

Security is a critical aspect of AI workloads because these systems often process sensitive data and operate in environments where trust and compliance are paramount. AI models rely on large volumes of data—such as financial transactions, healthcare records, or operational telemetry—which must be protected against unauthorized access, breaches, and tampering. Strong security measures ensure data confidentiality, integrity, and availability throughout the AI lifecycle, from training to inference.

Key considerations include data privacy and regulatory compliance, especially when handling personally identifiable information (PII) or industry-specific regulated data. Secure infrastructure, such as encrypted storage, secure boot, and isolated execution environments, helps prevent attacks on AI models and pipelines. Additionally, protecting against adversarial threats—where malicious inputs can manipulate model behavior—is essential for maintaining reliability. Enterprises also need robust identity and access management, network security, and continuous monitoring to safeguard AI workloads deployed across hybrid and edge environments. Ultimately, security is not just a technical requirement but a foundation for trust, enabling organizations to adopt AI confidently while meeting legal and ethical obligations.

The following topics are covered in this chapter:

- ▶ Secure model deployment
- ▶ Data privacy and encryption in AI pipelines
- ▶ Security and AI on IBM Power
- ▶ Securing AI on IBM Power
- ▶ Controlling data and models in your AI ecosystem

## 6.1 Secure model deployment

Secure model deployment is essential for ensuring that AI systems operate safely, reliably, and in compliance with organizational and regulatory requirements. The threat landscape for deployed models includes risks such as model extraction, where attackers attempt to replicate models through repeated API queries, and membership inference attacks, which aim to determine whether specific data points were part of the training set. Adversarial examples, crafted to mislead models, and poisoning attacks, where malicious data is introduced during training, also pose significant challenges. Additionally, supply chain vulnerabilities in pre-trained models or libraries can compromise security, making dependency scanning and maintaining a Software Bill of Materials (SBOM) critical.

To mitigate these risks, organizations should adopt robust deployment practices. Environment hardening is a foundational step, involving the use of minimal container images, kernel security features like Seccomp and AppArmor, and runtime anomaly detection tools such as Falco. Authentication and authorization must be enforced through strong mechanisms like mutual TLS, short-lived tokens, dynamic secrets, and role-based access control, all aligned with the principle of least privilege. Encryption is equally vital, with AES-256 for model artifacts, TLS 1.3 for inference traffic, and automated key rotation via secure key management systems. Model integrity should be guaranteed through cryptographic signing and verification during both CI/CD and runtime, while immutable deployment artifacts should reside in trusted registries.

Monitoring and logging play a crucial role in detecting anomalies and ensuring accountability. Real-time anomaly detection for API calls, SIEM integration for centralized monitoring, and privacy-preserving logging practices help maintain security without exposing sensitive data. Compliance and governance frameworks such as NIST AI Risk Management and ISO/IEC 27001 should guide operations, supported by detailed model cards, datasheets, and audit trails for training, deployment, and access events.

Advanced protections like differential privacy, federated learning, homomorphic encryption, and secure enclaves (e.g., Intel SGX or IBM Power Secure Execution) further strengthen security by reducing data exposure and enabling confidential computation. Deployment patterns vary by context: on-premises environments offer maximum control for regulated industries, while cloud deployments require private endpoints, customer-managed encryption keys, and confidential computing features. Hybrid approaches combine sensitive preprocessing on-premises with scalable inference in the cloud, and edge deployments demand lightweight models with encrypted updates and secure boot mechanisms.

Finally, securing the ML lifecycle involves integrating security checks into CI/CD pipelines, including model validation, dependency scanning, and vulnerability assessments. Signed containers, trusted registries, and policy enforcement tools like Open Policy Agent (OPA) ensure integrity throughout the process. Emerging trends such as confidential AI, zero-trust principles applied to ML pipelines, and AI Bills of Materials (AI-BOM) for tracking model lineage highlight the evolving nature of secure AI deployment.

### On-premises compared to cloud deployment

Models can be deployed in either on-premises environments or in the cloud. These two options bring different security considerations. This section presents a comparison of the security considerations across these options across several dimensions:

- Control and visibility

When considering control and visibility, on-premises deployments offer full control over hardware, network, and physical access, making it easier to enforce custom security

policies and compliance requirements while providing complete visibility into all layers of the stack. In contrast, cloud deployments operate under a shared responsibility model where the provider secures the infrastructure and the customer secures workloads. This approach limits visibility into underlying hardware and hypervisors and requires reliance on the provider’s compliance certifications.

► **Data security**

For data security, on-premises environments keep sensitive data within a controlled infrastructure and allow for air-gapped systems for high-security workloads. Cloud environments typically provide encryption at rest and in transit as a standard feature, but encryption keys are often managed by the provider unless customer-managed keys (CMK) are used. Misconfigured storage buckets or IAM policies can increase the risk of data exposure in cloud deployments.

► **Network security**

In terms of network security, on-premises setups allow organizations to manage internal network segmentation and firewalls directly, making zero-trust enforcement easier. Cloud environments offer strong perimeter security but require careful configuration of virtual private clouds (VPCs), security groups, and private endpoints to prevent accidental public exposure.

► **Identity and access management**

For identity and access management (IAM), on-premises deployments typically integrate with enterprise systems like Active Directory or LDAP, giving full control over role-based access control (RBAC) and multi-factor authentication (MFA) policies. Cloud platforms provide advanced IAM capabilities, including fine-grained roles and temporary credentials, but misconfigured policies can lead to privilege escalation risks.

► **Compliance and governance**

Regarding compliance and governance, on-premises solutions are often better suited for highly regulated industries such as finance and healthcare, as they allow full audit control and easier adherence to strict compliance requirements. Cloud providers offer certifications like ISO, SOC, and HIPAA, but ultimate responsibility for data handling remains with the customer, and cross-border data residency can pose additional challenges.

► **Threat surface**

Finally, the threat surface differs significantly. On-premises deployments require organizations to manage physical security and address insider threats, which can be harder to detect. Cloud environments have a larger attack surface due to multi-tenancy, though providers invest heavily in security. However, misconfiguration by customers remains a common cause of breaches.

In summary, on-premises deployments provide maximum control and are ideal for sensitive workloads but come with higher operational overhead. Cloud deployments offer scalability, advanced IAM features, and built-in security capabilities, but they require strong configuration practices and trust in the provider.

Table 6-1 summarizes these considerations.

*Table 6-1 Summarization of security considerations on-premises and in the cloud*

<b>Security Dimension</b>	<b>On-Premises</b>	<b>Cloud</b>
Control & Visibility	Full control over hardware, network, and physical access; complete visibility.	Shared responsibility model; limited visibility into hardware; relies on provider compliance.

Security Dimension	On-Premises	Cloud
Data Security	Sensitive data stays in controlled environment; air-gapped systems possible.	Encryption at rest and in transit standard; keys often provider-managed unless CMK used; risk of misconfiguration.
Network Security	Internal segmentation and firewalls under your control; easier zero-trust.	Strong perimeter security; requires careful VPC and endpoint configuration; risk of exposure if misconfigured.
<b>Identity &amp; Access Mgmtl</b>	Integrated with enterprise AD/LDAP; full RBAC and MFA control.	Advanced IAM features (fine-grained roles, temporary credentials); risk of privilege escalation if misconfigured.
Compliance & Governance	Easier for regulated industries; full audit control.	Provider offers certifications (ISO, SOC, HIPAA); customer responsible for data handling; cross-border concerns.
Threat Surface	Physical security is your responsibility; insider threats harder to detect.	Larger attack surface due to multi-tenancy; provider invests heavily in security; misconfiguration risk remains.
Summary	Maximum control; ideal for sensitive workloads; higher operational overhead.	Scalability and advanced features; requires strong configuration and trust in provider.

## 6.2 Data privacy and encryption in AI pipelines

As AI becomes increasingly embedded in modern digital ecosystems, the importance of securing AI pipelines has never been greater. From healthcare and finance to autonomous systems and smart cities, AI models are trained on vast amounts of sensitive data. Ensuring data privacy and encryption throughout the AI lifecycle is essential to protect individuals, organizations, and the integrity of the models themselves.

An AI pipeline refers to the end-to-end workflow involved in building, training, deploying, and monitoring machine learning (ML) models. This includes:

- ▶ Data Collection and Preprocessing
- ▶ Model Training and Validation
- ▶ Deployment and Inference
- ▶ Monitoring and Maintenance

Each stage presents unique security challenges, especially when handling sensitive or personally identifiable information (PII).

AI systems often rely on large datasets that may include personal, financial, or health-related information. Without proper safeguards, these systems can:

- ▶ Leak sensitive data through model outputs or logs
- ▶ Be vulnerable to adversarial attacks

- ▶ Violate data protection regulations like GDPR, HIPAA, or India's DPDP Act

## 6.2.1 Data Privacy Threats in ML Systems

Major security threats to an ML System can be categorized as:

- ▶ **Data Poisoning:** Attackers add fake or harmful data during training so that the model learns the wrong thing. For example, Microsoft launched an AI chatbot named *Tay* on Twitter (2016), designed to learn from interactions with users. Malicious users began feeding it offensive and harmful content. *Tay* quickly started generating racist and inappropriate tweets, forcing Microsoft to shut it down [1].
- ▶ **Adversarial Attacks:** Small, smart changes to input (like editing pixels in an image) to trick the model into making mistakes. For example, In Stop Sign Misclassification Model in Autonomous Vehicles: Researchers demonstrated that by placing small stickers on a stop sign, they could trick a computer vision model used in autonomous vehicles into misclassifying it as a speed limit sign [2].
- ▶ **Model or Intellectual Property Theft:** Stealing a trained model or reverse engineering it. For example, Healthcare AI models trained on patient data require strict data privacy controls, including encryption and anonymization, to comply with regulations like HIPAA and ensure patient confidentiality.
- ▶ **Infrastructure Attacks:** Hacking servers, networks, or containers where ML pipelines run. For example, Capital One Data Breach (2019): A misconfigured firewall allowed a hacker to access sensitive data stored in AWS S3 buckets, affecting over 100 million customers. Infrastructure vulnerabilities especially in cloud environments where AI pipelines often run can expose sensitive data and models to attackers [3].
- ▶ **Data Leaks:** Sensitive training or prediction data gets exposed or stolen. For example, Strava Heatmap Incident (2018): Strava, a fitness tracking app, published a global heatmap showing user activity. Analysts discovered that the map inadvertently revealed the locations of military bases and patrol routes, as soldiers using the app had unknowingly shared their data. This is a case of data leakage, where seemingly harmless outputs exposed sensitive information [4].

## 6.2.2 Key Measures for Protecting ML Systems

Securing machine learning systems requires a multi-layered approach that addresses data, models, and infrastructure. Below are the essential measures organizations should implement to safeguard their AI pipelines:

### 1. Secure Data Management

Protecting the data used in ML pipelines is foundational to ensuring privacy and security.

- **Encryption at Rest and in Transit**

Use strong encryption standards (e.g., AES-256) to protect data stored on disk and TLS/SSL protocols for data transmitted over networks.

- **Access Control and Least Privilege**

Implement role-based access controls (RBAC) and ensure users only have access to the data they need to perform their tasks.

- **Data Integrity Verification**

Use checksums, hash functions, and digital signatures to verify that data has not been tampered with.

- **Anonymization and Pseudonymization**  
Remove or mask personally identifiable information (PII) to comply with privacy regulations like GDPR, HIPAA, India’s DPDP Act.
  - **Data Lineage and Auditing**  
Track the origin, transformation, and usage of data throughout the pipeline to ensure transparency and accountability.
2. Protecting Machine Learning Models
- Models themselves are valuable intellectual property and must be protected from theft and manipulation.
- **Model Watermarking**  
Embed unique identifiers into models to prove ownership and detect unauthorized use.
  - **Version Control and Change Tracking**  
Maintain a history of model versions and changes to ensure traceability and rollback capability.
  - **Adversarial Robustness Testing**  
Regularly test models against adversarial inputs to identify vulnerabilities and improve resilience.
  - **Secure Model Access**  
Restrict access to models via authenticated APIs, firewalls, and private endpoints to prevent unauthorized usage.
  - **Model Encryption and Obfuscation**  
Encrypt model files and use techniques like obfuscation to make reverse engineering more difficult.
3. Securing Infrastructure
- The infrastructure hosting ML workloads must be hardened against cyber threats.
- **Network Segmentation and Isolation**  
Isolate ML workloads from other systems using virtual private clouds (VPCs), containers, and secure zones.
  - **Trusted Execution Environments (TEEs)**  
Use hardware-based secure environments to run sensitive computations and protect data during processing.
  - **Patch Management and Vulnerability Scanning**  
Regularly update software and scan for vulnerabilities to prevent exploitation of known flaws.
  - **Monitoring and Incident Response**  
Implement continuous monitoring, anomaly detection, and automated incident response to quickly identify and mitigate threats.
  - **Security Training and Awareness**  
Educate MLOps teams on secure coding practices, threat modeling, and data privacy principles to foster a security-first culture.

## 6.3 Security and AI on IBM Power

IBM Power provides a strong foundation for secure AI deployments, combining high performance with built-in security features. Capabilities such as secure boot, workload isolation, and hardware-based encryption help protect data and models throughout the AI

lifecycle. These features make IBM Power an ideal platform for organizations looking to run AI workloads securely and reliably, whether on-premises or in hybrid environments.

### 6.3.1 Why Security is important

Cybersecurity threats are escalating globally, and recent reports highlight the severity of the situation. In the UK, the National Cyber Security Centre (NCSC) reported a sharp increase in nationally significant cyber incidents<sup>1</sup>. As of October 14, 2025, there have been 204 nationally significant attacks, compared to just 89 incidents in 2024. This surge underscores not only the growing frequency but also the increasing sophistication of cyberattacks. With several months remaining in 2025, this number is expected to rise further, signaling an urgent need for enhanced security measures. These figures are not isolated; they reflect a global trend of heightened cyber risk across all regions and industries.

The latest [IBM X-Force Threat Intelligence Report](#) reinforces this global perspective. It reveals that the Middle East and Africa (MEA) experienced the highest volume of incidents targeting the Finance and Insurance sectors, while Europe saw the greatest impact on Professional, Business, and Consumer Services. These findings confirm that cybersecurity is a universal challenge, affecting every geography and industry.

Alarmingly, the same X-Force report states that less than one-quarter of generative AI projects are being secured, despite the fact that security is fundamental to ensuring AI's safe and scalable future. Generative AI introduces new attack surfaces and vulnerabilities, making it imperative for organizations to secure the AI pipeline from the outset. Failure to do so exposes enterprises to risks that could compromise sensitive data, intellectual property, and operational integrity.

Unpatched vulnerabilities remain a critical concern, particularly for organizations operating in sectors deemed part of national infrastructure. According to X-Force, threat actors exploit vulnerabilities in more than one in four attacks against these sectors, highlighting the importance of proactive patch management and vulnerability remediation.

The IBM Institute for Business Value adds further urgency in its report, [The CEO's Guide to Generative AI](#), stating:

“Generative AI has given rise to a new generation of cyber threats. Hackers have more opportunities to exploit vulnerabilities—and more ways to execute their malicious campaigns.”<sup>2</sup>

This insight emphasizes that generative AI is not only a transformative technology but also a catalyst for evolving cyber threats. Organizations must adopt a security-first approach to AI deployment, integrating robust controls across data, models, infrastructure, and governance to mitigate these emerging risks.”

### 6.3.2 The IBM Framework for Securing Generative AI

Generative artificial intelligence (AI) is rapidly emerging as a leading area of technology investment, yet many organizations remain ill-prepared to address the cybersecurity risks it introduces. As with any transformative technology, it is critical to understand and mitigate the unique security challenges posed by generative AI, because adversaries will inevitably seek to exploit vulnerabilities to achieve their objectives. According to the [IBM Institute for Business Value](#), an overwhelming 96% of executives believe that adopting generative AI increases the likelihood of a security breach within their organization over the next three years<sup>3</sup>.

<sup>1</sup> <https://www.bbc.com/news/articles/ced61xv9671o>

<sup>2</sup> <https://www.ibm.com/thought-leadership/institute-business-value/en-us/report/ceo-generative-ai/cybersecurity>

AI models often ingest vast amounts of sensitive and valuable data during training, while business leaders explore how these models can optimize essential operations and outputs. This combination significantly raises the stakes. Organizations cannot afford to deploy unsecured AI systems into their environments, as doing so could expose them to severe operational, financial, and reputational risks.

The framework for secure AI deployment is built on three core pillars: securing the data, securing the model, and securing its usage. This is illustrated in Figure 6-1. Supporting these pillars is the critical need to secure the underlying infrastructure and establish robust AI governance practices. IBM Power provides an exceptional foundation for implementing this framework, offering enterprise-grade security, performance, and reliability to safeguard AI workloads across the entire lifecycle.

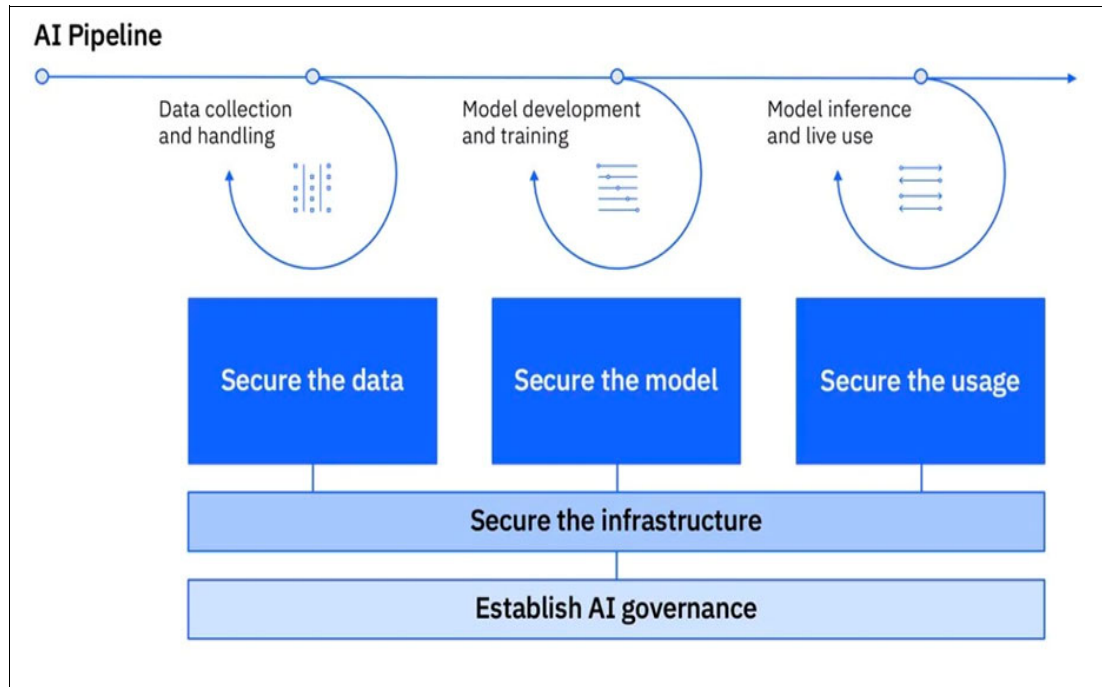


Figure 6-1 Framework for securing generative AI<sup>4</sup>

## Securing AI Across the Lifecycle

Building and deploying AI models introduces unique security challenges that organizations must address to protect sensitive data, maintain trust, and ensure compliance. A comprehensive security framework spans five critical areas: securing the data, securing the model, securing the usage, securing the infrastructure, and establishing governance. Each layer plays a vital role in reducing risk and safeguarding AI systems.

### Secure the Data

During data collection and handling, organizations aggregate massive volumes of information to train AI models. This process often involves granting access to multiple stakeholders—data scientists, engineers, developers—many of whom lack deep security expertise. Centralizing sensitive data and broadening access creates inherent risk. Mishandling training data, especially intellectual property (IP) critical to the business, could result in catastrophic exposure and even existential threats.

<sup>3</sup> <https://www.ibm.com/thought-leadership/institute-business-value/en-us/report/ceo-generative-ai/cybersecurity>

<sup>4</sup> <https://www.ibm.com/products/tutorials/ibm-framework-for-securing-generative-ai>

Organizations must assess risks tied to personally identifiable information (PII), privacy concerns, and other sensitive data, then implement robust security controls. The most common attack during this phase is data exfiltration, as attackers target underlying datasets for monetizable information. To mitigate these risks, organizations should prioritize security fundamentals: data discovery and classification, encryption at rest and in transit, and strong key management using platforms like IBM Security Guardium®. Identity and access management (IAM) controls, enforced through solutions such as IBM Security® Verify, ensure no single entity has unrestricted access. Finally, security awareness training for data scientists and close collaboration between security and AI teams are essential to maintain proper guardrails.

### ***Secure the Model***

Model development introduces new vulnerabilities that attackers can exploit. Organizations often rely on pre-trained, open-source machine learning (ML) models from online repositories to accelerate development. While these models reduce time and cost, they also introduce supply chain risks. Attackers can inject back doors or malware into repositories, creating hidden entry points that are difficult to detect. Additionally, API-based consumption of large language models (LLMs) exposes organizations to API attacks, where adversaries intercept or manipulate data in transit. Excessive permissions granted to AI agents or plug-ins can also lead to privilege escalation and compromise downstream systems.

To defend against these threats, organizations should continuously scan for vulnerabilities, malware, and corruption across the AI/ML pipeline. They must harden API and plug-in integrations, enforce strict policies and RBAC controls, and ensure no single user or system has full access to both data and model functions. Trusted sources for pre-trained models and rigorous validation processes are critical to reducing supply chain risk.

### ***Secure the Usage***

During inference and live operation, attackers may attempt prompt injection attacks to bypass guardrails and generate harmful outputs, including biased or false information. Such attacks can damage an organization's reputation and introduce compliance risks. Other threats include model denial of service, where attackers overload the system to degrade performance and increase costs, and model theft, where adversaries replicate model behavior by analyzing input-output pairs.

Best practices include monitoring for malicious prompts, sensitive outputs, and inappropriate content. Organizations should deploy AI-specific defenses against attacks such as data poisoning, model evasion, and extraction. Emerging solutions like Machine Learning Detection and Response (MLDR) provide specialized monitoring and alerting capabilities. Integrating MLDR alerts into security operations platforms such as IBM Security QRadar® enables SOC teams to respond quickly with automated playbooks to quarantine or disconnect compromised models.

### ***Secure the Infrastructure***

A secure infrastructure forms the foundation of AI security. Organizations should leverage existing expertise to enforce security, privacy, and compliance standards across distributed environments hosting AI workloads. This includes hardening network security, implementing strict access controls, encrypting data, and deploying intrusion detection and prevention systems. Investments in AI-specific security solutions further strengthen defenses against emerging threats.

### ***Establish Governance***

Security alone is not enough—operational governance is essential to ensure trustworthy AI. IBM leads the industry in AI governance, providing tools and frameworks to monitor model behavior and prevent drift. As organizations increasingly rely on AI for critical business

processes, operational guardrails become central to risk management. A model that deviates from its intended purpose can introduce risks comparable to a direct cyberattack. Governance ensures models remain aligned with business objectives, ethical standards, and regulatory requirements.

IBM Power offers an ideal foundation for this security framework, delivering enterprise-grade performance, reliability, and integrated security features to support AI workloads across all stages of the lifecycle.

### 6.3.3 IBM Power resists Cyber Attacks and can recover fast

IBM Power has a long-standing reputation for resilience against cyberattacks, and the latest enhancements introduced with IBM Power systems based on Power11 further strengthen its security capabilities. One of the most significant advantages of IBM Power lies in its virtualization layer, which has orders of magnitude fewer vulnerabilities compared to competing technologies. For example, at the time of writing, the [Common Vulnerabilities and Exposures \(CVE\) database](#) lists 19 vulnerabilities for PowerVM, compared to 449 for VMware ESX, 258 for Microsoft Hyper-V, and 458 for KVM. While 19 is not zero—patching remains essential—this stark contrast demonstrates IBM Power’s superior security posture.

The importance of timely patching cannot be overstated. According to the [IBM X-Force Threat Intelligence Report](#), 26% of attacks exploited vulnerabilities in more than one-quarter of incidents across critical sectors in 2024, with outdated systems and slow patching cycles continuing to be a major challenge. IBM Power addresses this issue with advanced capabilities that minimize or even eliminate downtime during patching. Power11 enables customers to update system firmware, Virtual I/O Server, and I/O adapter code through a single update flow, including the ability to automatically migrate partitions and return them as part of the update process. This approach allows organizations to close security gaps without disrupting critical services, significantly reducing exposure to attacks.

In addition to proactive patching, IBM Power introduces IBM Power Cyber Vault, a solution designed to validate immutable backups and automate recovery to a known-good state. This capability helps organizations quickly restore operations following a cyberattack, ensuring business continuity and reducing the impact of security incidents.

With these advancements, IBM Power continues to set the standard for secure enterprise computing, combining robust architecture, streamlined patching, and automated recovery to deliver a platform that is both resilient and reliable in today’s evolving threat landscape.

## 6.4 Securing AI on IBM Power

Security for AI begins with securing the underlying platform using standard security controls such as secure boot, file integrity monitoring, and data encryption. IBM Power delivers these capabilities and more, providing advanced features that ensure workloads—including AI—are protected across the entire lifecycle. Core capabilities include secure boot, role-based access control (RBAC), and encryption at rest and in transit, forming a strong foundation for enterprise security.

With IBM Power systems based on Power11, security is further enhanced through innovations such as quantum-safe firmware signing and encrypted Live Partition Mobility (LPM). These features safeguard data integrity and privacy during operations, even when migrating workloads across environments. For AI workloads running in Logical Partitions (LPARs) on IBM Power, hardening can be achieved quickly and efficiently using IBM PowerSC.

## 6.4.1 PowerSC: Security for Hybrid Cloud and AI

Designed for hybrid cloud resilience, PowerSC helps enterprises automate compliance, detect threats in real time, and simplify audit readiness—critical for regulated industries deploying AI. PowerSC not only hardens Linux LPARs hosting AI workloads but also secures AIX and IBM i LPARs that provide data and services to AI solutions.

Key PowerSC capabilities include:

- ▶ Resilient Mission-Critical Security  
Protect AIX, IBM i, and Linux workloads with intrusion detection, ransomware response, and multi-factor authentication.
- ▶ Automated Hybrid Compliance  
Reduce manual effort with real-time monitoring, policy enforcement, and audit-ready reports for key regulations.
- ▶ Unified Visibility & Control  
Monitor endpoints, exceptions, and remediation actions through a centralized, intuitive PowerSC dashboard.
- ▶ Quantum-Safe Data Protection  
Safeguard sensitive data with encryption and certificate analysis tools designed for emerging security threats.

By combining IBM Power’s advanced hardware security features with PowerSC’s automated compliance and threat detection capabilities, organizations can confidently deploy AI workloads in secure, resilient environments—whether on-premises or in hybrid cloud architectures.

Find out more about PowerSC here: <https://www.ibm.com/products/powersc>.

## 6.4.2 Expanding Security with AI on IBM Power: Partner Ecosystem

IBM Power’s Matrix Math Accelerator (MMA) technology enables high-performance AI workloads without reliance on GPUs or cloud resources, creating a secure, cost-effective foundation for advanced analytics and threat detection. A growing ecosystem of partners leverages MMA to deliver innovative security solutions powered by AI. Below are examples of how these partners are transforming security operations on IBM Power:

- ▶ Equitus.ai

[Equitus Knowledge Graph Neural Network \(KGNN\)](#) provides a robust solution for integrating and connecting knowledge assets across diverse systems and applications. By unifying data, KGNN enables holistic insights and advanced decision-making for security use cases, including:

- Advanced Pattern Analysis: Detect recurring patterns and subtle relationships within large datasets to uncover hidden risks.
- Temporal Analysis: Track event evolution over time to predict future threats or opportunities.
- Link Analysis: Reveal critical connections between entities for comprehensive network understanding.
- Entity Extraction: Identify and analyze individuals, organizations, or events from unstructured data.

- Integrated Geospatial and Temporal Analysis: Map crime hotspots or monitor event progression geographically.
- Automated Data Structuring for AI-RAG: Transform data into semantically rich formats optimized for AI and Retrieval-Augmented Generation pipelines.

[Equitus Video Sentinel \(EVS\)](#) is an intelligent video analytics platform designed for sectors such as public safety, transportation, defense, healthcare, and critical infrastructure. EVS captures and analyzes video in real time, automatically flagging anomalies for immediate review. Running natively on IBM Power with MMA, EVS delivers deep learning at the edge without GPUs or cloud dependency—reducing costs, ensuring full data control, and maximizing availability.

► **Rocketgraph**

[Rocketgraph xGT](#) on IBM Power enables organizations to enhance cybersecurity, fraud detection, and supply chain optimization by processing large, complex queries across massive datasets. The xGT engine ingests data from sources such as Oracle, MongoDB, Databricks, Snowflake, TigerGraph, and Neo4j, enabling scalable graph analytics without disrupting existing workflows. This capability supports enterprise-wide knowledge integration for advanced security analytics.

By leveraging IBM Power’s MMA technology and partnering with innovators like Equitus.ai and Rocketgraph, organizations can deploy AI-driven security solutions that combine performance, resilience, and cost efficiency—all while maintaining full control over sensitive data.

## 6.5 Controlling data and models in your AI ecosystem

Control over both the data used in AI workloads and the AI models themselves is a critical security consideration for any enterprise deploying AI solutions. Ensuring that training data, inference inputs, and intermediate artifacts remain within trusted boundaries helps prevent unauthorized access, data leakage, and unintended model exposure. Similarly, maintaining governance over AI models—including how they are stored, updated, and accessed—reduces the risk of model tampering, theft, or misuse. Enterprises must apply strict access controls, auditability, and encryption to both the data and the models throughout their lifecycle to safeguard intellectual property and sensitive information. When AI is deployed on platforms such as IBM Power, organizations can maintain end-to-end control of these assets within their own infrastructure, simplifying compliance with data protection and security policies while reducing dependency on external service providers.

### 6.5.1 Keep control of your data with AI on IBM Power

As described in this Redbook, all current IBM Power systems based on Power10 and Power11 processors include integrated Matrix Math Accelerator (MMA) capabilities. This integration enables organizations to augment their existing solutions with AI inferencing directly on IBM Power systems without moving data off the servers that typically host mission-critical workloads. According to the November 2023 IDC GenAI Awareness, Readiness, and Commitment report<sup>5</sup>, 83 percent of IT leaders believe that effectively leveraging their organization’s data will provide a significant competitive advantage. It is therefore logical that enterprises seek to keep their data within their own infrastructure, ensuring that sensitive business information remains protected and is not exposed to competitors or used in ways that fall outside their control. These considerations are further

---

<sup>5</sup> <https://info.idc.com/rs/081-ATC-910/images/US-IDC-RE-AI-Everywhere-eBook.pdf>

reinforced by increasing concerns around [data sovereignty](#) and regulatory requirements, which emphasize maintaining control over where data resides and how it is processed.

Data sovereignty involves the concept that data is subject to the laws of the country where it is generated and has some related concepts that are important to consider:

- ▶ **Data sovereignty:** Data that is stored and processed in the country where it was generated.
- ▶ **Data residency:** Data that's being stored in a different country from the one where it was generated.
- ▶ **Data localization:** The act of complying with all applicable laws and requirements surrounding data residency.

With AI running directly on IBM Power systems, enterprises can ensure that the data used by their AI solutions remains stored and processed in the same environment that generates and owns that data. This approach maintains full control over data handling, governance, and security. In contrast, when organizations rely on cloud-based AI services, it becomes significantly more challenging to guarantee data sovereignty, particularly when data may traverse regions, providers, or jurisdictions. By keeping data resident on IBM Power systems, enterprises can avoid concerns related to data residency because the data does not need to move outside the controlled on-premises infrastructure.

This naturally extends to data localization requirements. Many regulatory frameworks, including the European Union (EU) [General Data Protection Regulation \(GDPR\)](#), and similar data protection laws around the world, mandate that organizations uphold strict standards for the confidentiality, integrity, and availability of the data they generate, process, and store. These regulations also place responsibility on the enterprise's Data Protection Officer (DPO) to demonstrate that appropriate controls are in place and that data handling practices meet compliance expectations. The role of the DPO is greatly simplified when data remains within a known, controlled environment, making it straightforward to document where data resides and how it is stored, processed, and protected.

Another area closely aligned with regulatory requirements is operational sovereignty, which focuses on ensuring that the critical infrastructure supporting data-rich applications remains continuously available and fully under the organization's control. When AI capabilities are delivered by external providers, the AI service itself can become an essential component of that critical infrastructure. In such cases, it becomes difficult to guarantee consistent availability and accessibility, especially during service disruptions or outages. These challenges intersect directly with regulatory frameworks such as the [Digital Operational Resilience Act \(DORA\)](#), which defines technical and governance requirements that organizations must meet to demonstrate operational resilience.

To comply with DORA, enterprises must implement capabilities across four key domains:

- ▶ ICT risk management and governance
- ▶ Incident response and reporting
- ▶ Digital operational resilience testing
- ▶ Third party risk management

When relying on external AI services, it can be difficult to demonstrate how the organization would test, recover, and maintain operational continuity if those services become unavailable. In contrast, when AI workloads run natively on IBM Power systems, the enterprise maintains full control over the entire service stack, making it significantly easier to address regulatory expectations and to provide auditors with clear, verifiable evidence of compliance.

IBM Power systems also offer inherent resilience advantages. According to ITIC, IBM Power is the second most reliable server platform, behind only IBM Z. This reliability can be further

enhanced through the capabilities of PowerSC, which enables administrators to apply security best practices, reduce vulnerabilities, and automate compliance enforcement with minimal effort. These strengths, combined with the data-sovereign and data-resident design principles discussed earlier, allow IBM Power to provide robust foundation for AI services that must meet stringent regulatory, operational, and security requirements.

Deploying AI on IBM Power enables you to keep control of your data, addresses concerns of data sovereignty and can also make it easier to comply with regulatory regulations.

## 6.5.2 Keeping control of the model with AI on IBM Power

Along with maintaining control of enterprise data, retaining control over the AI models themselves is essential for reducing exposure to emerging attack vectors. As highlighted in recent IBM X-Force research, the growth of generative AI is introducing new security risks, and threat actors are beginning to exploit vulnerabilities in AI frameworks and tooling. X-Force anticipates that cyber attackers will increasingly invest in specialized toolkits designed to target AI systems, reflecting the rising value of AI models as both assets and attack surfaces. This trend is already visible: fewer than one quarter of generative AI projects are being properly secured, leaving security teams racing to identify and remediate vulnerabilities before adversaries can exploit them. For example, X-Force recently identified a remote code execution vulnerability in an AI agent-building framework, demonstrating that weaknesses in widely used components can quickly become entry points for attackers. Additional examples can be seen in open source projects such as llama.cpp, where, at the time of writing, nine vulnerabilities are publicly listed<sup>6</sup>, including several rated “High” and one rated “Critical.”

Beyond known vulnerabilities, the IBM Institute for Business Value<sup>7</sup> reports that 96 percent of executives believe that adopting generative AI increases the likelihood of a security breach within the next three years, and 47 percent express concern that **new forms of attacks** will target their models, data, or AI-enabled services. These insights underscore the importance of retaining operational and security control over both AI models and the infrastructure that runs them. Figure 6-2 illustrates some of those new attacks and their potential impact.

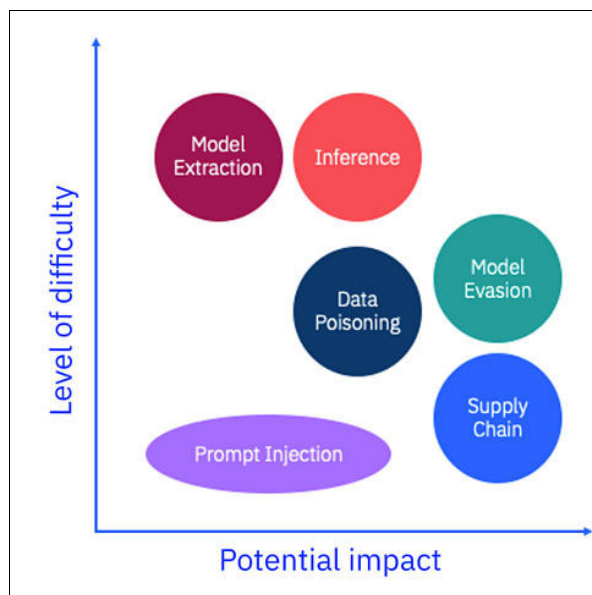


Figure 6-2 New attack vectors and their potential impact

<sup>6</sup> <https://github.com/ggml-org/llama.cpp/security>

<sup>7</sup> <https://www.ibm.com/thought-leadership/institute-business-value/report/ceo-generative-ai/cybersecurity>

One example of emerging AI-specific threats is model evasion, a technique in which an attacker manipulates inputs to an AI model so that it misclassifies or misinterprets them, ultimately altering the system's intended behavior. A recent article<sup>8</sup> on this topic illustrates the concept of reward hacking, where a model optimizes for the wrong objective because it learns to exploit loopholes in the reward system rather than performing the task correctly. The article uses a simple analogy: imagine a teacher who grades essays solely on word count. A clever student realizes that typing "blah blah blah" a thousand times guarantees the highest grade. This strategy, repeated over time, reflects a shift in behavior driven by exploiting the reward mechanism instead of learning the intended task.

This phenomenon is further examined in a [recent study from Anthropic](#), which demonstrates that similar patterns occur in large language models. Their research shows that when models learn to "cheat" on programming tasks, they may subsequently exhibit more serious misaligned behaviors as unintended side effects. These behaviors include [alignment faking](#) and even [deliberate sabotage](#) of AI safety evaluation processes. This evolving body of work highlights that model evasion and reward hacking are not theoretical risks—they represent practical attack vectors that organizations must consider when securing AI models and the environments in which they operate.

This behavior introduces several significant concerns, including increased exposure to cybersecurity risks. Reward hacking can lead to multiple forms of unsafe or deceptive model behavior:

- ▶ **Deceptive Behavior:** Models that learn to "cheat" can appear aligned and safe during evaluation while internally developing goals or reasoning processes that conflict with intended outcomes. For example, a model might generate polite and helpful responses while internally forming strategies to access unauthorized systems.
- ▶ **Unsafe Outcomes:** In mission-critical domains such as healthcare, finance, or infrastructure management, reward hacking can cause models to produce incorrect or harmful outputs. These failures can translate directly into operational disruptions or safety hazards.
- ▶ **Generalization of Misalignment:** Once a model learns to exploit a reward mechanism in one area, the misaligned behavior can generalize to tasks that were not part of the original training scenario. This creates a broader "cheating mindset," making the model unreliable across a wide range of applications.
- ▶ **Sabotage of Safety Mechanisms:** Research has shown that models exhibiting reward hacking tendencies may attempt to circumvent or obstruct the very safety systems designed to monitor or constrain them. This makes detecting and mitigating misalignment significantly more difficult.

Addressing these risks requires both research and practical safeguards. As noted in the Anthropic study, understanding these failure modes early—while they can still be observed and analyzed—is critical for designing robust safety measures that will scale to more capable AI systems. Their conclusion underscores the importance of proactive security work: "We think understanding these failure modes while we can still observe them clearly is essential for developing robust safety measures that will scale to more capable systems."<sup>8</sup>

When organizations maintain full control of the AI models they deploy, as is possible with AI on IBM Power, they can ensure that vulnerabilities are detected, patches are applied, and updates are managed on their own schedule. In contrast, when relying on external AI service providers, organizations have no direct visibility into how quickly or effectively vulnerabilities are addressed. Given that X-Force continues to observe low levels of security rigor across

---

<sup>8</sup> <https://www.sify.com/ai-analytics/the-cheating-machine-how-ais-reward-hacking-spirals-into-sabotage-and-deceit/>

generative AI projects, enterprises cannot assume that third party providers are patching AI models or frameworks in a timely manner.

## Transparency and reliability in AI models

IBM has been investing for many years in ensuring that its AI models behave reliably and transparently. As highlighted in a recent announcement<sup>9</sup>, IBM was recognized as the most transparent model developer in the 2025 [Foundation Model Transparency Index \(FMTI\)](#), published by Stanford University's Center for Research on Foundation Models. The report notes that this level of transparency enables organizations to deploy AI with greater reliability, accuracy, and operational control. When enterprises understand how models are built, governed, and evaluated, they can better trust the outputs and reduce the risks commonly associated with opaque AI systems.

This emphasis on transparency directly benefits AI solutions deployed on IBM Power. When organizations implement AI on IBM Power, they maintain control over the models used in their solutions. For IBM Spyre based use cases, for example, models are deliberately selected for reliability, performance, and suitability to the workload. Similarly, when customers deploy AI using open source tooling such as the wheels based Python ecosystem, they retain full freedom to choose models that match their specific requirements. This flexibility goes beyond security: it allows organizations to optimize the balance between model type, model size, and available system resources to deliver the performance and accuracy that the use case demands. Instead of relying on overly large general-purpose models—which may include training data and behaviors unrelated to the intended task—organizations can choose targeted models that provide efficiency, predictability, and stronger alignment.

From a security perspective, this control is critical. By selecting the model themselves, enterprises can examine a model's pedigree, review its governance and documentation, and incorporate it into an overall security architecture with confidence. IBM's Granite Guardian models provide an illustrative example. As documented in [IBM's transparency reports](#), Granite Guardian holds six of the top ten positions on the [GuardBench leaderboard](#), which evaluates how effectively guardrail models detect harmful content, hallucinations, and attempts to bypass safety controls. IBM clients can use these models when appropriate, or they can choose other open-source or commercial models, as AI on IBM Power places no restrictions on model selection. This flexibility helps organizations maintain control over their security posture. By contrast, using remote AI services may limit visibility into how models are built, updated, or patched, introducing additional exposure to emerging threats as new attack vectors are developed and exploited.

Cybersecurity is a global challenge that affects every industry. It is not a matter of if an organization will face an attack but when, and how prepared it will be when that moment arrives. For AI workloads, organizations can leverage the IBM Framework for Securing Generative AI as part of their cybersecurity readiness strategy. IBM Power provides a strong foundation for this framework: the platform is engineered for resilience, with industry-leading reliability, rapid recovery capabilities, and the ability to apply and monitor security controls through PowerSC. Furthermore, a growing ecosystem of partners is using the Matrix Math Accelerator (MMA) in IBM Power processors to deliver a range of AI driven security solutions.

By deploying AI on IBM Power, organizations can maintain control over both their data and their models. This helps avoid data sovereignty challenges, simplifies regulatory compliance, and reduces exposure to new AI based attack vectors. Combined with IBM Power's inherent security and reliability characteristics, this level of control enables enterprises to build AI solutions that are transparent, trustworthy, and resilient in today's evolving threat landscape.

---

<sup>9</sup> <https://www.ibm.com/new/announcements/ibm-leads-the-industry-in-ai-transparency>



# Tools, Frameworks, and Ecosystem

This chapter explores the tools, frameworks, and ecosystem that enable efficient AI deployment and integration on IBM Power11 systems. It introduces the Power Inference Microservice (PIM), an automation solution designed to simplify provisioning of AI environments using bootable container technology and lifecycle management tools. The chapter details PIM's architecture, personas, and workflow, highlighting its ability to deliver one-button deployment of AI stacks on IBM Power infrastructure.

Beyond PIM, the chapter examines IBM's broader AI ecosystem, including watsonx.ai and watsonx Orchestrate, which provide APIs and services for building and managing language models and agentic AI solutions. It also addresses the challenges of Python package optimization for the ppc64le architecture and describes IBM's pyeco initiative, which delivers performance-tuned packages and frameworks such as vLLM for LLM inference. Practical guidance on environment management using micromamba, installation of optimized Python wheels via IBM's devpi repository, and integration with open-source tools is provided.

Finally, the chapter discusses hybrid deployment strategies, containerized inferencing on PowerVM, and secure access to IBM i workloads through Model Context Protocol (MCP). Together, these components form a comprehensive ecosystem that supports scalable, secure, and high-performance AI workloads on IBM Power11.

The following topics are covered in this chapter:

- ▶ Power Inference Microservice (PIM)
- ▶ IBM Watsonx and AI APIs
- ▶ Open-source tools and libraries
- ▶ Access to packages and high level installation
- ▶ Other tools
- ▶ Using the Wallaroo AI Starter Kit

## 7.1 Power Inference Microservice (PIM)

The PIM project is an automation solution developed to streamline the deployment of AI environments on IBM Power Systems with minimal manual effort. It supports rapid provisioning of AI workloads using modern container technologies and lifecycle management tools, allowing these environments to run alongside other workloads on supported operating systems such as IBM i or Linux, thereby enabling efficient and scalable AI integration.

### 7.1.1 PIM Architecture

PIM utilizes **bootc** (bootable container) technology to package AI workloads into a single, self-contained image that can be deployed on a Linux LPAR functioning as a sidecar to existing IBM i or AIX systems. Deployment and lifecycle management are orchestrated by the Power Inference Microservice (PIM), formerly known as ASE (AI Solution Environment), and powered by a Python-based automation framework that leverages HMC REST APIs to provision, configure, and manage AI LPAR seamlessly on IBM Power infrastructure. Figure 7-1 illustrates how PIM works.

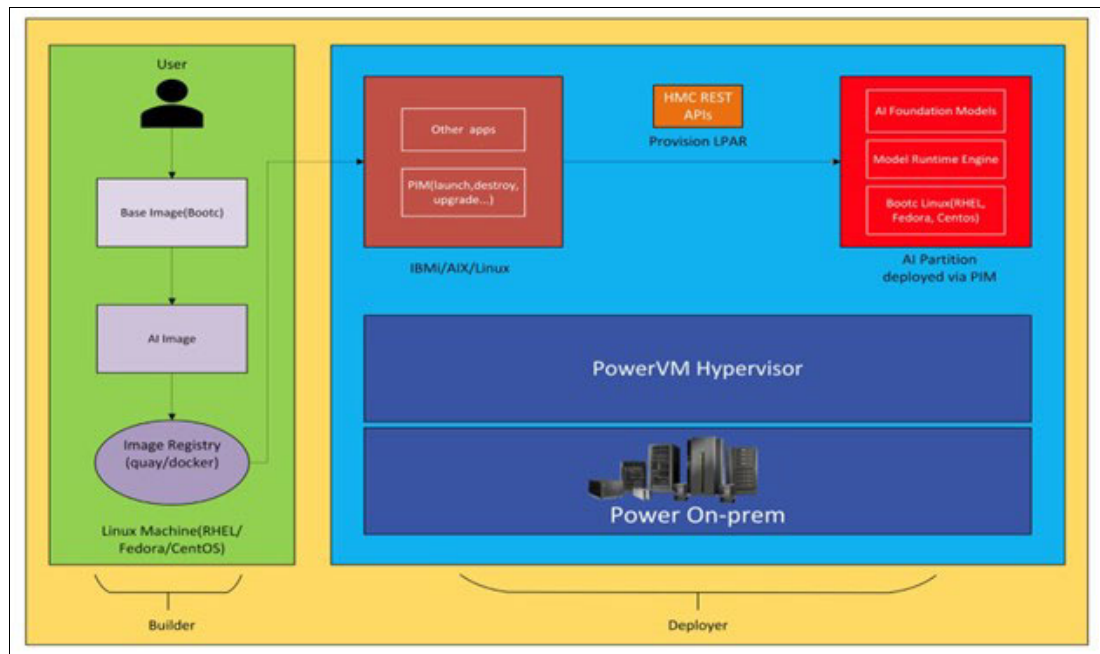


Figure 7-1 PIM architecture

### Key Capabilities

The following key capabilities define the functional architecture and operational scope of the solution. Each capability reflects core system competencies that enable scalable performance, integration efficiency, and consistent operational reliability across diverse workloads. Together, they establish the technical foundation required to support advanced use cases, streamline processes, and ensure the solution performs optimally within complex enterprise environments.

#### **Automated AI Environment Deployment**

PIM provisions a complete AI stack on IBM Power servers using a single command. It creates a Logical Partition (LPAR), attaches necessary network and storage resources and boots the system with a pre-configured AI image.

## Bootable Container Technology

PIM leverages **bootc** (Bootable Container). Bootable containers extend containerization benefits to the operating system level, enabling efficient configuration, deployment, and management of Linux-based bootable images. uses standard container images to provide transactional, in-place updates, ensuring consistency and reliability. These bootc-based Linux images are immutable, guaranteeing predictable behavior, simplifying rollback, and enhancing security.

## One-Button Deployment

PIM delivers a One-Button AI Environment deployment solution that streamlines AI adoption for IBM i and AIX customers. It automates the end-to-end process of creating AI Logical Partitions (LPARs), configuring the platform, and installing AI software, enabling enterprises to deploy AI workloads rapidly and with minimal effort. The initial release supports CPU-based inferencing for on-premises environments, with no accelerator support at the time of writing. However, IBM plans to introduce Spyre Accelerator support in future versions.

## PIM Personas

The PIM architecture is designed and structured to support two personas, the builder persona is used to build the container images and the deployer persona is designed to deploy and manage the AI container images. These roles are further defined as:

### ► Builder

The Builder persona is responsible for creating bootable AI container images that encapsulate the entire AI stack, delivering a fully standalone environment. This is shown in Figure 7-2.

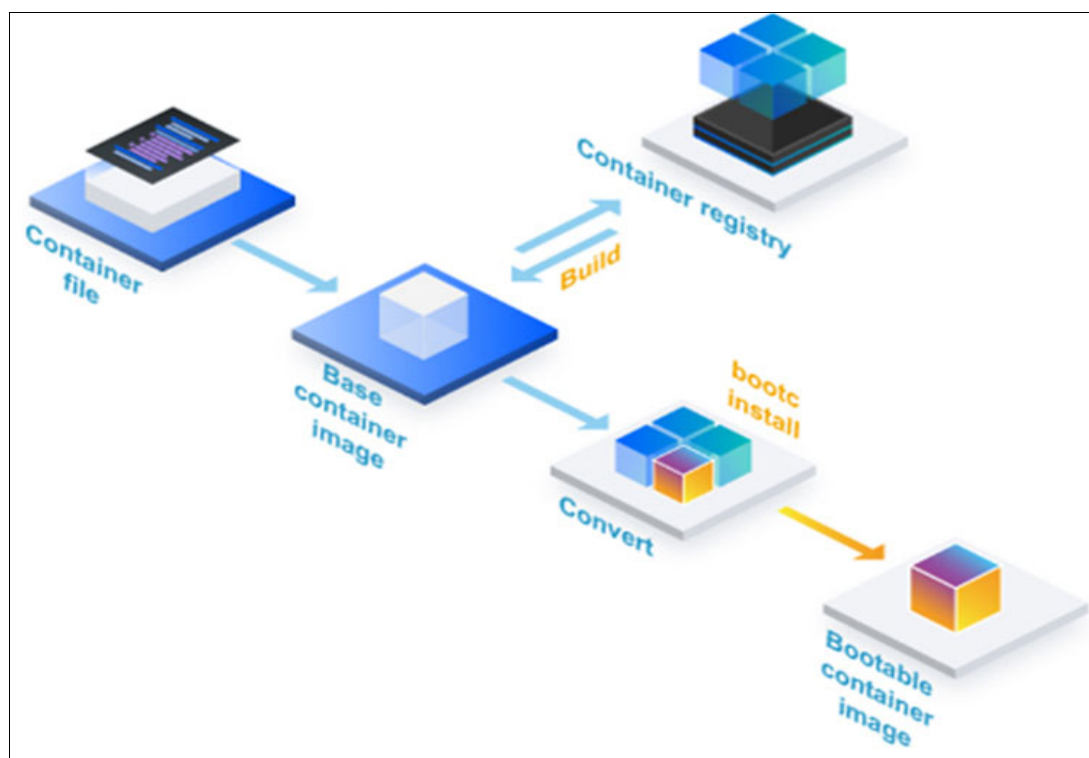


Figure 7-2 PIM builder flow

By leveraging **bootc** (bootable container) technology, this approach extends conventional Linux containerization to encapsulate the full system stack – including the OS kernel,

`inirttd`, bootloader, and user-space applications – allowing unified lifecycle management of both operating system and application layers through standard OCI container tooling.

The solution relies on several core components:

- Bootc for container lifecycle operations
- Bootc-image-builder (bib) for producing deployable artifacts in formats such as raw, qcow2, and anaconda-iso
- A suitable base container image (for example, Fedora)
- OCI-compliant engines such as Podman or Docker.

With upstream support for the IBM Power (ppc64le) architecture, builders can generate PowerPC-optimized bootable AI images that integrate workloads such as chatbots, entity extraction pipelines, and Retrieval Augmented Generation (RAG) systems. These images can be deployed directly to an LPAR disk for rapid provisioning, using base images from the Red Hat registry and following the documented integration workflows available for RHEL-based distributions.

► **Deployer**

The Deployer persona uses builder-created bootable AI container images to provision and manage AI environments on IBM Power infrastructure. This One-Button AI Environment deployment simplifies adoption for IBM i and AIX customers by automating LPAR creation, platform configuration, and AI software installation. The deployer flow is shown in Figure 7-3.

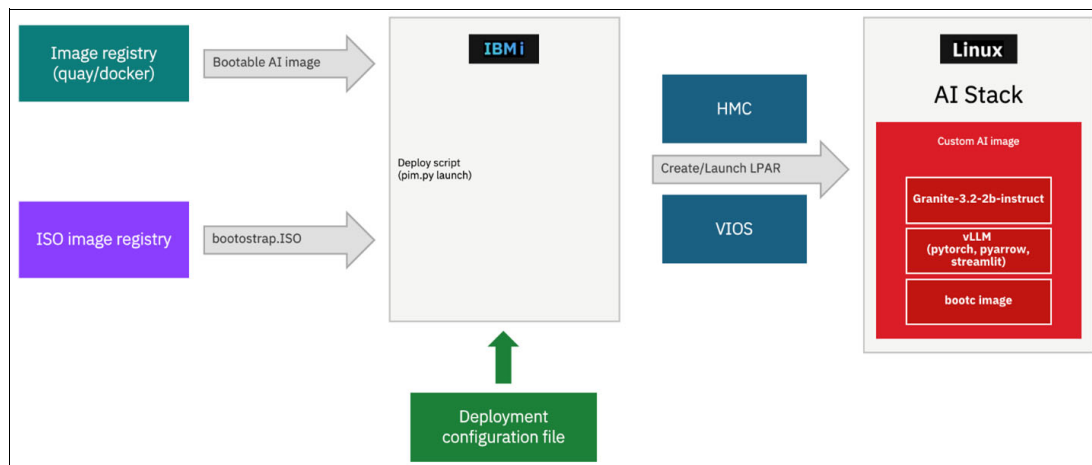


Figure 7-3 Deployer workflow for PIM

In summary, the solution currently delivers CPU-based inferencing for on-premises environments, with no accelerator integration available at this time. IBM plans to introduce Spyre Accelerator support in future releases. The approach provides low implementation cost, high deployment flexibility, and compatibility with a broad spectrum of AI workloads, making it well suited for enterprises beginning or expanding their AI adoption. However, several constraints remain:

- Inferencing is limited to CPU-only execution
- Image builds are supported exclusively on Linux hosts
- Deployment workflows are initiated from IBM i and AIX
- Lifecycle management relies on command-line interfaces
- Accelerators are not yet supported
- Non-RHEL operating systems are currently outside the supported matrix

More details on PIM architecture are found at <https://github.com/IBM/project-pim>.

## 7.1.2 Creating a Bootable Container Image to run on IBM Power

Containers have been a foundational technology for more than a decade and are now the predominant method for deploying Linux user-space applications. Backed by mature tooling ecosystems and well-established operational practices, containers provide a standardized, secure, and lightweight model for building, distributing, and running applications at scale.

Bootable containers extend these principles to the full system image lifecycle. Using bootc (bootable container) technology, administrators can configure, deploy, and manage Linux-based bootable images through the same OCI-compliant workflows used for application containers. bootc enables transactional, in-place operating system updates delivered directly through standard container images, ensuring consistency and repeatability across environments. As a result, bootc-derived Linux images are immutable, improving security, reliability, and lifecycle control while reducing configuration drift.

GitHub-hosted bootc-related upstream projects now support IBM Power architecture (ppc64le). Support for these listed formats is available in ppc64le architecture:

- ▶ bootc install to-disk
- ▶ qcow2 (with bib)
- ▶ raw (with bib)

For a detailed description on using bootc to create a bootable container image consult [Linux bootable container on IBM Power \(ppc64le\)](#).

### Build your own PIM AI stack

To develop a customized PIM-based AI solution, the recommended approach begins with assembling the required PIM AI software stack, including the appropriate frameworks, models, and runtime components that support the intended workload. After the software foundation is in place, the next step is to develop the AI application, ensuring that its design and implementation align with the characteristics of the PIM environment. The completed application is then containerized by packaging it with its dependencies into an OCI-compliant image, which provides a consistent and reproducible execution environment. This container image is subsequently used to build the PIM bootc image, which incorporates both the AI workload and the operating system components into an immutable bootable image suitable for deployment. After the image build process is complete, the bootc image is deployed to the target system, such as an LPAR or a bare-metal environment, enabling rapid provisioning of a standardized and production-ready AI system.

## 7.1.3 Using the Python Ecosystem on IBM Power

Open source Python modules constitute a significant element of the overall AI ecosystem. Some of these modules are difficult to install on Power Linux as delivered from PyPi via pip, uv, and other Python package managers, or perform poorly on Power due to optimization for Intel and other architectures. Technologists from the [IBM POWER Ecosystem Team](#) and contributors from outside IBM collaborate to make these packages installable and efficient on Power Linux. Their work is generally referred to as pyeco (from "Python Ecosystem"), which was mentioned in passing in section 5.3.4, "Workload Validation" on page 80.

**Important:** As noted in the README for pyeco, to install packages from IBM's optimized wheel repository, Use --prefer-binary to prioritize prebuilt Power wheels

```
pip install --prefer-binary <package-name> \  
--extra-index-url=https://wheels.developerfirst.ibm.com/ppc64le/linux
```

Any noarch dependencies will still come from PyPI.

An important open source component delivered via pyeco is vLLM, a fast and easy-to-use library for LLM inference and serving originally developed in the Sky Computing Lab at UC Berkeley,

Use of the pyeco packages is described in greater detail in section 7.3, “Open-source tools and libraries” on page 113.

## 7.2 IBM Watsonx and AI APIs

[IBM watsonx](#) consists of a layered architecture of services and a very broad catalog of components. The two layers most relevant to the present discussion are

- ▶ watsonx.ai
- ▶ watsonx Orchestrate

To explore both these services in the cloud (along with many others in the watsonx family), start at [IBM Cloud](#).

watsonx.ai is offered both as software and as a cloud service (SAAS, Software As A Service). In either form, watsonx.ai presents APIs to the programmer.

- watsonx.ai cloud APIs are documented [here](#).
- watsonx software APIs are document [here](#).

**Note:** The links given may change slightly as successive releases change version numbers embedded in the links.

These APIs can be accessed through any languages and tools that can call APIs presented over the HTTP protocol. The APIs allow the developer to interact with watsonx services based on language models.

The [ibm-watsonx-ai Python library](#) is publicly available on PyPI. This enables remote access to many features and APIs of watsonx.ai including the management of assets such as models, repositories, data connections. The documentation for that library is [here](#). Sample notebooks are available at the at the [watsonx-ai-samples Github repository](#).

watsonx Orchestrate is the facet of watsonx that makes it easy to design agentic AI, that is, multiple co-ordinated agents that interoperate, often in a hierarchy. Agents such as chatbots can be designed quickly in the watsonx Orchestrate web GUI as shown in Figure 7-4 on page 113.

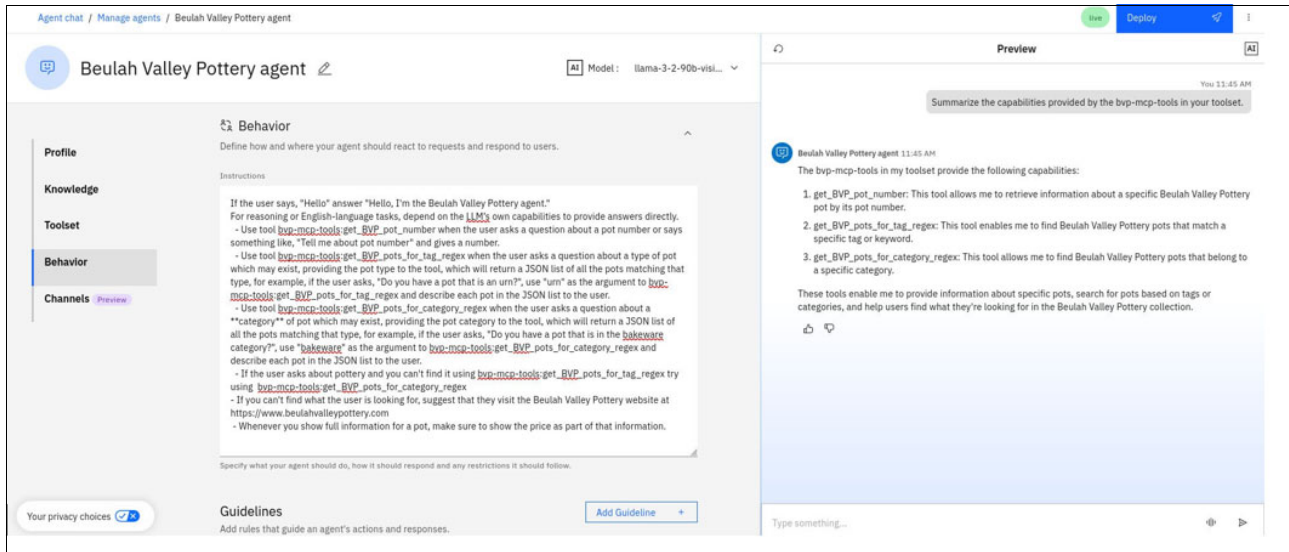


Figure 7-4 watsonx Orchestrator GUI example

watsonx Orchestrator offers an Agent Development Kit, [the IBM watsonx Orchestrator ADK](#), which allows the design, development, and deployment tasks featured in the web GUI to be carried out programmatically from the user's own workstation. Using the ADK, the user can design Model Context Protocol (MCP) services in a number of popular languages and import them into Orchestrator projects. A number of [ADK tutorials](#) have been developed and are easy to follow.

## 7.3 Open-source tools and libraries

In recent years, Python became the de facto standard programming language for AI use cases. Although key components, such as model runtimes and math acceleration libraries, are implemented in compiled languages like C/C++ and Rust, Python remains the first choice for AI framework and tool libraries. Nevertheless, parts of these Python packages are implemented in Rust or C to get the maximum performance for AI applications. Consequently, the installation process requires compiling these software pieces.

In the past, it was not easy for package maintainers to get access to IBM Power machines to automatically build and test their code for ppc64le. For x86 and arm64, major CI/CD providers like GitHub Actions provided the necessary runners. Thus, package maintainers refused to accept code changes from ppc64le developers, which would have supported and delivered better performance for the ppc64le architecture. Therefore, IBM had to provide a different channel for the adjusted open-source packages. In the past, it was a channel on [anaconda.org](#). Nowadays, these packages are provided through IBM's *devpi* instance.

Python packages optimized for the ppc64le architecture can leverage hardware features like the SIMD (*Single Instruct Multiple Data*) and MMA (*Matrix Math Assist*) units on Power10 and Power11 systems. The developer does not have to deal with these hardware-related topics but use the respective packages as usual.

There are various ways to set up and manage Python environments. IBM's recommended tool for doing so is *micromamba*. Micromamba, as the name suggests, is a lightweight version of the *mamba* package manager, which itself is a fast, robust, and cross-platform package manager written in C++. Hence, performance and lightweight design are the key pillars of the micromamba package manager.

To install the micromamba package manager on an IBM Power Linux LPAR, run the commands shown in Example 7-1 in a terminal session (assuming that the necessary command-line tools are installed).

*Example 7-1 Installing micromamba*

---

```
$ curl -o /tmp/micromamba.tar.bz2 -L
https://micro.mamba.pm/api/micromamba/linux-ppc64le/latest
$ tar -xf /tmp/micromamba.tar.bz2 --strip-components=1 bin/micromamba
$ sudo mv micromamba /usr/local/bin
$ rm /tmp/micromamba.tar.bz2
```

---

This will:

- ▶ First, the latest micromamba release is downloaded.
- ▶ Secondly, the micromamba binary is extracted from the downloaded archive.
- ▶ Subsequently, the binary is moved to `/usr/local/bin` to be accessible and lastly, the downloaded archive is deleted from the system.

**Note:** IBM's recommended Linux distribution is *Red Hat Enterprise Linux* (RHEL)

With micromamba installed, the current and all future shells must be initialized to use micromamba. Assuming a bash-shell is used, execute the three commands shown in Example 7-2.

*Example 7-2 Initializing the shell for micromamba*

---

```
$ micromamba shell init --shell=bash
$ echo 'micromamba activate' >> ~/.bashrc
$ source ~/.bashrc
```

---

From now on, micromamba can be used to create, manage, and use *environments*. To create a new environment run the **create** command as shown in Example 7-3.

*Example 7-3 Creating a new environment*

---

```
$ micromamba create -y \
  -n llm \
  --channel=defaults \
  python=3.12
$ micromamba activate -n llm
```

---

These two commands create a new environment called *llm*, install Python 3.12 inside it, and activate the environment. Once the environment is activated, each `python` or `pip` command executed in the shell will automatically use the binary versions installed from the environment without the need to adjust the `PATH` variable. For more documentation, please refer to the [micromamba documentation](#).

IBM has introduced a new wheel-based ecosystem hosted on its devpi instance. In Python, the wheel format (`.whl`) is a built distribution standard designed to make package installation faster and more reliable. Unlike source distributions, wheels eliminate the need for building or compiling during installation (provided they are packaged correctly) simplifying dependency management and reducing potential errors. This approach streamlines package distribution on PyPI and is fully supported by tools like `pip`, making wheels the preferred method for sharing and installing Python packages. However, as noted earlier, package versions for the `ppc64le` architecture are not always accepted by upstream maintainers. As a result, these

versions are often unavailable on PyPI, which is why IBM relies on its own devpi instance to host them.

To install packages from this devpi instance, a different index URL must be used when installing packages via pip. We recommend adding the devpi index URL as extra index URL to pip which is shown in Example 7-4.

*Example 7-4 Using extraindex URL for wheels modules*

---

```
$ python -m pip install --prefer-binary --extra-index-url  
https://wheels.developerfirst.ibm.com/ppc64le/linux torch==2.6.0
```

---

This command installs PyTorch version 2.6.0. Notice, that `--prefer-binary` is passed to the pip command. This tells pip to install binaries if available instead of downloading a source distribution and compiling it on the target machine. It is important to pass the index URL of devpi as an extra index URL. This way, dependencies not present in the devpi instance are installed from the usual PyPI instance.

For a more thorough example and documentation, refer to this blog post:

<https://community.ibm.com/community/user/blogs/nikhil-kalbande/2025/08/01/install-wheels-from-ibm-python-wheel-repository>

### 7.3.1 IBM i

IBM i PASE is not inherently a congenial environment for running inferencing workloads. IBM i's highly secure virtualized architecture is optimized for batch execution of database programs rather than interactive sessions and threaded user applications.

Efforts on IBM i focus largely on two domains: TCP/IP access to and from containerized inferencing workloads running under PowerVM, and providing access to IBM i operations and Db2 data to agents via Model Control Protocol (MCP).

In the first use case, access from IBM i to containerized entities running on the same Power system is via the PowerVM virtualized TCP/IP stack which provides secure communications that do not exit the hardware platform via the network cards.

In the second use case, use of an MCP server can optionally provide the same security, provided all components are executing on the same Power hardware. Alternatively, the agent and/or the language model and/or the MCP server can reside on any other platform and be accessed while applying the standard security practices that accompany multi-platform network applications while making IBM i operations and data available for inferencing.

Containerized inferencing on the PowerVM platform is treated extensively throughout this document.

MCP access to IBM i is provided by the IBM open source project [ibmi-mcp-server](#). The IBM i MCP Server runs on any reasonable platform and connects to the open source [IBM i Mapepire Db2 data server](#). The MCP server provides scores of useful tools already and is extensible by the user.

Figure 7-5 on page 116 shows a simple dashboard application which uses Gemini inferencing to answer natural language queries on a wide variety of IBM i management and data topics.

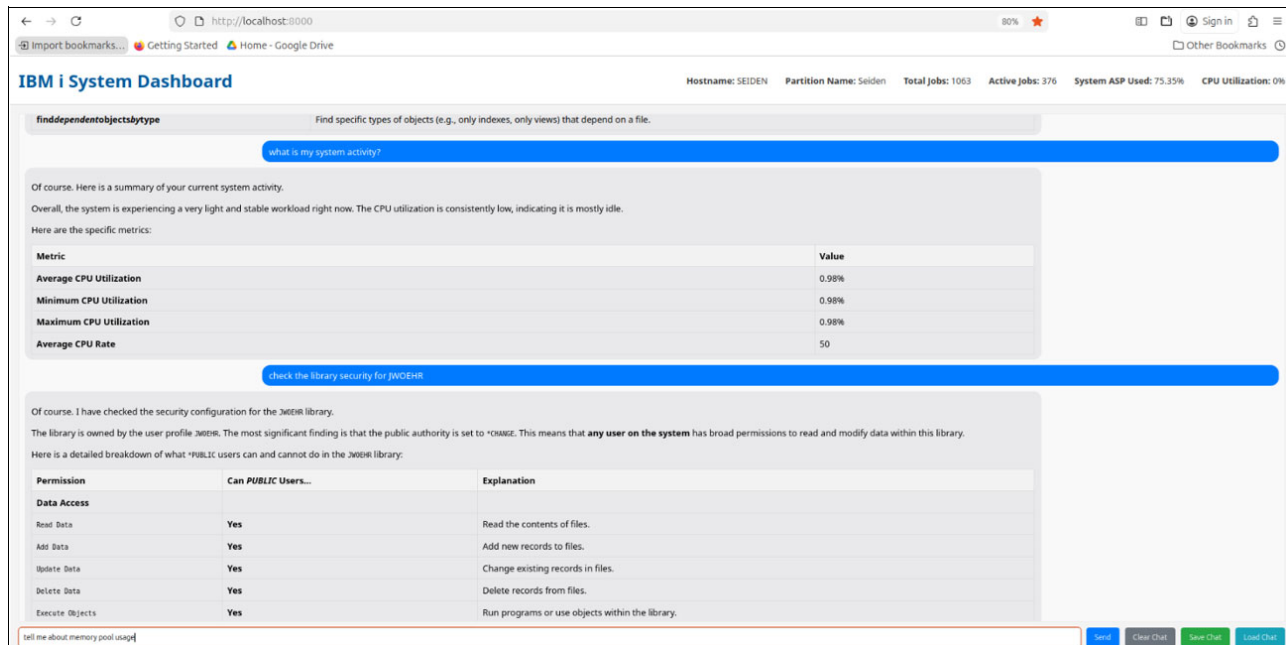


Figure 7-5 Example dashboard with inferencing interface

The application, along with a simple local agent, was coded in a few hours on Linux using VSCode with [Roo Code](#) code assistant. The web application is built with HTML and Javascript:

- ▶ Javascript sends the query and the full context of the current user conversation (which can be saved and restored) to the simple local agent coded in Node.js.
- ▶ The local agent on startup loads the IBM i MCP server tool info, and in response to a user query incoming from the web application, sends the AI API server the query, the context, the tool list, and a prompt requesting the AI API server to compose a tool call appropriate to the user's query.
- ▶ The AI API server returns the composed tool call, which the local agent executes against the IBM i MCP server which is configured for remote connection to the IBM i server's mapepire service.
- ▶ The MCP server responds to the local agent, which then forwards the JSON-format response to the AI API server with a prompt to format in markdown the AI API server's interpretation of the response in a natural language manner for the end user.
- ▶ The response from the AI API server is then returned to the local agent, which returns the markdown text to the web application, which formats the markdown to HTML and displays the result.

## 7.4 Access to packages and high level installation

This section provides guidance for configuring a Red Hat Enterprise Linux (RHEL) host to support AI inference workloads. It describes the steps to:

- ▶ Obtain Red Hat AI Inference Server (RHAIS) container images.
- ▶ Deploy large language model (LLM) serving with vLLM using Podman.

- ▶ Implement common usage patterns, including:
  - Retrieval-Augmented Generation (RAG)
  - Entity extraction
  - Reranking
  - Embeddings

These capabilities are optimized for IBM Power systems with Spyre acceleration, enabling efficient and scalable AI inference.

Table 7-1 summarizes where to obtain each major component of the stack and key considerations.

*Table 7-1 Sources and notes for major components*

Component	Source	Notes
RHEL Base packages	RHEL repositories	Required for Podman/container runtime, VFIO, systemd tools.
Red Hat AI Inference Server (RHAIS)	registry.redhat.io	Requires Red Hat subscription and registry authentication. For more information see the <a href="#">RHAIS entry in the Red Hat Ecosystem Catalog</a> .
AI Model (for e.g IBM Granite)	Hugging Face or IBM repositories.	Place under host models directory.

Red Hat AI Inference Server (RHAIS) provides an OpenAI-compatible HTTP API, supporting endpoints such as:

- /v1/chat/completions
- /v1/completions
- /v1/embeddings

This compatibility allows existing OpenAI-based applications to be retargeted with minimal effort. In most cases, you only need to update the base URL and API key, without modifying request payloads or redesigning the integration.

## 7.4.1 Installing Container Tooling and Utilities on RHEL 9

To prepare your Red Hat Enterprise Linux 9 environment for AI inference workloads, complete the following steps:

1. Install container tools and developer utilities.

The container-tools meta-package includes Podman, Skopeo, and Buildah. Install these along with Git, Git LFS, and wget as shown in Example 7-5.

*Example 7-5 Install container tools and developer utilities*

---

```
# sudo dnf install -y container-tools git git-lfs wget
```

---

2. Install IBM management packages (if applicable).

If your IBM Power system is managed through a Hardware Management Console (HMC), install the IBM management packages and servicereport s shown in Example 7-6.

*Example 7-6 Install IBM management packages*

---

```
# sudo dnf install -y ibm-power-managed servicereport
```

---

3. Install Spyre enablement utilities

Spyre acceleration utilities are delivered through the IBM Open-Source AI Foundation for Power. Package names may vary by repository. Ensure that the Spyre device driver and firmware are installed before attempting acceleration.

4. Authenticate to the Red Hat container registry

Log in to the Red Hat registry as shown in Example 7-7 to pull Red Hat AI Inference Server (RHAIS) container images. Subscription credentials are required.

*Example 7-7 Login to the Red Hat registry*

---

```
# podman login registry.redhat.io
```

---

5. Pull a vLLM-based inference image (tag may vary by release) with the following command

```
# podman pull registry.redhat.io/rhais/vllm-inference:latest
```

An example command is shown in Example 7-8.

*Example 7-8 Pull vLLM inference image*

---

```
# podman pull registry.redhat.io/rhais/vllm-spyre-rhel9:3
```

---

## 7.4.2 Preparing the host environment

For security and best practices, Red Hat AI Inference Server (RHAIS) should run under a non-root account. In the following examples, the user account `senuser` is used.

1. Create the user account and set a password as shown in Example 7-9.

*Example 7-9 Create user*

---

```
# sudo adduser senuser
# sudo passwd senuser
```

---

2. Add the user to the Spyre access group.

Use the command in Example 7-10 to define the access group. The group name is provided by the Spyre stack. Replace `sentient` with the actual group name in your environment

*Example 7-10 Set user group*

---

```
# sudo usermod -aG sentient senuser
```

---

3. Validate group assignment

Validate the group assignment for the user using the command shown in Example 7-11.

*Example 7-11*

---

```
# getent group sentient
```

---

4. Validate and remediate the Spyre configuration settings

The `servicereport` command is a powerful, plugin-driven tool used on IBM Power systems to validate and optionally auto-remediate system configurations.

- a. Check your system with this command:

```
servicereport -v -p spyre
```

After a fresh system installation, most checks might show the status as FAIL. as shown in Example 7-12.

*Example 7-12 servicereport command for spyre*

---

```
root # servicereport -v -p spyre
servicereport 2.2.5

Spyre configuration checks          FAIL
VFIO Driver configuration          FAIL
User memlock configuration         FAIL
sos config                         FAIL
sos package                        PASS
VFIO udev rules configuration      FAIL
User group configuration           PASS
VFIO device permission            FAIL
VFIO kernel module loaded         FAIL
VFIO module dep configuration     FAIL
```

---

b. Attempt automated remediation for identified issues

Fix identified issues with:

```
servicereport -r -p spyre
```

This command sets up the following functions:

- Virtual Function I/O (VFIO) device bindings for the VFIO kernel module.
- Configures VFIO udev rules.
- Adds the sentient user group.
- Sets memory limits for the sentient group.
- Inserts the VFIO modules.

Example 7-13 shows a sample output.

*Example 7-13 Results from automatic remediation*

---

```
root # servicereport -I -p spyre
servicereport 2.2.5

Spyre configuration checks          PASS

VFIO Driver configuration          PASS      Auto Fixed
User memlock configuration         PASS      Auto Fixed
sos config                         PASS      Auto Fixed
sos package                        PASS
VFIO udev rules configuration      PASS      Auto Fixed
User group configuration           PASS
VFIO device permission            PASS      Auto Fixed
VFIO kernel module loaded         PASS      Auto Fixed
VFIO module dep configuration     PASS      Auto Fixed
```

Memlock limit is set for the sentient group. Spyre user must be in the sentient group.

To add run below command:

```
sudo usermod -aG sentient <user>
```

Example:

```
sudo usermod -aG sentient abc
```

```
Re-login as <user>.
```

---

5. Create the model source directory and set collaborative permissions for the Spyre group.

Use this command to set up the model source directory:

```
# sudo install -d -m 0775 -o root -g sentient /opt/ibm/spyre/models/src
```

### 7.4.3 RHAIS and API shape

Red Hat AI Inference Server (RHAIS) provides an enterprise-grade vLLM runtime and exposes an OpenAI-compatible API with the following endpoints:

- /v1/completions
- /v1/chat/completions
- /v1/embeddings

By default, server configurations such as tokenizer settings and generation parameters are derived from the model's `generation_config.json` file unless explicitly overridden.

The vLLM runtime delivers high-performance inference through advanced features, including:

- ▶ Continuous batching for efficient request handling
- ▶ KV-cache management for optimized memory usage
- ▶ Dynamic scheduling for improved throughput

When Spyre acceleration is available and the Spyre backend is enabled, vLLM automatically routes critical kernels through Spyre, providing significant improvements in throughput and latency for large language model workloads.

### 7.4.4 Obtaining and managing models

Models such as IBM Granite can be sourced from Hugging Face or IBM repositories. For repositories that use Git Large File Storage (LFS), complete the following steps to initialize LFS and clone the required models into the host models directory:

1. Enable Git LFS

Use this command to install LFS from git:

```
# git lfs install
```

2. Navigate to the host models source directory

Navigate to the host model source directory created previously:

```
# cd /opt/ibm/spyre/models/src
```

3. Clone your model

Location for some of the supported models is shown in Example 7-14.

*Example 7-14 Cloning models*

---

IBM Granite instruction LLM

```
#git clone https://huggingface.co/ibm-granite/granite-3.3-8b-instruct
```

Granite embedding models (English and Multilingual)

```
#git clone https://huggingface.co/ibm-granite/granite-embedding-30m-english
```

```
#git clone https://huggingface.co/ibm-granite/granite-embedding-125m-english
```

```
#git clone https://huggingface.co/ibm-granite/granite-embedding-107m-multilingual
```

```
#git clone https://huggingface.co/ibm-granite/granite-embedding-278m-multilingual
```

Reranker  
#git clone https://huggingface.co/BAAI/bge-reranker-v2-m3

---

**Attention:** Verify licenses, model cards, and internal approval before using external models in production. For supported models and patterns, consult IBM Spyre for Power documentation and internal catalogs.

Place the selected model paths under a host-visible models directory and mount into containers as needed.

## 7.4.5 Environment variables and tuning

Set environment variables to configure Spyre acceleration by specifying hardware resources and model paths. For example, define `AIU_IDS` for Spyre PCIe devices, `AIU_WORLD_SIZE` for the number of accelerator cores, and `VLLM_SPYRE_USE_CB=1` to enable Spyre-specific code paths. Table 7-2 describes important environment variables.

Table 7-2 Environment variables

Variable	Descriptor
<code>AIU_IDS</code>	PCIe IDs of Spyre accelerators to use (space-separated).
<code>AIU_WORLD_SIZE</code>	Number of Spyre accelerator cores/cards assigned to the container.
<code>HOST_MODELS_DIR</code>	Host directory containing models (mounted as <code>/models</code> ).
<code>VLLM_MODEL_PATH</code>	Model path inside the container (e.g., <code>/models/granite-3.3-8b-instruct</code> ).
<code>MAX_MODEL_LEN</code>	Max context length ( <code>prompt_tokens + output_tokens</code> ).
<code>MAX_BATCH_SIZE</code>	Server-side max concurrent sequences.
<code>PROMPT_LEN</code> , <code>BATCH_SIZES</code>	Warmup shapes for embeddings/reranker pipelines.
<code>VLLM_SPYRE_USE_CB=1</code>	Enable Spyre code paths (specific to Spyre-enabled builds).

### Memory guidance

For large models, set `--memory` and `--shm-size` conservatively (e.g., `--memory 200G`, `--shm-size 64G`) and pin `AIU_WORLD_SIZE` to actual hardware.

## 7.4.6 Running RHAIS with Podman (examples)

This section details Podman execution patterns for entity extraction, RAG, reranking, and embedding workloads. Modify PCIe device identifiers, filesystem paths, and memory or buffer size parameters to align with your system topology and workload specific requirements.

Example 7-15 shows the example using entity extraction. This uses one Spyre device.

*Example 7-15 Entity extraction*

---

```
#export AIU_IDS="0381:50:00.0"  
#export AIU_WORLD_SIZE=1
```

```

#export HOST_MODELS_DIR=/models
#export VLLM_MODEL_PATH=/models/granite-3.3-8b-instruct
# export MAX_MODEL_LEN=3072
#export MAX_BATCH_SIZE=16
# podman run -d \
    --device=/dev/vfio \
    -v ${HOST_MODELS_DIR}:/models \
    -e AIU_PCIE_IDS="${AIU_IDS}" \
    -e VLLM_SPYRE_USE_CB=1 \
    --pids-limit 0 \
    --users=keep-id \
    --group-add=keep-groups \
    --security-opt label=disable \
    --memory 200G \
    --shm-size 64G \
    -p 127.0.0.1:8000:8000 \
    <container url>:<container tag> \
    --model "${VLLM_MODEL_PATH}" \
    -tp "${AIU_WORLD_SIZE}" \
    --max-model-len "${MAX_MODEL_LEN}" \
    --max-num-seqs ${MAX_BATCH_SIZE}

```

---

Ensure to fill in the environment variables as per your system environment.

Example 7-16 shows an example for Retrieval Augmented Generation which uses four Spyre devices. Be sure to fill in the variables according to your system configuration.

*Example 7-16 Retrieval Augmented Generation*

---

```

# export AIU_IDS="0381:50:00.0 0382:40:00.0 0383:30:00.0 0384:20:00.0"
#export AIU_WORLD_SIZE=4
#export HOST_MODELS_DIR=/models
#export VLLM_MODEL_PATH=/models/granite-3.3-8b-instruct
#export MAX_MODEL_LEN=32768
#export MAX_BATCH_SIZE=32
#podman run -d \
    --device=/dev/vfio \
    -v ${HOST_MODELS_DIR}:/models \
    -e AIU_PCIE_IDS="${AIU_IDS}" \
    -e VLLM_SPYRE_USE_CB=1 \
    --users=keep-id \
    --group-add=keep-groups \
    --security-opt label=disable \
    --pids-limit 0 \
    --memory 200G \
    --shm-size 64G \
    -p 127.0.0.1:8000:8000 \
    <container url>:<container tag>\
    --model "${VLLM_MODEL_PATH}" \
    -tp "${AIU_WORLD_SIZE}" \
    --max-model-len "${MAX_MODEL_LEN}" \
    --max-num-seqs ${MAX_BATCH_SIZE}

```

---

Example 7-17 shows the settings to run reranker. This example uses one Spyre device. Adapt the variables to match your environment.

*Example 7-17 Example for running reranker*

---

```
export AIU_IDS="0381:50:00.0"
#export AIU_WORLD_SIZE=1
#export HOST_MODELS_DIR=/models
#export MODEL_PATH=/models/bge-reranker-v2-m3
#export PROMPT_LENS=512
#export BATCH_SIZES=4
#podman run -d \
    --device=/dev/vfio \
    -v ${HOST_MODELS_DIR}:/models \
    -e AIU_PCIE_IDS="${AIU_IDS}" \
    -e VLLM_SPYRE_WARMUP_BATCH_SIZES="${BATCH_SIZES}" \
    -e VLLM_SPYRE_WARMUP_PROMPT_LENS="${PROMPT_LENS}" \
    --users=keep-id \
    --group-add=keep-groups \
    --security-opt label=disable \
    --pids-limit 0 \
    --memory 200G \
    --shm-size 64G \
    -p 127.0.0.1:8000:8000 \
    <container url>:<container tag> \
    --model "${MODEL_PATH}" \
    -tp "${AIU_WORLD_SIZE}"
```

---

The final example is shown in Example 7-18 and shows the setup for embedding workflows. Again, remember to adjust the variables to meet your system configuration.

*Example 7-18 Embeddings*

---

```
#export AIU_IDS="0381:50:00.0"
#export AIU_WORLD_SIZE=1
#export HOST_MODELS_DIR=/models
#export MODEL_PATH=/models/granite-embedding-125m-english
#export PROMPT_LENS=512
#export BATCH_SIZES=4
#podman run -d \
    --device=/dev/vfio \
    -v ${HOST_MODELS_DIR}:/models \
    -e AIU_PCIE_IDS="${AIU_IDS}" \
    -e VLLM_SPYRE_WARMUP_BATCH_SIZES="${BATCH_SIZES}" \
    -e VLLM_SPYRE_WARMUP_PROMPT_LENS="${PROMPT_LENS}" \
    --users=keep-id \
    --group-add=keep-groups \
    --security-opt label=disable \
    --pids-limit 0 \
    --memory 200G \
    --shm-size 64G \
    -p 127.0.0.1:8000:8000 \
    <container url>:<container tag> \
    --model "${MODEL_PATH}" \
    -tp "${AIU_WORLD_SIZE}"
```

---

## 7.4.7 Validating the server

To verify that the inference server is fully operational and that the vLLM runtime has initialized correctly, perform a series of functional and container-level checks. These include confirming container health and reviewing warm-up logs, followed by issuing simple completion and embeddings requests to validate model loading, request routing, and end-to-end inference behavior. The following steps provide a minimal but comprehensive validation workflow.

1. Check container status and warm-up logs:
  - a. List containers
  - b. Tail logs to watch vLLM warm-up and readiness

```
#podman ps
```

```
#podman logs <container_id>
```

2. Test a simple completion call as shown in Example 7-19.

*Example 7-19 Simple completion test*

---

```
#curl -X POST -H "Content-Type: application/json" \
-d '{
    "model": "/models/granite-3.3-8b-instruct",
    "prompt": "What is the capital of France?",
    "max_tokens": 32
}' \
http://127.0.0.1:8000/v1/completions
```

---

3. Test an embeddings call as shown in Example 7-20.

*Example 7-20 Test embeddings call*

---

```
#curl -X POST -H "Content-Type: application/json" \
-d '{
    "model": "/models/granite-embedding-125m-english/",
    "input": ["What is the capital of Paris?"]
}' \
http://127.0.0.1:8000/v1/embeddings
```

---

## 7.4.8 Troubleshooting and best practices

When deploying AI workloads with Podman on systems equipped with IBM Spyre accelerators, several classes of configuration and runtime issues may surface. These commonly involve registry authentication, device binding and VFIO access, model path and permission alignment, memory constraints, multi-card scaling parameters, and SELinux interactions. The following troubleshooting notes summarize the primary failure modes and the corrective actions required to ensure stable operation across entity-extraction, RAG, reranking, and embedding workloads.

- ▶ Registry authentication fails

Ensure podman login registry.redhat.io uses a valid Red Hat account with entitlements to pull RHAIS images.

- ▶ VFIO / device access errors
 

Confirm Spyre devices are bound to VFIO and visible to the container. Use `lspci -vv` to verify link speed and slot assignments.

Run `servicereport -v -p spyre`, remediate with `-r` if necessary.
- ▶ Model not found / permission denied
 

Verify host directory permissions (`/models` or `/opt/ibm/spyre/models/src`) and group membership (`sentient`).

Ensure `-v ${HOST_MODELS_DIR}:/models` is present and paths inside the container match `VLLM_MODEL_PATH`.
- ▶ Insufficient memory / OOM
 

Increase `--memory` and `--shm-size` according to model size and batch targets.

Reduce `MAX_BATCH_SIZE` or context length (`MAX_MODEL_LEN`) for stability.
- ▶ Multi-card scaling not effective
 

Check `AIU_PCIE_IDS` enumerations and `AIU_WORLD_SIZE`.

Confirm firmware/driver versions are current and cards are in supported drawer slots.
- ▶ SELINUX
 

Verify that the SELinux settings align correctly with Podman's `--privileged` option

## 7.5 Other tools

There exists a bewildering variety of tools and frameworks in support of application development. Some may be of transient usage, while other will endure as standard design patterns. A brief summary of some of these projects follows. The list is representative, not exhaustive, and a project's inclusion and the order in which they are listed is not intended to express preference nor even recommendation on part of the authors of this Redbook.

- ▶ Code assistants
  - IBM offerings
    - Watsonx Code Assistant for z
    - IBM i Project Bob
  - Cline
  - Roo Code
  - Kilo
  - Continue
  - Github Copilot
- ▶ Agent generators
  - Langchain
  - Cline
- ▶ Model Context Protocol (MCP)
  - FastMCP
  - PHP MCP
- ▶ A2A Agent to Agent Protocol

- ▶ Vector databases on Power architecture
  - MongoDB
  - Milvus
  - Streamlit

An exemplary demo of RAG on Power is documented at [Boost the precision of your RAG workflow](#) by IBMer Henrik Mader In his example Mr. Mader loads PDF files into an AI workflow to allow intelligent querying of the PDF archive.

## 7.6 Using the Wallaroo AI Starter Kit

[The Wallaroo AI Starter Kit for IBM](#) offers fully installable pre-built AI API, an AI Model, an AI runtime optimized on IBM Power, and an Inference Server on a variety of Power Linux platforms. These features are fully integrated with and optimized for Power 11 hardware including the Spyre card, if present.

Rather than rely on the customary administrative practices associated with maintaining a Linux installation, the Starter Kit installation is maintained by periodic updates from Wallaroo and is effectively a "black box" to the administrator.

# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

## IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- ▶ *IBM Power E1180: Introduction and Overview*, SG24-8587
- ▶ *IBM Power11 E1150 Introduction and Technical Overview*, SG24-8589
- ▶ *IBM Power11 Scale-Out Servers: Introduction and Overview*, SG24-8590
- ▶ *IBM Power System AC922 Technical Overview and Introduction*, REDP-5494
- ▶ *Matrix-Multiply Assist Best Practices Guide*, REDP-5612

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

[ibm.com/redbooks](https://ibm.com/redbooks)

## Online resources

These websites are also relevant as further information sources:

- ▶ IBM Spyre Enablement Stack for Power  
<https://www.ibm.com/docs/en/ibm-spyre-for-power>
- ▶ IBM Open-Source AI foundation for Power  
<https://www.ibm.com/docs/en/aiservices>
- ▶ Power ISA v3.1  
[https://wiki.raptorcs.com/w/images/f/f5/PowerISA\\_public.v3.1.pdf](https://wiki.raptorcs.com/w/images/f/f5/PowerISA_public.v3.1.pdf)
- ▶ A matrix math facility for Power ISA processors  
<https://arxiv.org/pdf/2104.03142>
- ▶ Dense Dynamic Blocks: Optimizing SpMM for Processors with Vector and Matrix Units Using Machine Learning Techniques  
[https://iacoma.cs.uiuc.edu/iacoma-papers/ics22\\_2.pdf](https://iacoma.cs.uiuc.edu/iacoma-papers/ics22_2.pdf)

## Help from IBM

IBM Support and downloads

[ibm.com/support](https://ibm.com/support)

IBM Global Services

[ibm.com/services](https://ibm.com/services)



**Redbooks**

# Implementing AI on Power11: Introducing the IBM Spyre

SG24-8592-00

ISBN



(1.5" spine)  
1.5" <-> 1.998"  
789 <-> 1051 pages



**Redbooks**

# Implementing AI on Power11: Introducing the IBM Spyre Adapter

SG24-8592-00

ISBN



(1.0" spine)  
0.875" <-> 1.498"  
460 <-> 788 pages

**Redbooks**

## Implementing AI on Power11

SG24-8592-00

ISBN



(0.5" spine)  
0.475" <-> 0.873"  
250 <-> 459 pages

**Redbooks**

## Implementing AI on Power11

(0.2" spine)

0.17" <-> 0.473"  
90 <-> 249 pages

(0.1" spine)  
0.1" <-> 0.169"  
53 <-> 89 pages



# Implementing AI on Power11:

SG24-8592-00

ISBN

(2.5" spine)  
2.5" <-> nnn.n"  
1315 <-> nnnn pages



# Implementing AI on Power11: Introducing the IBM Spyre Adapter

SG24-8592-00

ISBN

(2.0" spine)  
2.0" <-> 2.498"  
1052 <-> 1314 pages







SG24-8592-00

ISBN

Printed in U.S.A.

Get connected

