Redbooks

ibm.com/redbooks

Enriching Linux on IBM Z Workloads with AI

Sambasiva Andaluri

Raymond Chiang

Daniele Comi

Gayathri Gopalakrishnan

Andreas Krebbel

Karen Medhat

Padma Mohapatra

Dominic Roehm

IBM LinuxONE

IBM Z

IBM

Redpaper

IBM Redbooks

Enriching Linux on IBM Z Workloads with AI

June 2024

REDP-5712-00

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

First Edition (June 2024)

This edition applies to Version 3, Release 1, of IBM z/OS.

This document was created or updated on June 20, 2024.

Contents

Notices vii
Trademarks ix

Preface xi
Authors xi
Now you can become a published author, too! xiii
Comments welcome xiii
Stay connected to IBM Redbooks xiv

Chapter 1. Introduction to AI and machine learning 1
1.1 Understanding the basics of AI and ML 2
1.2 Why AI on IBM Z 3
1.2.1 Introduction to data gravity and governance 3
1.2.2 Overview of Linux on IBM Z 4
1.2.3 Relevance of the Linux ecosystem of IBM Z for AI 4

Chapter 2. Getting started with AI on Linux for IBM Z 7
2.1 Setting up your IBM Z system 8
2.1.1 Hardware configuration 8
2.1.2 IBM z16 Integrated Accelerator for AI software configuration 10
2.1.3 IBM Z processor features 12
2.1.4 AI software 16

Chapter 3. The power of Linux on IBM Z for AI workloads 17
3.1 Train anywhere, deploy on IBM Z 18
3.2 Scalability and performance optimization 18
3.2.1 Ongoing analysis to optimize performance 19
3.2.2 Making intelligent decisions around capacity management 20
3.2.3 Optimizing for cost 20
3.3 Addressing security and compliance concerns 20
3.3.1 Security concerns 21
3.3.2 Compliance concerns 21
3.4 Technical advancements in data management and storage 22
3.4.1 Advantages of IBM Z in data management and storage 22
3.4.2 Data security and compliance 23
3.4.3 Scalability and performance 24
3.5 A toolkit for success 24
3.5.1 The IBM Z advantage 24
3.5.2 Real-world applications 25
3.6 AI open-source ecosystem on IBM Z 26
3.6.1 How to install open-source AI packages 26
3.6.2 How to get started with open-source AI frameworks 28
3.6.3 Endianness in saved AI models 29
3.7 Packages with IBM Z optimizations 31

Chapter 4. AI case studies for Linux workloads on IBM Z 33
4.1 AI case studies 34
4.1.1 Supervised learning 34
4.1.2 Unsupervised learning 35

4.1.3 Reinforcement learning . . . 35
4.1.4 Natural language processing . . . 37
4.1.5 Computer vision . . . 38
4.1.6 Time series analysis . . . 39
4.2 Workloads processing and security . . . 41
4.2.1 Batch processing with short batch window . . . 41
4.2.2 Security . . . 41
4.2.3 Data gravity . . . 41
4.2.4 AI Toolkit for IBM Z . . . 42
4.3 Success story . . . 42

Chapter 5. AI Model inferencing on IBM Z . . . 43
5.1 Performance and environmental benefits of co-locating AI inferencing on IBM Z with OLTP 44
5.2 Use case 1: Core count requirement for IBM Z in-transaction inferencing versus x86 . 45
5.2.1 Use case 2: IBM Z in-transaction inferencing response time versus remote inferencing in x86 + GPU . . . 47
5.2.2 Use case 3: Energy consumption of IBM Z in-transaction inferencing versus remote inferencing in x86 + GPU . . . 48
5.2.3 Use case 4: Comparing Random Forest classifier of IBM Z in-transaction inferencing response time versus remote inferencing in x86 + GPU . . . 49
5.3 The technologies behind the high performance of IBM Z . . . 50
5.3.1 Single-instruction, multiple-data on IBM z16 Telum processor . . . 51
5.3.2 IBM Z Deep Learning Compiler . . . 52
5.3.3 IBM-zDNN-plug-in to TensorFlow . . . 53
5.3.4 IBM Snap ML . . . 54
5.4 Leveraging the IBM z16 Integrated Accelerator for AI in model inferencing server . . . 55
5.5 Requirements of a model inference server . . . 57
5.5.1 AI Toolkit for IBM Z and IBM LinuxONE . . . 57
5.6 Use case implementations . . . 60
5.6.1 Implementing use cases 1 to 3 as in-transaction inferencing . . . 60
5.6.2 Implementing inferencing in a separate container . . . 60
5.7 Performance monitoring and tuning . . . 61
5.8 Monitor IBM z16 Integrated Accelerator for AI usage . . . 63
5.8.1 CPU Measurement Facility . . . 63
5.8.2 Processor Activity Instrumentation Facility . . . 65
5.9 Tuning . . . 67
5.9.1 Thread-pinning to utilize multiple AI accelerators . . . 67
5.9.2 Transparent Huge Pages . . . 68
5.9.3 Touch memory of the output tensors . . . 68
5.9.4 Improving IBM z16 Integrated Accelerator for AI utilization . . . 68
5.10 Next steps . . . 69
5.10.1 AI on IBM Z and IBM LinuxONE Discovery Workshop . . . 69

Chapter 6. Security and privacy . . . 71
6.1 Security . . . 72
6.1.1 System security . . . 72
6.1.2 Workload security . . . 73
6.2 Privacy . . . 75
6.2.1 Compliance and regulation . . . 75
6.2.2 Data privacy . . . 76
6.2.3 Conclusion . . . 76

Chapter 7. Future trends . . . 79

7.1 Exploring emerging AI technologies for Linux 80
7.2 Predicting the future of AI in Linux workloads 80
7.2.1 watsonx future applications 80
7.2.2 Homomorphic encrypted AI 83

Related publications 85
IBM Redbooks 85
Online resources 85
Help from IBM 85

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 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 Watson®	Redbooks (logo) ®
DB2®	IBM Z®	WebSphere®
Db2®	IBM z13®	z/Architecture®
IBM®	IBM z14®	z/OS®
IBM Cloud®	IBM z16™	z/VM®
IBM Cloud Pak®	Netezza®	z13®
IBM FlashSystem®	Parallel Sysplex®	z15®
IBM Research®	PowerPC®	z16™
IBM Telum®	Redbooks®	

The following terms are trademarks of other companies:

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.

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

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

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

Preface

In the rapidly evolving landscape of technology, the symbiosis between traditional computing platforms and innovative AI capabilities has become not just a trend but a necessity. As organizations and individuals seek to harness the power of AI for workload optimization, the marriage of Linux and AI emerges as a compelling frontier.

This IBM Redpaper publication discusses the technical intricacies of artificial intelligence (AI) and machine learning (ML) within the robust IBM Z ecosystem, exploring the synergy between Linux-based systems and the transformative potential of AI. It extends into the technical intricacies of integrating AI-enhanced workloads, shedding light on security concerns, and projecting the transformative impact of AI across industries.

This exploration encompasses fundamental technical principles, addressing critical aspects of setup, including hardware configurations, software prerequisites, and the nuances of Linux deployment. Readers gain insights into the technical advantages inherent in leveraging Linux on IBM Z, particularly with the advancements introduced by the IBM z16™. Case studies dissecting various AI applications in Linux workloads on IBM Z, spanning supervised and unsupervised learning, natural language processing, and time series analysis will be discussed. Nuanced technical discussions cover performance metrics, security considerations, and future trajectories associated with the IBM Telum® processor, a linchpin in AI and ML hardware innovation, are also included.

As we delve into automation, ML, and deep learning, the fusion of these powerful domains unfolds as a catalyst for innovation and efficiency. This publication enables you to leverage the convergence of Linux and AI. Whether you seek to optimize workloads, build intelligent systems, or simply explore the limitless possibilities at the intersection of Linux workloads and AI, this publication endeavors to be your companion on this innovative journey.

This IBM Redpaper publication is intended for data scientists, Chief Data Officers (CDOs), and infrastructure personnel looking to leverage existing AI models in a Linux on IBM Z® environment without requiring deep knowledge in IBM Z. Take advantage of the IBM z16™ Integrated Accelerator for AI for models running on Z to gain insights from data in place while remaining secure and compliant while simplifying the AI software installation pipeline to keep your AI software stack up-to-date with the latest versions and capabilities.

Authors

This paper was produced by a team of specialists from around the world.

Sambasiva Andaluri is a Principal IBM Cloud® HPC Architect in the USA. He has 30 years of experience in software development, SaaS, and architecture field. He has worked at IBM for 2 years. His areas of expertise include C/C++/Java/Python development, SaaS, multi-cloud, performance engineering, large scale distributed systems for AI and ML and HPC. He has written extensively on Power, Storage, multi-cloud SaaS architectures at IBM and with an external publisher.

Raymond Chiang is a Hybrid Cloud Architect in IBM Canada. He specializes in IBM WebSphere® and Java EE application design, implementations, and system integrations, especially on the IBM Z platform. His other areas of expertise include ML, IBM z/OS® Connect, Kubernetes, and Red Hat OpenShift Container Platform.

He is a co-author of the following IBM® Redbooks® publications, *IBM System z in a Mobile World: Providing Secure and Timely Mobile Access to the Mainframe*, SG24-8215, and *SOA Transition Scenarios for the IBM z/OS Platform*, SG24-7331.

Daniele Comi is a Data Scientist and Software Engineer in IBM Italy. He has 2 years of experience in data analytics, ML, and deep learning. He has worked at IBM for 2 years. His area of expertise include machine/reinforcement/deep learning from the architectural-level to the more scientific-level, and of the various AI frameworks, obtained through his MSc in Computer Science Engineering. He has a profound knowledge of Linux with a course certification on AI on IBM Z. He is part of the newly formed Generative AI team in Italy. Daniele has written extensively on cutting-edge deep learning techniques and encoder-decoder generative models papers through a collaboration with IBM Research®.

Gayathri Gopalakrishnan works for IBM India and has over 23 years of experience as a technical solution and IT Architect, working primarily in consulting. She is a results-driven IT Architect with extensive experience in spearheading the management, design, development, implementation, and testing of solutions. A recognized leader, applying high-impact technical solutions to major business objectives with capabilities transcending boundaries. Adept at working with management to prioritize activities and achieve defined project objectives, ability to translate business requirements into technical solutions.

Andreas Krebbel works for IBM and has over 20 years of experience with Linux core components like compilers and run-time libraries. As the open-source maintainer of GCC, Binutils, and the GNU C Library he implemented the exploitation and optimization for 8 IBM Z CPU generations. In his role as Architect for AI and Toolchain on Linux on IBM Z, Andreas is responsible for the design and development of a wide range of core components of the Linux on IBM Z operating system, which include compilers, AI frameworks, core libraries, and performance tooling.

Karen Medhat is a Customer Success Manager Architect in the UKI and the youngest IBM Certified Thought Leader Level 3 Technical Specialist and Open Group Distinguished Technical Specialist. She is the chair of the IBM UK Technical Consultancy Group and a leader in IBM Open Innovation community. She is a senior leader for IBM TechXchange in the UKI, as well as an EMEA leader for automation product scouts. She holds an MSc degree with honors in Engineering in AI and Wireless Sensor Networks from the Faculty of Engineering, Cairo University, and a BSc degree with honors in Engineering from the same faculty. She co-creates curriculum and exams for different IBM professional certificates and has also created and co-created courses for IBM Skills Academy in various areas of IBM technologies. She serves on the review board of international conferences and journals in AI and wireless communication. She is also an IBM inventor and experienced in creating applications architecture and leading teams of different scales to deliver customers' projects successfully. She frequently mentors IT professionals to help them define their career goals, learn new technical skills, or acquire professional certifications. She has authored publications on Cloud, IoT, AI, wireless networks, microservices architecture, and blockchain.

Padma Mohapatra works for IBM India and has over 20 years of experience as an application architect, working primarily in mainframe modernization on the IBM Z systems. He has been with IBM for 15 years and has been in the architect profession for almost 5 years. He has extensive experience in analysis, architecture, design, project development, testing, and support of mainframe applications. He has experience in cloud migration and modernization of mainframe systems and is actively working on patenting.

Dominic Roehm works for IBM Research & Development in Germany and is the Chief Performance Architect for Linux on Z in the global Infrastructure Performance and Architecture organization. He holds a PhD. in Physics from the University of Stuttgart, Germany.

His expertise includes high performance computing, GPU computing as well as special hardware accelerators for AI frameworks. His responsibilities range from competitive performance benchmarking to full stack performance optimizations and the development of prototypes or new back ends for AI frameworks. Furthermore, he has been leading the Linux performance claim work for several IBM Z and IBM LinuxONE generations.

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

Robert Haimowitz and Makenzie Manna
IBM Redbooks, Poughkeepsie Center

Artem Minin and Andrew Sica
IBM US

Marcus Kraft
IBM Germany

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`

Comments welcome

Your comments are important to us!

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

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

  `ibm.com/redbooks`

- Send your comments in an email to:

  `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

1

Introduction to AI and machine learning

In this chapter, we will start giving an initial cover to the fascinating world of artificial intelligence (AI) and machine learning (ML) while shedding light on their basic principles. We delve into why AI on IBM Z is a strategic necessity in today's data-driven landscape. Exploring the significance of data gravity and governance, we introduce the robust IBM Z ecosystem and emphasize the profound impact AI can have on its Linux workloads. Our aim is to equip you with the essential knowledge required to appreciate the symbiotic relationship between AI, Linux, and IBM Z, setting the stage for the deeper explorations that follow in this publication.

1.1 Understanding the basics of AI and ML

AI and ML are transformative technologies that have reshaped the modern tech landscape. At their core, AI and ML are about enabling computers to mimic human intelligence and make data-driven decisions. AI generally refers to the concept of computers performing tasks that typically require human intelligence to solve complex problems. Machine learning is a subset of AI that focuses on algorithms that allow machines to learn from data and improve their performance over time. It is crucial to distinguish between AI and ML, depending on the task in front of you.

The fundamental components of AI and ML include: data, algorithms, and computing power.

Data forms the foundational building blocks that empower these technologies to reach new heights. It is not just a passive component but rather the dynamic fuel that propels AI and ML forward. High-quality, diverse datasets are indispensable for the success of AI and ML applications.

Through data, models learn to recognize patterns, adapt to changing circumstances, and make predictions that extend far beyond the capabilities of traditional computing systems. This process allows us to develop and refine the highly sophisticated models we have at our disposal today. These models aim to approximate ideal functions that capture the essence of the tasks they are designed to tackle, whether it's natural language understanding, image recognition, or complex decision-making processes.

While data is essential to AI and ML, algorithms, the mathematical engines that process and analyze data, are necessary to extract insights. Machine learning algorithms take various forms, each tailored to specific problem-solving scenarios.

- Supervised learning algorithms learn from labeled data, effectively taking the guidance of known outcomes to discern patterns and predict future results.
- Unsupervised learning algorithms operate without the luxury of labeled data, exploring data structures to identify inherent relationships and patterns.
- Alternatively, reinforcement learning algorithms master the art of decision-making through repeated trial and error, constantly adapting their strategies based on feedback.

These algorithms bridge the gap between raw information and actionable intelligence, enabling AI and ML systems to perform tasks that are often indistinguishable from those executed by humans.

The journey to harness the full potential of AI and ML has its challenges. Some of these algorithms require substantial computational power to chew through the vast datasets and complex mathematical operations needed to uncover valuable insights. This is where high-performance computing environments like IBM Z and its dedicated AI acceleration components come into play. These specialized systems provide the computational muscle required to tackle the specific AI and ML on Z workloads, unlocking the doors to previously unattainable achievements on IBM Z in areas such as natural language understanding, image recognition, financial forecasting, and autonomous decision-making.

Data and algorithms are the twin pillars of AI and ML. The synergy between them defines the capabilities and potential of these technologies. With the appropriate data and the correct algorithms running on high-performance computing platforms like IBM Z, we have the power to push the boundaries of what AI and ML can achieve.

1.2 Why AI on IBM Z

In the world of enterprise computing, the IBM Z mainframe is synonymous with unwavering reliability and security. With AI becoming more common, the question arises: Why AI on IBM Z? The answer lies in the commitment by IBM to deliver high value and near-zero risk with a focus on high performance. This translates into low-latency AI processing for real-time analytics, such as credit card fraud detection. The engineering prowess of IBM gave rise to the IBM z16 hardware and software, delivering massive scalability and enabling millions or even billions of AI inferences daily. IBM Z empowers organizations to seamlessly integrate AI-powered insights into their transactional applications and operational processes. It facilitates ML and deep learning alongside transactional applications, delivering real-time insights, improving decision-making, and enhancing customer experiences.

All this is possible thanks to the introduction of the IBM Telum Processor, a pivotal innovation designed to deliver in-transaction inference in real-time and at scale. Telum is equipped with a dedicated on-chip accelerator for AI inference, ensuring low-latency AI embedded directly in transactional workloads. It also features enhanced performance, security, and availability.

Recently announced, the AI Toolkit for IBM Z further improves the adaptability of AI on IBM Z, another why AI and IBM Z are a natural fit.

IBM Z offers the ideal platform for businesses looking to embrace AI without compromising on security, performance, or reliability. AI on IBM Z is a necessary step for organizations aiming to stay ahead in a competitive business environment.

1.2.1 Introduction to data gravity and governance

The concept of data gravity casts a profound shadow over the digital landscape, underscoring the notion that data possesses its own mass, an invisible but substantial weight that grows with every byte, record, and piece of information gathered and stored. This digital gravity, similar to the gravitational pull of celestial bodies, exerts a powerful influence on how organizations interact with, harness, and leverage their data resources.

In today's data-driven world, where organizations of all sizes accumulate vast troves of information, the implications of data gravity are far-reaching. The more data an organization accumulates, the more challenging it becomes to manage, analyze, and extract valuable insights from that ever-expanding universe of information. This is true for enterprises handling substantial volumes of sensitive data, such as financial records, healthcare information, or personal customer details. As data accumulates, its gravitational pull becomes palpable, causing issues related to data accessibility, data silos, and the capacity to derive actionable intelligence.

In response to these challenges, IBM Z emerges as a stalwart sentinel in the digital realm, uniquely positioned to address the complexities of data gravity. The robust data security and governance capabilities of IBM Z serve as a beacon of assurance in an increasingly uncertain data landscape.

At its core, IBM Z is synonymous with trust, reliability, and security. It is a platform that has stood the test of time, guarding sensitive data for decades across diverse industries. For organizations concerned about data privacy, security breaches, and compliance with ever-stringent regulatory requirements, IBM Z is a bastion of protection.

Within the heart of IBM Z, a powerful suite of data governance capabilities is at work, ready to safeguard the integrity and confidentiality of data in motion and at rest.

Advanced encryption mechanisms provide an impenetrable shield for sensitive information, ensuring that even during a breach, the data remains inscrutable to unauthorized entities. Secure multi-tenancy functionality enables organizations to run diverse workloads on the same infrastructure without compromising the security boundaries that keep them isolated and protected. And in the secure enclave of trusted execution environments, the most sensitive operations can take place, immune to external interference.

By embarking on the journey to AI on IBM Z, organizations can tap into the full potential of their data assets while navigating the complex terrain of data gravity with confidence. The synergy between AI and the formidable security infrastructure of IBM Z empowers enterprises to break free from the gravitational pull of data complexity. They can analyze data with unprecedented precision, revealing hidden insights, automating critical decisions, and driving innovation with data as the catalyst.

Furthermore, the assurance of compliance with stringent regulations, such as the General Data Protection Regulation (GDPR), the Health Insurance Portability and Accountability Act (HIPAA), or financial industry mandates, remains unwavering, even in the face of ever-evolving legal requirements. With IBM Z, organizations are not only equipped to address today's data gravity but also to confront the challenges of tomorrow, forging ahead in a data-centric world where security, governance, and intelligence converge to illuminate new horizons. Taking as example the recently discussed European Union Artificial Intelligence (EU AI) Act, with IBM Z the various organizations harnessing it can be already compliant. IBM not only has as its core values a strict implementation of the EU AI Act, it also encourages them. For more information, see IBM statement on EU AI Act vote in European Parliament.

1.2.2 Overview of Linux on IBM Z

Linux has established itself as a dominant operating system in the modern data center, and its compatibility with IBM Z architecture further extends its capabilities. Linux on IBM Z provides a highly reliable, scalable, and secure environment for running critical workloads. With the ability to run multiple Linux instances on a single IBM Z server, organizations can consolidate their infrastructure, reducing costs and improving resource utilization. Today, Linux has become the natural habitat of AI applications, providing the best possible environments in terms of framework interoperability and AI performance.

AI workloads benefit from the robustness of Linux on IBM Z, taking advantage of its high availability, scalability, and exceptional performance. The seamless integration of AI into Linux environments on IBM Z creates a powerful platform for running AI and ML applications at scale.

1.2.3 Relevance of the Linux ecosystem of IBM Z for AI

In the ever-evolving landscape of AI, the demand for robust and scalable computing environments has never been higher. As organizations strive to extract valuable insights from vast amounts of data, the role of Linux on mainframes, particularly
IBM Z, has become increasingly relevant in the AI ecosystem. This chapter explores the symbiotic relationship between Linux and IBM Z, elucidating the advantages that it brings to the field of AI.

Linux as a foundation for AI workloads

Linux has emerged as a predominant operating system for AI applications, owing to its open-source nature, flexibility, and widespread community support. The marriage of Linux with IBM Z creates a potent combination that leverages the strengths of both technologies.

Organizations can harness the power of Linux on IBM Z to run AI workloads efficiently, taking advantage of the mainframe's exceptional processing capabilities.

Security and Reliability

AI applications often deal with sensitive and confidential data. The robust security features of IBM Z, including pervasive encryption and hardware-based isolation, provide a secure foundation for hosting AI workloads. Linux on IBM Z inherits these security attributes, ensuring that AI models and datasets are protected against potential threats. Moreover, the mainframe's reliability ensures uninterrupted processing, a critical factor in AI scenarios where downtime can be costly.

Integration with hybrid cloud environments

As enterprises increasingly adopt hybrid cloud strategies, the synergy between Linux on IBM Z and hybrid cloud environments becomes crucial. The seamless integration of AI workloads across on-premises IBM Z systems and various cloud platforms enhances flexibility, scalability, and resource optimization. This integration facilitates the creation of hybrid AI architectures that balance performance, cost-effectiveness, and data privacy.

Optimization for AI workloads

IBM Z is engineered for high-performance computing, and when paired with Linux, it can be optimized for specific AI workloads. The architecture allows for parallel processing and efficient resource utilization, leading to faster training times and improved inference performance. This optimization is advantageous for AI applications that demand real-time insights, such as those used in financial transactions, healthcare diagnostics, and autonomous systems.

Cost efficiency and resource utilization

Linux on IBM Z offers a cost-effective solution for running AI workloads by efficiently utilizing resources and minimizing operational overhead. The mainframe's virtualization capabilities enable organizations to consolidate workloads, reducing the need for multiple servers. This consolidation not only saves on hardware costs but also streamlines management, making it easier to deploy, monitor, and scale AI applications.

The integration of Linux on IBM Z presents a compelling proposition for organizations seeking a powerful, secure, and efficient platform for AI workloads. The symbiosis of these technologies addresses the challenges of modern AI, providing a foundation that is not only robust and scalable but also capable of meeting the stringent security and reliability requirements of AI applications in diverse industries. As enterprises continue to advance their AI initiatives, the relevance of Linux on IBM Z will undoubtedly play a pivotal role in shaping the future of AI computing.

In this IBM Redpaper publication, we explore how organizations can harness the power of AI on IBM Z to enrich their Linux workloads. We delve into the practical aspects of deploying AI and ML applications, taking advantage of the unique capabilities of IBM Z to ensure the highest levels of data governance, security, and performance.

2

Getting started with AI on Linux for IBM Z

In this chapter, we describe the first step to get started with AI on Linux for IBM Z by exploring the fundamental aspects of setting up your system, understanding hardware and software requirements, installing Linux on IBM Z, and configuring it for AI workloads. As we delve deeper into this exciting domain, we also touch upon the critical processes of data collection and preprocessing.

2.1 Setting up your IBM Z system

Setting up an IBM Z system involves various hardware and software components, and the specifics may vary depending on your exact requirements and the model of the IBM Z system you are working with. There are a few variables involved when setting up an IBM Z system.

2.1.1 Hardware configuration

Relevant hardware configuration variables involved when setting up an IBM Z system include:

- Model

  IBM Z systems come in various models, including the IBM z14®, IM z15, and the newest addition, the IBM z16. With IBM z16 as introduced in 1.2, “Why AI on IBM Z” on page 3 you have access to the newest Telum with IBM z16 Integrated Accelerator for AI. The choice of model depends on your workload and performance requirements. For instance, if you have a high-transaction processing workload, you might choose a model with more cores and memory to handle the workload efficiently. In contrast, if you are running fewer demanding workloads, a lower-end model may suffice. The model also determines the scalability options, such as the ability to add additional processing capacity, as needed.

- Processor type

  IBM Z systems use z/Architecture® processors, which are designed for reliability, security, and high-performance transactional computing. The choice of processor type can significantly impact the system's performance. The processor's clock speed, number of cores, and features like Simultaneous Multithreading (SMT) influence the system's ability to execute instructions in parallel and handle multi-threaded workloads efficiently.

  For AI and data-intensive workloads, selecting a processor with support for specialized vector instructions (for example, Vector Facility) can significantly improve performance. The processor type also affects encryption and security features, making it crucial for secure workloads.

- Memory

  The amount of Random Access Memory (RAM) in your IBM Z system is critical for performance. Memory is used to store active data and code, reducing the need to access data from slower storage devices. The amount of memory should align with the workloads you plan to run.

  AI workloads often involve large datasets and complex model structures, so sufficient memory is essential. More memory allows for better performance with larger workloads because it reduces the need to swap data between memory and disk, which can significantly slow down processing. In-memory processing is essential for real-time and low-latency AI applications, which can be dynamically adapted during operation without any interruption of service or system restarts.

- Storage

  When configuring storage for an IBM Z system, consider the type and amount of storage required. IBM Z systems support various storage types, including Direct Access Storage Devices (DASD), Solid-State Drives (SSD), NVMe (Non-Volatile Memory express) flash driven storage units and tape drives. The choice of storage media depends on your data access speed requirements.

  For AI workloads, especially those involving large datasets, fast storage like IBM FlashSystem® NVMe is crucial to minimize data retrieval times.

IBM FlashSystem uses NVMe-over-fabric that combines fiber channel (zfcp for IBM Z) storage infrastructure management with high transactional, high bandwidth storage, which is even beyond SSD arrays.

For more technical information on the latest version of IBM Storage FlashSystem, see FlashSystem 5000, 5200 and 5300.

You also need to consider the capacity of your storage to ensure you have sufficient space for storing data, models, and logs generated by AI workloads, and for that, larger drives for more static data interaction are available on IBM Z. Proper storage configuration can help avoid I/O bottlenecks and ensure data integrity and availability.

- Networking

  Configuring network interfaces and connectivity for communication with other systems and external networks is fundamental to performance. High-speed network interface cards (NIC) with support for technologies like Remote Direct Memory Access (RDMA) may be essential for such heavy AI workloads.

- Virtualization

  IBM Z systems support multi-level virtualization. You can configure Logical Partitions (LPAR) using IBM Parallel Sysplex® Resource Manager (IBM PR/SM), virtual machines using operating environments like IBM z/VM® and Kernel-based Virtual Machine (KVM), and containers. Virtualization setup involves several technical aspects:

  – LPAR Virtualization

    IBM PR/SM or LPAR provides the first layer of virtualization, allowing the division of a physical IBM Z system into up to 85 logical systems. Each LPAR operates independently, running its own operating system with strict memory separation. This level of virtualization holds an Evaluation Assurance Level 5 (EAL5) certification, helping ensure a high level of security.

    To share data between LPARs, various methods can be employed. Shared disk storage and network connections, such as Open Systems Adapter (OSA) or HiperSockets, facilitate efficient communication and data exchange. Hipersockets are high-performance, low-latency network connections that enable fast communication between LPARs.

    For more information, see HiperSockets.

  – IBM z/VM and KVM Virtualization

    Within an LPAR, a second layer of virtualization can be implemented using IBM z/VM or KVM. This enables the creation of hundreds or even thousands of virtual machines (VM), with the option to share system memory segments. Operating systems running as z/VM or KVM virtual machines benefit from hardware-supported acceleration for virtual memory management. For data sharing between IBM z/VM guests and between KVM guests, additional protocols and mechanisms come into play.

    - Inter-User Communication Vehicle (IUCV) facilitates communication between virtual machines and is particularly useful for sharing data.
    - Shared Memory Communications (SMC) and SMC2 protocols support shared memory communications, enhancing data exchange capabilities among IBM z/VM guests.

- Distributed Console Support Services (DCSS) is a device driver that enables efficient cross-memory data sharing between IBM z/VM guests.
- The Inter-VM Shared Memory device allows KVM guests to share data seamlessly, enhancing collaboration between virtual machine.

For a full list of IBM z/VM protocols and mechanisms, see the IBM z/VM Library.

IBM Z enables you to run different types of scalable workloads securely on the same physical system, tailor and manage your IT infrastructure and business processes with virtualized resources, and process and share data where needed.

2.1.2 IBM z16 Integrated Accelerator for AI software configuration

The chart displayed in Figure 2-1 on page 11 illustrates the seamless integration of AI accelerators into the enterprise AI and ML solution stack. This integration offers exceptional flexibility and interoperability, particularly in the realm of model training and construction. At its core, an AI model represents a computational paradigm designed to emulate intelligent behaviors. The process of running such models is a multifaceted endeavor, involving the execution of intricate computations, the orchestration of data flows, and the strategic mapping of these operations to the underlying hardware architecture, all with the goal of achieving optimal performance.

In the expansive realm of AI frameworks, TensorFlow and PyTorch are leading frameworks for deep learning models. TensorFlow stands out with its direct support for accelerators. This robust framework not only facilitates the efficient training of AI models but also ensures seamless deployment, maximizing the potential of AI accelerators for unparalleled performance. The functionality of PyTorch on IBM Z remains a work in progress concerning accelerator exploitation. The ongoing dedication to enhancing the compatibility of PyTorch with accelerators signifies a commitment to providing a comprehensive, integrated solution for the execution of AI models on IBM Z systems.

The dynamism of AI necessitates a versatile approach to training, and IBM recognizes this by affording customers the freedom to choose their preferred platforms. Whether leveraging the power of on-premises IBM Z or embracing the flexibility of a hybrid cloud environment, users can train AI models without constraint. This adaptability forms the bedrock of the “Train anywhere, Deploy on IBM Z” paradigm, a concept made possible by the ONNX technology. ONNX, a standardized format for AI models, not only streamlines development and training but also introduces a level of interoperability that transcends the confines of specific frameworks. Data scientists benefit from the flexibility to construct and train models in their preferred environments, abstracting away the concerns of downstream inference-related challenges. IBM's commitment to simplifying deployment further materializes through the finely tuned ONNX model compiler, designed specifically for IBM Z. This optimization effort extends beyond ONNX to embrace key open-source frameworks like TensorFlow, encompassing components such as TensorFlow Serving. The result is a seamless integration with IBM Z systems, ensuring a harmonious marriage between cutting-edge AI technologies and robust, transactional workloads. Introducing ONNX as a mediator between frameworks not only facilitates support for a diverse array of frameworks but also pioneers a novel approach to AI model compute handling. This involves the compilation of models into runnable code with the IBM Z Deep Learning Compiler (zDLC), marking a transformative shift in the way AI models are processed and executed.

The panoramic landscape of AI accelerators, TensorFlow, PyTorch, and ONNX converges into a holistic approach, empowering users with unparalleled flexibility and interoperability in AI model training and construction.

The narrative unfolds as a testament to IBM's commitment to ushering in a new era of adaptability, efficiency, and innovation within the ever-evolving domain of AI and ML.

The combination of these resources ensures that AI workloads can be efficiently processed, making the most of the AI accelerators integrated into the IBM Z and IBM Telum environments. This seamless integration optimizes AI model training, deployment, and execution within enterprise environments.

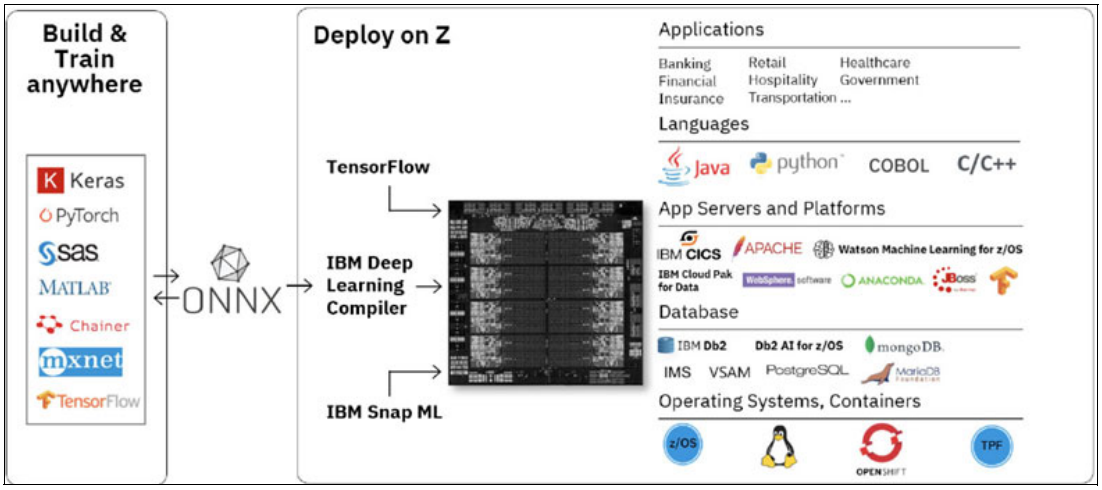

Figure 2-1 ONNX ecosystem for IBM z16 Integrated Accelerator for AI on IBM Telum

There are several key steps involved with effectively utilizing the IBM z16 Integrated Accelerator for AI. This approach streamlines the integration of AI processing capabilities into applications, abstracting away low-level complexities and ensuring portability across different AI frameworks.[1]

High-level programming API

Although it is technically possible to interact directly with the IBM z16 Integrated Accelerator for AI using low-level Neural Network Processing Assist (NNPA) instructions, the use of a high-level API is considered more practical. TensorFlow, a robust and widely adopted deep learning framework, plays a pivotal role in this context. Leveraging the high-level API of TensorFlow simplifies the process of interfacing with the IBM z16 Integrated Accelerator for AI, offering developers a standardized set of functions tailored for seamless integration. This abstraction layer not only enhances portability but also mitigates the potential for errors, allowing developers to focus on the overarching logic of their AI models. TensorFlow's direct support for accelerators ensures optimal performance, making it a cornerstone for efficient AI model training and deployment on IBM z16. Similarly, PyTorch, recognized for its dynamic computational graph and intuitive programming interface, is instrumental in the AI landscape. While PyTorch effectively operates on IBM z16, ongoing efforts are directed towards maximizing accelerator exploitation. Despite this work in progress, PyTorch's adaptability and versatility position it as a formidable framework for AI development on IBM z16. The high-level APIs provided by TensorFlow and PyTorch not only facilitate the conversion of data tensors into the required format for AI accelerators but also contribute to enhanced code reusability and maintainability. Developers working with TensorFlow can seamlessly harness the accelerator's capabilities, given its direct support, while those using PyTorch benefit from a framework that aligns with the ongoing efforts to optimize for accelerators on IBM z16.

[1] https://doi.org/10.1145/3470496.3533042

IBM Z Deep Neural Network (zDNN) Library

zDNN, an open-source library provided by IBM, acts as an interface for developers to access and interact with the IBM z16 Integrated Accelerator for AI. By utilizing this library, developers can access a set of high-level functions and tools that abstract away the intricacies of the AI accelerator, offering a platform-independent and standardized way to harness its power.

IBM Z Deep Learning Compiler

Another approach to effectively use the IBM z16 Integrated Accelerator for AI is through the IBM Z Deep Learning Compiler (zDLC). This compiler is designed to work with models in the Open Neural Network Exchange (ONNX) format. Recently, the compiler has been revamped using the Multi-Level Intermediate Representation (MLIR), which enables multi-phased optimization. The key advantage of the IBM zDLC is that it allows developers to optimize for both the general-purpose core architecture and the AI accelerator. The flexibility of the IBM zDLC design is crucial as it positions the software for supporting multiple levels of AI acceleration as the field evolves. A significant benefit of the compiler-based approach is that it precomputes and converts weights and other static parameters required by the accelerator at compile time. This minimizes the need for run-time conversions and ensures that only input features require dynamic processing. An important aspect of what IBM zDLC enables is the tight integration of the AI model with the application. The compiled AI model is linked as executable code to the application or the serving framework. Therefore, there is no dependency on the actual AI framework when running the model.

Mapping and compilation

When it comes to mapping an operation from a high-level AI model to the AI accelerator, developers have options. The operation can be mapped natively from the AI framework through a device to a corresponding operation in a low-level library, like zDNN. From there, the hardware NNPA instruction controlling the AI accelerator is called to execute the operation.

Alternatively, developers can export the AI model to ONNX format and then compile it using the IBM zDLC. The compilation process results in an executable that is linked to the native zDNN library. This executable can be run on the hardware, utilizing the AI accelerator's capabilities to execute the operation efficiently.

The integration of the AI accelerator into software applications is made more accessible and efficient using high-level programming APIs, libraries like zDNN, and advanced compilation tools like the IBM zDLC. These tools abstract the hardware complexity and provide a standardized and portable means of harnessing the power of AI acceleration, enhancing the capabilities of AI frameworks and applications.

2.1.3 IBM Z processor features

In this section, we delve into the technical details of the IBM z16 Integrated Accelerator for AI on IBM Telum, a cutting-edge chip designed for high-performance AI and ML workloads. The IBM z16 Integrated Accelerator for AI is a critical component of the Telum architecture, providing exceptional AI processing capabilities.[2]

Architecture overview

IBM Telum is the next-generation chip designed for IBM z16 and IBM LinuxONE systems, aimed at meeting the requirements of enterprise-class workloads. This chip offers significant advancements in performance and system capacity compared to its predecessor, the IBM z15®.

[2] https://doi.org/10.1145/3470496.3533042

Telum is fabricated using Samsung's cutting-edge 7 nm. Telum boasts an impressive 22.5 billion transistors, indicating a high level of integration and complexity. This large transistor count enables it to handle a wide range of computational tasks efficiently. Telum operates at a clock speed of over 5 GHz. High clock speeds are crucial for delivering high performance and responsiveness in enterprise workloads. Telum is comprised of eight Simultaneous Multi-Threading 2 (SMT2) cores. SMT2 technology allows each core to execute multiple threads simultaneously, increasing overall processing capacity (see Figure 2-2).

Each core in Telum is equipped with a specific cache hierarchy, which includes:

- A 128 KB L1 instruction cache, responsible for storing frequently used program instructions.
- A128 KB L1 data cache, dedicated to holding frequently accessed data.
- A 32 MB semi-private L2 cache, shared by a subset of the processor cores.

The cache design of the Telum has received significant advancements. In earlier designs, L3 caches were shared across processor chips, and a separate chip was responsible for managing a large L4 cache. In contrast, Telum integrates all this cache logic into a single chip.

The L2 cache is quadrupled in size and now offers a generous 32 MB capacity while maintaining very low latency. A virtual L3 cache of 256 MB is created by leveraging the semi-private L2 caches on the chip and connecting them through a high-bandwidth ring infrastructure. Telum can also operate in large-scale systems that can connect up to 32 individual chips and 40 TB of memory using multiple chip configuration systems.

The IBM z16 Integrated Accelerator for AI on Telum is a compact yet powerful component, spanning an area of approximately 7 mm² on the chip, roughly one-third the size of a general-purpose core. It is directly connected to the chip fabric and positioned at the chip level, sharing its area with I/O logic that requires a specific physical placement pattern. While this choice makes the physical design and timing closure more challenging, it offers flexibility for future expansions as there is no fixed bounding box for the accelerator.

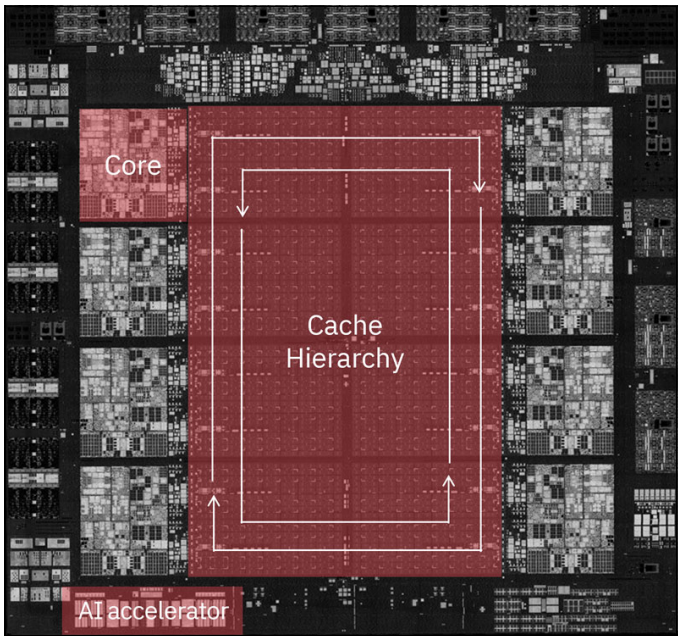

Figure 2-2 IBM Telum chip die: IBM z16 Integrated Accelerator for AI on IBM Telum Processor Industrial Product

Physical implementation

The physical implementation of the IBM z16 Integrated Accelerator for AI is illustrated in Figure 2-3. It is comprised of various components, including Processor Tiles (PT), Processing Elements (PE), Special Function Processors (SFP), scratch pad memory, First-In, First-Out (FIFO) data mover/formatters, and prefetcher/write-back units. The AI accelerator is equipped with around one million flip-flop equivalent circuits and a 512 KB scratch pad memory for data prefetching and write-back operations. This accelerator integrates seamlessly with IBM Telum cores, utilizing a novel NNPA instruction, which works in conjunction with firmware running on both the core and the accelerator.

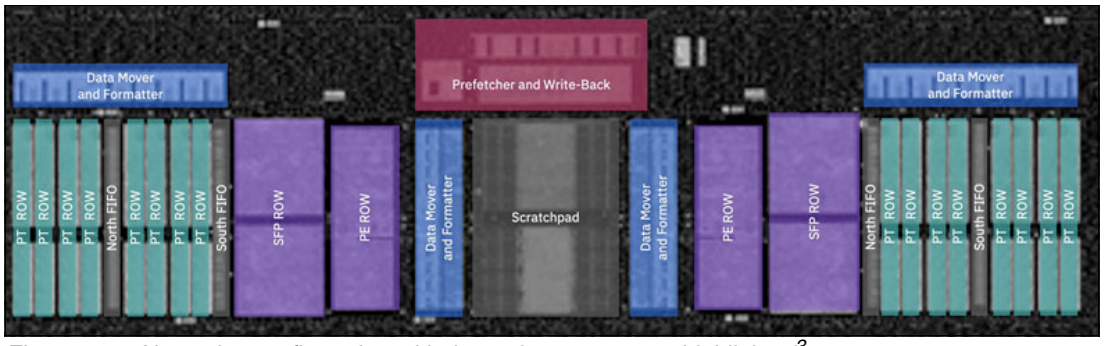

Figure 2-3 AI accelerator floor plan with the main components highlighted[3]

NNPA

NNPA is a key innovation in IBM Telum's design. It is an interface to the IBM z16 Integrated Accelerator for AI, the IBM z16 Integrated Accelerator for AI. This non-privileged instruction is memory-to-memory in nature, allowing operands and tensor data to reside in the user's program memory, with registers used to hold pointers to operand descriptors. The NNPA instruction supports a range of AI operations.

The NNPA instruction streamlines various AI functions, making it highly efficient for tasks like matrix multiplication, element-wise operations, and activation functions. It also provides a query function (QAF) that allows software to determine the supported functions, facilitating seamless migration of AI models between different firmware levels and machine types.

Operation classes

The IBM z16 Integrated Accelerator for AI supports various operation classes, including:

- Elementwise ops: These encompass basic tensor element arithmetic operations, often used in convolutional neural networks (CNNs).
- Activation ops: These functions efficiently implement common activation functions like rectified linear unit (ReLU), Sigmoid, and various Long short-term memory (LSTM) and gated recurrent unit (GRU) activation functions.
- Normalization ops: These operations include SoftMax and 2D-batch normalization.
- Pooling ops: This class handles 2D pooling operations, including average and maximum pooling.
- Systolic ops: These operations are responsible for critical AI functions, such as 3D convolution and matrix multiplication, which make up a significant portion of AI model inference and training.

[3] https://doi.org/10.1145/3470496.3533042

IBM z16 Integrated Accelerator for AI micro-architecture

The AI accelerator's micro-architecture is divided into two main components: compute arrays and data movers.

Compute arrays

The AI accelerator boasts the following two separate compute arrays, each designed for specific tasks:

- Matrix array

  The first array is known as the matrix array, consisting of 128 Processor Tiles (PTs) organized into rows. These PTs are optimized for tasks like matrix multiplication and convolution operations. Each PT incorporates an eight-way Single instruction, multiple data (SIMD) engine, tailored for multiply-accumulate operations in DLFloat-16 format. This format is crucial for deep learning tasks and offers better quality and convergence compared to other formats.

- Data Processing and Activation Array

  The second array is the data processing and activation array, comprised of 32 processor tiles organized into 2 rows. One row features Processing Elements (PE), which are eight-way SIMD engines optimized for arithmetic, logical, look-up, and type conversion functions in DLFLT-16 format. The other row consists of Special Function Processors (SFP), which are a super set of PE, capable of executing operations on DLFloat-32 elements with four-way SIMD. This array is employed for non-systolic functions and data preparation and gathering for systolic functions.

Data movers

Efficient data flow and manipulation are essential for maximizing the AI accelerator's performance. The accelerator employs data movers to manage data around the compute arrays with low latency. The data flow begins with intelligent data prefetch, which loads data just-in-time into the 512 KB scratch pad.
Multiple sections within the scratch pad enable double buffering of data and compute streams, allowing for parallelism and increased performance.

Data formatters ensure that data arrives at the PT compute engines in the required format and layout. Complex function compute arrays (PE/SFP) may perform additional data manipulation before sending data to PT compute engines. The data movers provide over 600 GB/s of data bandwidth, ensuring efficient data transfer.

The write-back engine collects results from the scratch pad and stores them back to caches or memory through the chip ring with a write bandwidth of 80+ GBps.

This IBM z16 Integrated Accelerator for AI on IBM Telum is a critical component that significantly enhances the chip's AI processing capabilities, making it well-suited for a wide range of AI and ML workloads. Its innovative design and efficient micro-architecture enable Telum to deliver outstanding AI performance while maintaining compatibility with a diverse range of AI operations.

2.1.4 AI software

In this section, we provide an overview of the AI software available on IBM Z.

IBM Z Application Performance Management Connect

IBM Z Application Performance Management Connect, or simply APM Connect, is a solution available on IBM z16 that aims to provide real-time monitoring, analysis, and management capabilities for applications running on IBM Z mainframe systems. This solution is designed to help organizations optimize the performance of their mainframe applications and ensure a seamless end-user experience. For more information, see IBM Z APM Connect overview.

IBM Operational Decision Manager

Available as an automation capability and core program offering provided by the IBM Cloud Pak® for Business Automation, available for IBM z16, IBM Operational Decision Manager can be used to discover, capture, analyze, automate, and govern rules-based business decisions on-premises or in the cloud. For more information, see Operational Decision Manager.

IBM Cloud Pak for Data on IBM Z

To confidently leverage your enterprise data within a secure, resilient IBM Z and IBM LinuxONE private cloud infrastructure, IBM Cloud Pak for Data on IBM Z is available for IBM z16 LinuxONE Emperor 4.

IBM Cloud Pak for Data is an integrated data and AI platform designed to help organizations collect, organize, and analyze data for AI and ML. It offers a wide range of tools and services for data management, data science, and AI model development. While IBM Cloud Pak for Data is typically deployed on various cloud environments, including public, private, or hybrid clouds, it is also compatible with IBM Z mainframes for organizations that require on-premises or hybrid cloud deployments.

The following are some key points to consider when running IBM Cloud Pak for Data on IBM Z:

- Hybrid cloud deployment: IBM Cloud Pak for Data is designed for hybrid cloud deployments. You can run it on various cloud infrastructures, including on-premises IBM Z mainframes.
- Data integration: The platform can connect to a wide range of data sources, including mainframe data, to integrate and consolidate data from multiple locations.
- Data governance: IBM Cloud Pak for Data provides data governance and compliance features, making it suitable for organizations that require strict data security and regulatory compliance, which is common in financial and healthcare sectors where IBM Z is often used.
- AI and ML: The platform offers tools and services for building and deploying AI and ML models to analyze and gain insights from your mainframe data.
- Model deployment: IBM Cloud Pak for Data enables you to deploy AI models and applications on IBM Z to ensure that your insights are actionable and can be integrated with your mainframe applications.

For technical details on the latest version of IBM Cloud Pak for Data, see IBM Cloud Pak for Data 4.8.x.

3

The power of Linux on IBM Z for AI workloads

Workloads running on Linux on IBM Z inherit the reliability, scalability, and security of the IBM Z platform. Linux on IBM Z workloads share resources, reduce costs, and improve performance and efficiency. IBM z16 is built for Linux and cloud computing and can expedite AI acceleration to enable decision velocity. Quantum-safe technologies help protect your business now and into the future and a flexible infrastructure meets resiliency and compliance demands. Additionally, with an open-source solution and a high-performance operating system, Linux on IBM Z provides a scalable infrastructure that can quickly, efficiently, and cost-effectively process large amounts of data to generate actionable business insights in real-time.

In this chapter, we illuminate the exceptional capabilities of Linux on IBM Z when applied to AI workloads and outline the immense potential and advantages of leveraging Linux on IBM Z for AI applications. Our journey begins with the versatility offered by Linux on IBM Z, allowing you to “Train anywhere, deploy on IBM Z.” We then dive into the intricacies of scalability and performance optimization, where we unravel the layers of ongoing analysis to optimize performance, delve deeper into how to make intelligent decisions surrounding capacity management and discover strategies for optimizing costs, ensuring your AI workloads run efficiently and economically. We also address security and compliance concerns, exploring how to create a fortified environment for your AI projects. The chapter unfolds further with a focus on technical advancements in data management and storage, uncovering the advantages of utilizing IBM Z in data management and storage, and gain a better grasp of the significance of data security and compliance. Additionally, we explore how scalability and performance are intricately tied to the technical advancements in this domain.

By the end of this chapter, you gain a holistic understanding of how Linux on IBM Z empowers AI workloads. The benefits extend beyond theoretical knowledge; you will be equipped with practical insights and strategies that can be directly applied to the efficiency, security, and scalability of your AI projects.

3.1 Train anywhere, deploy on IBM Z

IBM recognizes that the AI and inferencing landscapes are different. The training landscape serves as the playground for data scientists that are focused on improving model accuracy. Within the training landscape, data scientists use platforms that may be ideal for training, but are not necessarily efficient for deploying/inferencing models. The “train anywhere, deploy on IBM Z” approach enables you to build and train models on the platform of your choice (including on-premises on IBM Z in a hybrid cloud), leveraging any prior investments. You can then deploy those models to an environment that has transactional and data affinity to the use case, such as transactional processing on IBM Z.

Enabling this strategy, IBM is architecting solutions to enable model portability to IBM Z without requiring additional development efforts for deployment. We are investing in ONNX technology, a standard format for representing AI models allowing a data scientist to build and train a model in the framework of choice without worrying about the downstream inference implications. To enable deployment of ONNX models, we provide an ONNX model compiler that is optimized for IBM Z. In addition to this, we are optimizing key open-source frameworks such as TensorFlow and TensorFlow Serving for use on IBM Z.

Open-source availability and scalability

With open-source availability, enterprises can consistently develop, deploy, and implement next-generation applications by navigating on-premises and between different cloud environments with ease. With an open-source ecosystem and cloud-native deployment options, Linux on IBM Z system enables enterprises to scale existing apps and roll out emerging technologies across all types of cloud environments.

3.2 Scalability and performance optimization

The high workload density, with up to thousands of virtual Linux servers on an IBM z16 multi frame, usually means fewer components, lower management effort, and fewer software licenses compared to competitive platforms.

Run the Yahoo Cloud Serving Benchmark (YCSB) on MongoDB without sharding on IBM z16 multi frame with 6 IFLs in total and achieve the same throughput as on MongoDB with 4 shards compared to x86 systems with 144 cores in total, which provides a 24:1 core consolidation ratio in favor of IBM z16 multi frame.

DISCLAIMER: Performance results based on IBM internal tests running YCSB 0.10.0 (YCSB) benchmark (read-mostly) on MongoDB Enterprise Release 5.0.6 with 3-node replication. IBM z16 A01 MongoDB was set up without sharding but with 2 replicas. IBM z16 A01 configuration: Logical Partition (LPAR) with 4 dedicated cores and 2 LPARs with 1 core each, each with Simultaneous Multi-Threading (SMT) and 128 GB memory, 1 TB FlashSystem 900. x86 config: 9 Intel Xeon Gold 5218 CPU @ 2.30 GHz with Hyperthreading turned on, 192 GB memory, 1 TB local RAID5 SSD storage, Red Hat Enterprise Linux 8.4 running MongoDB, driven remotely by YCSB using 2 x86 servers with total 128 threads. Results may vary.

Using IBM Java 8 SR7 or IBM Semeru Runtime Certified Edition 11, run Business Rules Processing with IBM Operational Decision Manager 8.11.00 on Linux on IBM z16 for up to 70% higher throughput per core versus running the same application on a compared x86 server.

DISCLAIMER: Performance results are based on the average of measurements done using IBM Operational Decision Manager (ODM) 8.11.0 with IBM Java 8.0.7.10 and IBM Semeru Runtime Certified Edition 11.0.15.0 on IBM z16 and on a compared x86 server. Two different configurations were tested: executing 2005 rules (from a ruleset containing 14560 rules), and executing 80 rules (from a ruleset containing 300 rules). IBM z16 configuration: Linux on IBM z16 LPAR with Red Hat Enterprise Linux 8.5 (Ootpa) and 4 IFLs (SMT). x86 server configuration: Red Hat Enterprise Linux release 8.6 (Ootpa) and 4 SMT-2 cores (Cascade Lake Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz). Results may vary

Impressive scalability, horizontal and vertical, is provided with the IBM Z capabilities in combination with the virtualization technologies IBM z/VM and KVM. Resources can be prioritized dynamically and efficiently between workloads, delivering them whenever and wherever they are needed. IBM z/VM virtualization technology offers deep integration with IBM Z, allowing for high levels of resource sharing, data-in-memory techniques, outstanding I/O bandwidth, availability, and security. KVM virtualization enables the use of Linux administration skills on IBM Z. KVM is delivered with the Linux distributions for IBM Z and is optimized to benefit from the IBM Z capabilities.

The high flexibility of IBM Z is shown not only in the hardware capabilities. The available solution portfolio for Linux on IBM Z products and frameworks (from IBM, independent software vendors, and open source) is immense. The .NET 7.0 framework is available on IBM Z with Red Hat Enterprise Linux 8.7 and later, and Red Hat Enterprise Linux 9.1 and later. IBM Dynamic Partition Manager provides a simplified configuration for Linux servers, allowing for a quick and easy adoption.[1]

3.2.1 Ongoing analysis to optimize performance

The ultimate goal of the IT operations team is to ensure the infrastructure under their management is able to support the needs of the business. When issues occur, it may be the result of unexpected constraint or capacity issues within the environment. The mainframe today is part of a complex hybrid cloud ecosystem where transactions are initiated from many external sources, like a website or mobile device, and ultimately drive workloads on IBM Z, such as updating a customer record within IBM Db2® on IBM Z. Workloads can spike at any time of the day and these patterns are less predictable than ever, making it critical that adequate resources are available to ensure the risk of operational issues is minimized.

When analyzing operational data, teams can be overwhelmed by both the breadth and depth of information available to them. The mainframe is blessed with a rich set of metrics available through SMF and other sources; however refining these to identify what is important and what is not is time consuming. It is often exacerbated by the multitude of products used by the operations team across different domains, resulting in a lack of system-wide insight.

To know what to focus on when performing performance analysis, many mainframe operational teams also depend on key domain experts, armed with years of experience and detailed knowledge of the unique nature of their enterprise's applications and environment. When challenged with the loss of these experts, newer members of the operations teams need more guidance on where to focus their efforts.

[1] https://www.ibm.com/downloads/cas/X6OLNMGE

This widening skills and expertise gap makes it difficult to build deep data insights and reports, which is critical for performance root cause analysis, ultimately making it costly to identify and validate capacity and performance optimization opportunities.

3.2.2 Making intelligent decisions around capacity management

Traditionally, performance and capacity management is a reactive task. Data would be collected daily, then processed in batch overnight to create a set of reports to analyze. This limits the ability to make timely decisions and generate ad hoc reporting to deep dive into issues. Such approaches are not satisfactory today and the need is for near real-time access to information with the ability to analyze and correlate data from multiple sources to make accurate decisions. The data needs to be collected, curated and reported on a near real-time fashion not so much to provide alerting but to have the detailed information to hand to make accurate decisions if an incident has been detected.

When making longer-term capacity planning decisions, the focus is often around making the best use of existing resources and where to make smart investments, for example, purchasing of new hardware or upgrading to latest technologies. Blindly making these capital investments can result in mistakes where the expected benefits are not realized to the level expected. Pulling together the right data to make the correct decision can be difficult and time consuming, to say nothing of the capacity planner skills that may be needed to make judgments and recommendations. The ability to model out changes in growth and other patterns quickly and easily can help identify where to optimize resources, or take preventative actions, to avoid performance and capacity constraints in the future.

3.2.3 Optimizing for cost

Cost management has also been a task that often falls to the same team responsible for performance and capacity management. Given some of the ways software is licensed on the mainframe, the configuration and management of the IBM z/OS environment can impact its cost. Therefore, the operations team needs to balance optimal performance with licensing costs. This balance can go too far in one direction; if the system is capped or constrained too much then workloads can back up and service level agreement (SLA) compliance can be impacted. Having a clear view of which workloads are contributing to software costs, and how they vary over time, is an essential set of reports the performance team need to review. This is true whether the enterprise is using a traditional Rolling 4-hour average to manage software costs, or the more recent Tailored Fit Pricing model.

When there is complete view of the workloads as they relate to cost, then informed decisions can take place around optimizing the existing workloads. This could be moving non-critical workloads to quieter times on the system, removal of non-valuable applications, database re-orgs, or recompilation of some applications with a more modern compiler to reduce the number of MSUs it consumes. With evidence to support decision making, there is greater chances of success in performance and capacity management.

3.3 Addressing security and compliance concerns

The amalgamation of AI with Linux workloads on IBM Z mainframes is a technical marvel, providing unparalleled performance and efficiency. Yet, this fusion also introduces intricate security and compliance challenges. In this section, we delve into the technical nuances of these concerns and discuss the rigorous measures required to ensure the secure and compliant operation of AI-enriched Linux workloads on IBM Z.

3.3.1 Security concerns

When it comes to security concerns, we will focus on the following three categories: data protection and privacy, AI model security, and insider threats.

Data protection and privacy

From a technical standpoint, safeguarding data privacy and protection is paramount. AI applications often necessitate access to sensitive data, and failure to adequately secure this information can result in grave consequences. Employing robust encryption techniques, secure access controls, and stringent authentication mechanisms is essential.

To address this technical concern, organizations should implement end-to-end data encryption protocols, both at rest and in transit. Furthermore, secure APIs, which restrict AI models' interaction with sensitive data, are imperative. Compliance with data protection regulations, like the General Data Protection Regulation (GDPR), the Health Insurance Portability and Accountability Act (HIPAA), and the California Consumer Privacy Act (CCPA), should be intricately woven into the technical fabric to mitigate legal ramifications.

AI model security

AI models, from a technical perspective, are susceptible to an array of attacks, including adversarial attacks, model inversion, and model stealing. These vulnerabilities can lead to the AI model generating erroneous predictions or revealing sensitive information. Ensuring the security of AI models is a multifaceted task.

To mitigate these technical risks, a proactive approach is necessary that includes regular updates and patches for AI models to fortify them against known vulnerabilities. Additionally, anomaly detection systems can be deployed to identify unusual AI model behavior that could signify an attack. The implementation of security audits and penetration testing serves as a technical safety net to uncover potential weaknesses.

Insider threats

The technical dimension of insider threats is a substantial concern in environments deploying AI-enriched Linux workloads on IBM Z. Personnel with access to the system can intentionally or inadvertently compromise security through the misuse of their privileges or unauthorized data disclosure.

Addressing this technical challenge necessitates the implementation of strict access controls, continuously monitored through technical means. User behavior analytics, which scrutinizes digital footprints, becomes an invaluable tool to detect suspicious activities. Privileged access management tools play a technical role in confining access to critical resources. Employee training on security protocols and practices is the last line of defense, promoting a culture of heightened security awareness.

3.3.2 Compliance concerns

When it comes to compliance concerns, we will focus on the following categories: regulatory compliance, data governance, and ethical concerns.

Regulatory compliance

Technical adherence to industry-specific regulations and standards is a critical aspect when incorporating AI into Linux workloads on IBM Z. Failing to meet these technical compliance requirements can lead to legal repercussions, financial penalties, and damage to an organization's reputation.

Various industries have distinct technical compliance mandates, such as healthcare's HIPAA, finance's Sarbanes-Oxley Act (SOX), or the broader GDPR. A deep understanding of these regulations is required, and organizations must translate this understanding into technical safeguards.

Data governance

From a technical perspective, effective data governance is pivotal for compliance when integrating AI into Linux workloads on IBM Z. Proper management of data, including collection, storage, access, and retention, must be carried out. Identifying where data resides and who can access it is fundamental to maintaining technical compliance.

In this context, organizations should implement data classification systems and access control policies, enforced through technical means. Technical tools for data auditing and monitoring are critical to ensure data usage aligns with regulatory requirements.

Ethical concerns

Though not a strictly technical requirement, ethical concerns are a technical aspect to consider when enriching Linux workloads with AI on IBM Z. The potential for AI algorithms to perpetuate biases or make ethically questionable decisions necessitates a thoughtful approach.

From a technical perspective, organizations should develop and enforce ethical guidelines for AI development and deployment. These guidelines should include technical measures for ensuring fairness, transparency, and accountability in AI models. Regular technical audits of AI models for biases, as well as ongoing evaluation of their impact on decisions, are essential in addressing ethical concerns.

Enriching Linux workloads with AI on IBM Z holds enormous technical potential for organizations to harness data and make informed decisions. Yet, these prospects come with intricate security and compliance technical challenges that demand rigorous attention. Data privacy, AI model security, insider threats, regulatory compliance, data governance, and ethical considerations, when addressed with precision and technical sophistication, ensure the secure and compliant operation of AI-enriched Linux workloads on IBM Z.

Through a harmonious blend of innovation and technical security measures, organizations can unlock the full potential of AI while safeguarding their data, reputation, and compliance with industry standards. The balance between technical innovation and security is the key to realizing a secure and bright future for AI-driven Linux workloads on IBM Z.

3.4 Technical advancements in data management and storage

The convergence of AI with Linux workloads on IBM Z mainframes represents a significant technical leap in data management and storage. The renowned reliability, scalability, and security of IBM Z, coupled with specialized tools such as watsonx.data and IBM Data Virtualization Manager for z/OS, create a robust foundation for AI-driven data solutions. In this section, we will delve into these IBM products and explore their pivotal roles in the technical aspects of managing and storing data for AI workloads.

3.4.1 Advantages of IBM Z in data management and storage

IBM Z mainframes, recognized for their technical prowess, excel in data management and storage for AI applications. The inherent attributes of reliability, security, and scalability make them the ideal platforms for AI workloads that demand robustness and data integrity.

Data management with IBM watsonx.data

IBM watsonx.data is a multifaceted IBM product that addresses the technical complexities of AI data management. It offers a comprehensive suite of tools designed for Linux workloads on IBM Z, simplifying the storage, organization, and utilization of data.

Technical features of watsonx.data include the following:

- Data integration: watsonx.data's technical core enables the integration of data from diverse sources into a unified platform. This integration process simplifies the data preprocessing needed for AI analysis.
- Data quality and governance: Ensuring data accuracy and adhering to governance standards is critical. watsonx.data incorporates tools for data cleansing, transformation, and governance, allowing data scientists to maintain the quality and regulatory compliance of the data.
- Scalability: The technical scalability of IBM Z, coupled with watsonx.data's capabilities, enables organizations to manage and process increasingly large datasets. This scalability ensures high-performance data management even as data volumes grow.

IBM Data Virtualization Manager for z/OS

IBM Data Virtualization Manager for z/OS, another essential IBM tool, enhances data management on the IBM Z platform. This tool streamlines data access and integration, reducing the technical complexities associated with data movement.

Technical features include:

- Data source integration: Connect to a variety of data sources, including databases and cloud storage, to provide a technical solution for accessing data from various locations.
- Data virtualization: Simplify data access for AI applications by creating a virtualized layer that abstracts the complexities of data sources. This technical layer offers a unified interface for accessing data, eliminating the need to manage multiple data connectors.
- Performance optimization: Technical optimizations ensure that AI workloads running on IBM Z can retrieve and integrate data swiftly and efficiently, minimizing data latency and ensuring timely data access.

Real-world application

To demonstrate the technical efficacy of these data management and storage tools, let us explore a real-world application.

In the logistics and supply chain industry, real-time data access and analysis are crucial for route optimization, inventory management, and demand forecasting. By utilizing watsonx.data and IBM Data Virtualization Manager for z/OS, companies can efficiently integrate data from IoT devices, GPS sensors, and inventory databases. The tools manage and provide real-time access to data, enabling AI algorithms to make data-driven decisions in real-time, thereby enhancing supply chain efficiency.

3.4.2 Data security and compliance

In the realm of AI workloads, data security and compliance are paramount technical considerations. The built-in security features of IBM Z, including hardware-based encryption, secure service containers, and secure boot, align with watsonx.data's data governance capabilities to provide a holistic approach to data security and compliance. Together, they ensure that data remains protected against unauthorized access and adheres to industry-specific regulations.

3.4.3 Scalability and performance

The technical scalability of IBM Z and performance attributes play a pivotal role in data management for AI workloads. As data volumes expand, the platform can handle the increased demand without compromising on performance. watsonx.data's data integration and transformation capabilities, combined with optimized data access, ensure that AI applications running on IBM Z can process and analyze data swiftly and efficiently, making the most of the mainframe's capabilities.

The integration of AI into Linux workloads on IBM Z marks a significant milestone in data management and storage. The reliability, security, and scalability of IBM Z, when complemented by specialized tools like watsonx.data and IBM Data Virtualization Manager for z/OS, create an ideal environment for AI-driven data solutions.

These technical tools simplify data integration, ensure data quality and governance, provide efficient data access, and enhance data security and compliance. Real-world applications across industries demonstrate the power of this technical synergy, from supply chain optimization to healthcare diagnostics.

In this era of digital transformation, the harmonious interplay between IBM Z and these data management and storage tools empowers organizations to navigate the complexities of AI and data management. The technical strengths of IBM Z, together with these tools, offer a solid foundation for innovation and data-driven decision-making, ensuring that organizations remain at the forefront of technological advancements.

3.5 A toolkit for success

The convergence of AI and Linux workloads on IBM Z mainframes has ushered in an era of unprecedented computational power and performance. These technologies, when combined, enable organizations to process vast amounts of data and gain deeper insights. In this section, we explore the specific tools, libraries, and real-world examples that demonstrate how AI can be seamlessly integrated into Linux workloads on IBM Z.

3.5.1 The IBM Z advantage

With IBM Z, you can harness the power of AI without compromising the core values of trust, availability, and performance.

Tools for AI on IBM Z

The tools for AI on IBM Z represent a critical advancement in the domain of mainframe application modernization and AI integration. The IBM watsonx Code Assistant for IBM Z is designed to expedite the translation process of COBOL to Java while significantly enhancing developer productivity on the platform. This tool's functionality spans across the entire application modernization lifecycle, starting with its adept application discovery capability. By mapping out the technical landscape of the application and understanding its dependencies, the Code Assistant establishes a solid foundation for subsequent processes.

Following the discovery phase, the tool seamlessly transitions into automated refactoring, a process that aims to decompose monolithic applications into modular COBOL business services. Leveraging the insights gleaned from application discovery, the Code Assistant identifies and isolates specific elements ripe for refactoring, thereby facilitating the transformation of legacy applications into more agile, service-oriented architectures.

However, the true innovation of the IBM watsonx Code Assistant for IBM Z lies in its utilization of generative AI algorithms to execute the intricate task of transforming COBOL business services into object-oriented Java code.

This application of generative AI not only streamlines the modernization process, but also ensures the integrity and functionality of the resulting Java code. By leveraging AI-driven automation, organizations can mitigate the risks associated with manual code translation while accelerating the pace of modernization initiatives. Furthermore, the integration of IBM watsonx Code Assistant with IBM Z ensures that the performance, security, and resiliency capabilities inherent to the platform are seamlessly preserved throughout the modernization journey.

You can find a more detailed discussion of IBM watsonx Code Assistant for IBM Z and its capabilities in 7.2.1, “watsonx future applications” on page 80.

In parallel, the AI Toolkit for IBM Z and IBM LinuxONE emerges as another indispensable resource for organizations seeking to harness the power of AI within their mainframe environments. Comprising a suite of popular open-source AI frameworks, meticulously adapted and supported for IBM Z and IBM LinuxONE hardware, this toolkit represents a paradigm shift in the accessibility and implementation of AI technologies within traditionally conservative computing ecosystems.

By providing elite support and rigorous security validation, the AI Toolkit is designed to address the unique challenges associated with deploying AI frameworks on mainframe architectures. Through features such as the IBM z16 Integrated Accelerator for AI, organizations can achieve unparalleled levels of performance optimization, thereby unlocking new possibilities for AI-driven innovation. From accelerating TensorFlow inference to deploying ML models with Snap ML, the AI Toolkit empowers organizations to capitalize on both deep learning and traditional ML approaches, thereby fostering a culture of continuous innovation and digital transformation.

You can find a more detailed discussion of the featured technologies, including the NVIDIA Triton Inference Server, of the AI Toolkit for IBM Z and IBM LinuxONE and how to get started using the Toolkit in 5.5.1, “AI Toolkit for IBM Z and IBM LinuxONE” on page 57.

The tools for AI on IBM Z ecosystem, encompassing the watsonx Code Assistant for IBM Z and the AI Toolkit for IBM Z and IBM LinuxONE, represents a transformative leap forward in the realm of mainframe application modernization and AI integration. By combining the power of generative AI with the robustness of IBM Z hardware, these tools enable organizations to modernize their legacy applications with unprecedented speed, efficiency, and confidence, thus unlocking new opportunities for innovation and growth in the digital age.

3.5.2 Real-world applications

To illustrate the effectiveness of AI on Linux workloads on IBM Z, let us consider two real-world examples: financial fraud detection and manufacturing quality control.

Financial fraud detection

Financial institutions deal with massive amounts of transaction data daily. Using the mainframe capabilities of IBM Z in conjunction with AI tools, these institutions can deploy sophisticated fraud detection models. IBM Watson® Machine Learning can help develop and deploy models that analyze transaction patterns, and IBM z/OS Container Extensions can ensure these models run securely in isolated containers. The result is an efficient, reliable, and highly secure fraud detection system.

Manufacturing quality control

Manufacturing companies dealing with intricate product lines and stringent quality standards can benefit from AI-driven quality control. By deploying AI models on IBM Z through IBM Watson Machine Learning, these companies can inspect products using image recognition and predictive analytics. This ensures that defective items are detected and removed from the production line, thereby maintaining the highest quality standards.

Linux workloads enriched with AI on IBM Z offer organizations the potential to unlock the full power of their data, thanks to a host of specialized tools and libraries tailored for AI workloads. The IBM Z platform's robustness, scalability, and security are perfectly aligned with the requirements of AI applications.

Machine Learning for IBM z/OS, IBM z/OS Platform for Apache Spark, and IBM z/OS Container Extensions are just a few of the tools and libraries that enable customers to embrace AI on IBM Z. These resources ensure that AI workloads on IBM Z are not only technically feasible but also efficient and secure, helping organizations make the most of their data-driven insights.

3.6 AI open-source ecosystem on IBM Z

The open-source software development process nowadays is a proven model for developing mission critical software. The resurgence of AI started with that spirit right from the beginning. The open-source ecosystem around Linux is where AI models are developed, trained, and deployed.

The most popular AI frameworks like TensorFlow and PyTorch have been developed as Open-Source right from the beginning and are now backed by strong communities including many companies and volunteers from around the world.

As a long-time contributor to many open-source projects, IBM fosters a strong AI open-source foundation for its products like watsonx, IBM Cloud Pak for Data, and the AI Toolkit. While these products are the way to go for mission critical AI deployments, most of the AI journeys start with open-source. For Data Scientists, it is natural to start by installing the components they need from the Python Package Index or Anaconda.

Once the first AI application runs well in the Linux on IBM Z environment, the road towards an enterprise deployment leads to the more sophisticated AI products, which combine the power of open source with IBM products to implement data governance, model monitoring, and explainability.

3.6.1 How to install open-source AI packages

In contrast to many standard tools and libraries most of the AI components are not delivered as part of a Linux distribution. Therefore, alternate channels are needed to handle these packages and its dependencies.

Linux distribution agnostic package managers enable AI package installation across different operating system layers.

Anaconda

Anaconda on IBM Z and IBM LinuxONE is a distribution of binary packages with a strong focus on AI.

It comes with the conda package management tool and can be used to install AI libraries and tools on Red Hat Enterprise Linux, SUSE Linux Enterprise Server, and Ubuntu. Anaconda tries to be as self-contained as possible. It not only packages the additional packages required on top of the Linux distribution, but also essential core libraries, compilers, and interpreters to provide a consistent experience across different Linux distributions and hardware platforms.

IBM works with the company behind Anaconda, Anaconda Inc., to make additional packages and updates available.

Anaconda is available in the following different editions:

- Miniconda
- Anaconda Individual Edition
- Anaconda Commercial Edition

For detailed information on installing Anaconda on Linux or other platforms, see Installing on Linux.

For more information, see Anaconda on IBM Z and IBM LinuxONE.

For a guided example of a credit default risk prediction model leveraging Python and Conda with Linux on Z, see Data and AI: Credit Default Risk Prediction.

Enabling additional packages with the community channels

In addition to the package repository maintained by Anaconda Inc. a community managed edition is available as well. The conda-forge build recipes come without support and warranties. If required, the conda-forge channel can be activated in addition to the official channels. For more information, see “How can I install packages from conda-forge?”

Using IBM provided build recipes

IBM provides additional build recipes in Open-CE project GitHub repository.

These can be used to build packages, which cannot be found in the Anaconda base layer or conda-forge, like TensorFlow. For more information, see Capabilities on Linux on IBM Z and IBM LinuxONE.

Python Package Index

The Python Package Index (PyPI) is the standard way of installing additional components in the Python language environment. It does not focus on AI packages, but thanks to the popularity of Python in the AI world, it has become became the unofficial standard for installing AI software components.

While most of the packages on The Python Package Index (PyPI) are not available pre-built for the IBM Z platform, most of them can be installed seamlessly. The packages, which include native components, are built on the fly by pip in the local environment. In some cases, this requires additional packages to be installed in the underlying distro. For system libraries used during build, the additional development package is required. A typical candidate is the OpenBLAS library, required for NumPy and many others.

In contrast to Anaconda, pip relies on critical components provided by the underlying Linux distribution. One of these components is the Python interpreter. Depending on the exact version of the interpreter, different versions of packages are installed.

The PyPI provides a much larger selection of packages. In order to benefit from the optimized and stable foundation of Anaconda, PyPI packages can be installed on top of an Anaconda base layer.

3.6.2 How to get started with open-source AI frameworks

Many AI frameworks and libraries can easily be installed using the methods from the last chapter. Using Anaconda and PyPI you will have access to a wide range of tools and ML frameworks like NumPy, Pandas, matplotlib, Jupyter, SciPy, scikit-learn. In this chapter we will have a look at important deep learning frameworks and how to install them as open-source components in a Linux on IBM Z and IBM LinuxONE environment.

TensorFlow

TensorFlow is one of the most popular AI frameworks. It has been developed by Google and is fully open-source.

Using the IBM Container Registry Image

IBM provides ready-to-go container images in the IBM container registry. These images are built from scratch by IBM. The content is vetted for known vulnerabilities and can be downloaded and used free of charge.

The TensorFlow image is already set up to make use of the IBM z16 Integrated Accelerator for AI and can be deployed in Linux on Z and IBM z/OS zCX. For more information, see TensorFlow.

Building TensorFlow from source

Building TensorFlow directly from the source repository is a rather involved task and not recommended. It is recommended to use the following IBM provided build recipes instead:

- Use the publicly available build recipes for conda. Instructions can be found in the CP4D on IBM Z manual. For more information, see Capabilities on Linux on IBM Z and IBM LinuxONE.
- Use the publicly available build script for Ubuntu distros. For more information, see Building TensorFlow.

Converting the AI model to ONNX

Leverage the ONNX interchange format so that TensorFlow models can be run in alternate frameworks and do not need TensorFlow to be installed in the environment where the AI model will be deployed. If the TensorFlow AI model has been developed on a non-IBM Z platform, it is recommended to run the tf2onnx open-source tool on that platform to convert it into the ONNX format. Copy the ONNX file over to IBM Z or IBM LinuxONE and compile it there using the IBM Z Deep Learning Compiler (zDLC).

For detailed instructions on converting TensorFlow models, using the tf2onnx tool, to ONNX format and compiling them with zDLC, see 5.3.2, “IBM Z Deep Learning Compiler” on page 52.

PyTorch

PyTorch is a frequently used AI framework and became particularly popular with the raise of Transformer models and the Hugging Face model repository. PyTorch has been developed originally by Meta but is now under the umbrella of the Linux Foundation.

PyTorch on IBM Z and IBM LinuxONE does not make use of the IBM z16 Integrated Accelerator for AI but has been optimized to leverage the IBM Z vector instruction set (SIMD).

It is highly recommended to use 2.1.0 or later IBM Z and IBM LinuxONE.

Using the Anaconda package

The recommended way of installing PyTorch is through Anaconda using the following command:

```
$ conda install pytorch
```

Building from source

Building PyTorch from source is not difficult and can be done by following the standard build instructions. For more information, see Install PyTorch.

Converting to ONNX

PyTorch provides a built-in mechanism to save models in the ONNX format. Use the **`torch.onnx.export`** command to save a model in the ONNX format.

IBM Z Deep Learning Compiler

The zDLC is a static compilation framework, which compiles AI models in the ONNX format into executable code for the IBM Z platform. The generated code can either be compiled to run on CPU or it can leverage the IBM z16 Integrated Accelerator for AI. The output is a Linux shared library, which can be linked either directly to the application or to a serving framework, such as the NVIDIA Triton Inference Server discussed in detail in “NVIDIA Triton Inference Server” on page 58. In both cases no other AI frameworks needs to be installed to run the AI model.

The easiest way to install the zDLC is through the IBM Container Registry.

The IBM Z Deep Learning Compiler uses the ONNX Multi-Level Intermediate Representation (ONNZX-MLIR) compiler, which is fully open-source. For more information, see Installation of ONNX-MLIR on Linux / OSX.

3.6.3 Endianness in saved AI models

Systems access entities larger than a byte in memory in different ways. Little-endian systems (x86, ARM, IBM PowerPC® Linux, and more) store the least significant byte at the lowest address, while big-endian systems (IBM Z, SPARC, PowerPC AIX®, and more) store the most significant byte first. This has an impact when systems with different endianness talk to each other or share files. This also affects AI model storage formats.

While formats designed for portability, like ONNX, do handle endianness correctly, the saved model formats of TensorFlow and PyTorch did not support this initially.

IBM worked with the TensorFlow and PyTorch communities to get the proper support added. However, care must be taken when migrating models from older framework versions.

TensorFlow

In versions prior to 2.5.0, TensorFlow models always utilized the host endian format, which relied on the system's endianness (little-endian on x86 and big-endian on IBM Z). This limitation prevented migrating models between different systems with different endianness.

With TensorFlow 2.5.0 and above, models are consistently saved in the little-endian format. Consequently, when loading a model on an IBM Z system, TensorFlow applies byte swap operations to convert the input into big-endian values. This change enables model portability across various systems, including x86 and IBM Z.

It is strongly advised to utilize TensorFlow 2.5.0 or higher to avoid potential issues during model loading. Although rare, compatibility issues may still arise when loading models from older TensorFlow versions on IBM Z. In such cases, a simple conversion script can be employed to resolve the issue.

Table 3-1 assumes that the model is saved in the TensorFlow 2 saved model format.

For TensorFlow 1 models:

1. Load the model into TensorFlow 2 on little-endian.
2. Save the model as TensorFlow 2 saved model.

Load it into TensorFlow 2 on IBM Z.

Table 3-1 Loading TensorFlow 2 on IBM Z

	Saved on			
	< 2.5.0		**>= 2.5.0**	
Load on	IBM Z	x86	IBM Z	x86
IBM Z < 2.5.0	✓	X*	X*	X*
IBM Z >= 2.5.0	X*	✓	✓	✓

* Use our Endian_converter.

PyTorch

After 2.1.0, TensorFlow has been enhanced to include a marker, which indicates the endianness used when saving the model. With that the model can be converted to host endianness at load-time. For models that have been saved before 2.1.0 no marker is included. These models are assumed to be little-endian. By employing this convention, we can successfully load PyTorch AI models on IBM Z that were saved with older PyTorch versions on x86 systems. This is a typical use case for models provided by the Hugging Face model repository.

Table 3-2

	Saved on			
	< 2.1.0		**>= 2.1.0**	
Load on	IBM Z	x86	IBM Z	x86
IBM Z < 2.5.0	✓	X*	✓	X*
IBM Z >= 2.5.0	X**	✓	✓	✓

**Set default load endianness to big using the following command:

```
from torch.serialization import LoadEndianness
torch.serialization.set_default_load_endianness(LoadEndianness.BIG)
```

Load and save with newer PyTorch version on IBM Z first, this will make sure to add the little/big endian marker into the file.

3.7 Packages with IBM Z optimizations

The following math libraries are packages with IBM Z optimizations:

- OpenBLAS (Basic Linear Algebra Subprograms):
  - A package delivered with the Linux distribution or Anaconda
  - SIMD (Single Instruction, Multiple Data) optimizations since 0.3.10
  - A base math algebra library with a very-fast matrix multiplication implementation
  - Used by popular data science libraries such as NumPy, SciPy, and Pandas
- Eigen
  - A CPU Compute back end for TensorFlow
  - SIMD optimization built into the TensorFlow core
  - Consumed by TensorFlow at build-time, no separate installation required
- NumPy
  - A standard Python package for tensor operations, commonly used for data preparation and simple machine learning use cases
  - Installable through pip or conda
  - Uses the matmul function from the OpenBLAS library, taking advantage of its SIMD optimizations
  - Optimized for the IBM Z SIMD instruction set since 2.42.3
- Sleef
  - A vectorized math library used, such as PyTorch
- oneDNN
  - SIMD optimized for IBM Z

4

AI case studies for Linux workloads on IBM Z

In this chapter we introduce an overview of case studies for AI on IBM Z. We review types of learning (supervised, unsupervised, and reinforcement), natural language processing, computer vision, time series analysis, batch processing with short batch window, security, and data gravity. You will find information about how different types of AI can be used for enriching Linux workloads on IBM Z, especially by using the AI Toolkit for IBM Z.

4.1 AI case studies

IBM Z systems run critical workloads in industries like finance, healthcare, government, and more. Known for robustness, reliability, and security, the IBM Z platform is the ideal environment for infusing AI into mission critical workloads to support your organization's business goals. This section provides an overview of how different types of AI and ML can be applied to enhance industry-specific use cases.

4.1.1 Supervised learning

You can apply supervised AI and ML techniques to various workloads running on IBM Z systems in a range of use cases. Some examples of these use cases include:

- Security and anomaly detection

  IBM Z mainframes are often used for secure, mission-critical workloads. Supervised ML models can be used to detect security anomalies, such as unauthorized access attempts, data breaches, or abnormal system behavior.

- Resource optimization

  Machine learning algorithms can be applied to analyze system performance and workloads, helping to optimize resource allocation and improve the efficiency of IBM Z systems.

- Predictive maintenance

  Supervised ML can be used to predict when workloads running on the mainframe are likely to fail. This allows for proactive maintenance, reducing downtime for the running workloads.

- Capacity planning

  ML models can forecast future workloads and resource requirements, helping organizations plan for system capacity needs.

- Root cause analysis

  In the event of system failures or performance issues, supervised AI and ML can assist in identifying the root causes and recommending solutions.

- User behavior analysis

  Understanding how users interact with mainframe systems can help improve user experience and security. ML models can analyze user behavior patterns.

- Fault tolerance

  Machine learning can be used to create self-healing workloads, where the running workloads can identify issues and take corrective actions to keep the workloads running without human intervention.

4.1.2 Unsupervised learning

You can apply unsupervised AI and ML to workloads running on IBM Z systems in a variety of use cases, including the following:

- Security and anomaly detection

  Unsupervised AI and ML can be employed to continuously monitor IBM Z system logs and user behavior on the running workloads to detect anomalies that may indicate security breaches. Clustering algorithms can group similar system activities and similar application-level activities. Any deviation from these clusters can be flagged as an anomaly.

- Performance optimization

  System optimization is crucial. Predictive maintenance and performance tuning are common in mainframe environments. Unsupervised learning can be used to identify patterns in performance data and to suggest optimizations. Clustering and dimensionality reduction techniques can help in this context.

- Data compression and management

  As vast amounts of data can run on mainframes, data compression is essential for efficient storage and data management. Unsupervised learning can be used to identify redundant data and patterns for better compression algorithms.

- Capacity planning

  Ensuring that the mainframe has adequate capacity to manage workloads is crucial. Under- or over-provisioning can lead to unnecessary costs. An effective approach to managing capacity is to utilize unsupervised machine learning techniques, which can analyze historical workload data and uncover usage patterns.

- Data classification and tagging

  Data on mainframes may need to be classified or tagged for compliance and data management purposes. Unsupervised learning can help automatically classify data by analyzing its content and usage patterns, making data management more efficient.

- Root cause analysis

  When issues occur, identifying the root cause is essential for quick resolution. Unsupervised learning can help identify patterns in system logs or events leading up to issues, making it easier to pinpoint root causes.

- Capacity and workload balancing

  Balancing workloads across multiple mainframes is essential to optimize resource utilization. Unsupervised learning can help analyze the workload distribution and suggest ways to optimize resource allocation.

4.1.3 Reinforcement learning

You can apply reinforcement learning to many use cases on IBM Z through the following steps:

1. Data collection

   Gather historical data related to the specific use case. This data will be used to train and test reinforcement learning models.

2. Model development

   Develop a reinforcement learning model tailored to the use case. Frameworks like TensorFlow or PyTorch can be used model development.

3. Training

   Train the model using the historical data. Reinforcement learning typically involves trial and error, so the model learns through interactions with the environment.

4. Testing

   Evaluate the model's performance through testing. Fine-tune it based on testing results.

5. Deployment

   Deploy the reinforcement learning model in the IBM Z environment. Ensure that it integrates seamlessly with existing processes.

6. Monitoring

   Continuously monitor the model's performance and make necessary adjustments to ensure it meets the desired objectives.

7. Feedback loop

   Implement a feedback loop to capture real-time data and adjust the model's behavior as conditions change.

Additionally, reinforcement learning can be applied to the following use cases:

- Secure transactions

  IBM Z systems are known for their robust security features. Customers use them for processing secure transactions, such as financial transactions, healthcare record management, and more. Reinforcement learning can focus on enhancing security measures, automating threat detection, and ensuring regulatory compliance.

- Batch data processing

  IBM Z can be relied on for batch data processing, such as payroll processing, inventory management, and report generation. Implement reinforcement learning to optimize batch processing schedules, improve resource allocation, and predict processing times more accurately.

- Real-time analytics

  IBM Z systems are used for real-time analytics in industries like finance and telecommunications. In this case, reinforcement learning can help in real-time decision-making, fraud detection, network optimization, and personalized customer services.

- Hybrid cloud integration

  IBM Z can be integrated with hybrid cloud environments. Reinforcement learning can assist in optimizing workload distribution between on-premises and cloud-based resources, ensuring cost efficiency and high performance.

4.1.4 Natural language processing

You can apply natural language processing (NLP) to workloads running on IBM Z systems for many use cases, including log analysis and building virtual assistants for Linux environments.

NLP for Linux log analysis

NLP can be used in the following ways for Linux log analysis:

- Log parsing

  NLP can be used to extract meaningful information from unstructured log data. NLP models can help identify patterns, anomalies, and critical events within logs.

- Anomaly detection

  NLP can be used to identify anomalies in log files. By training NLP models on normal log data, you can detect deviations from the norm, which may indicate security breaches or system issues.

- Root cause analysis

  NLP can help in pinpointing the root causes of issues by analyzing logs for error messages and patterns. This can be particularly useful for troubleshooting and system maintenance.

- Automated alerts

  NLP models can be employed to trigger alerts or notifications when specific conditions or events are detected in log data.

Building chatbots or virtual assistants for Linux environments

Additionally, you can use NLP to build chatbots and virtual assistants for Linux environments:

- Command line interaction

  Create chatbots or virtual assistants that allow users to interact with Linux systems through natural language commands. These bots can interpret user input, execute shell commands, and provide responses or take actions based on the user's requests.

- Monitoring and reporting

  Chatbots can be used to monitor system health, resource utilization, and log data. Users can ask questions or request status updates, and the chatbot can provide real-time information.

- Troubleshooting and support

  Virtual assistants can assist users in troubleshooting common Linux issues by providing step-by-step guidance, running diagnostic commands, and offering solutions to common problems.

- Security and access control

  Chatbots can help enforce security policies by managing user access, permissions, and system configurations through natural language interaction.

4.1.5 Computer vision

You can apply computer vision to workloads running on IBM Z systems in different industries and business use cases, including the following:

- Industrial automation

  In manufacturing and industrial environments, IBM Z can serve as a powerful data processing and analytics platform to support the massive data generated by computer vision systems. It can process data from cameras and sensors, enabling real-time quality control, predictive maintenance, and automation of production processes.

- Surveillance and security

  For surveillance and security applications, IBM Z provides a secure and highly available platform for managing and analyzing video feeds from multiple cameras. It can process and store vast amounts of video data, run analytics for object detection and behavior analysis, and trigger alerts when suspicious activities are detected.

- Healthcare

  In healthcare, IBM Z can be used for secure and compliant storage of medical images and data. Computer vision algorithms can be executed on IBM Z to process and analyze medical images, helping radiologists and healthcare professionals in diagnosis and treatment planning.

- Retail

  IBM Z is well-suited for managing the complex data processing needs of retail operations. It can handle inventory management, pricing optimization, and sales data, while computer vision systems can track inventory levels, optimize shelf stocking, and enhance the checkout process.

- Transportation and traffic management

  IBM Z can support transportation and traffic management systems by processing vast amounts of data from cameras, sensors, and traffic control systems. It can provide real-time analytics for traffic monitoring, automated toll collection, and traffic flow optimization.

- Autonomous vehicles

  Autonomous vehicles generate enormous volumes of sensor and camera data. IBM Z can be used to process this data for object detection, route planning, and decision-making. It ensures the safety, reliability, and security of autonomous vehicle operations.

- Agriculture

  IBM Z can assist in managing data from various sensors and cameras used to monitor crop health, assess yields, and automate agricultural processes. It can process this data to optimize irrigation, fertilization, and harvesting.

- Optical character recognition (OCR)

  IBM Z can provide the computing power needed for OCR applications. It can handle the processing of large volumes of scanned documents and images, converting text into machine-readable data for various purposes like document management and data extraction.

4.1.6 Time series analysis

Time series analysis is a statistical technique that involves analyzing data points collected or recorded over a specific time interval. When applied to IBM Z systems, it can provide valuable insights in various use cases including log analysis and predictive maintenance.

In the context of IBM Z systems, time series analysis finds applications in various scenarios, providing actionable insights and foresight into system behaviors. The following examples describe how time series analysis can be employed on IBM Z, particularly with the advanced IBM z16 architecture:

- Performance monitoring

  Time series analysis is invaluable for monitoring the performance of IBM Z mainframes over time. By analyzing metrics such as CPU utilization, memory usage, and I/O operations, organizations can identify trends and patterns that help optimize resource allocation, ensuring efficient workload processing.

- Log analysis

  Analyzing log data over time is crucial for detecting irregularities and potential security threats. Time series analysis enables organizations to track changes in user access patterns, system log activities, and network traffic, enhancing the ability to identify and respond to security incidents promptly.

- Predictive maintenance

  Utilizing historical data points, time series analysis facilitates the prediction of hardware failures or system issues. By monitoring metrics related to disk health, processor performance, and other critical components, organizations can proactively address potential issues before they lead to system downtime. For example, in an environment with complex machinery (such as manufacturing), time series data related to equipment performance can be collected and stored on IBM Z. Time series analysis can then help predict maintenance requirements by identifying patterns and deviations in the machine data. By monitoring factors like temperature, pressure, and vibration, over time, predictive maintenance models can anticipate when equipment might fail, reducing downtime and maintenance costs.

- Capacity planning

  Time series analysis aids in capacity planning by examining resource utilization patterns over time. This information is essential for making informed decisions about scaling hardware resources, ensuring that the IBM Z system can accommodate growing workloads without compromising performance.

- Workload optimization

  Organizations can leverage time series analysis to understand the dynamics of different workloads running on IBM Z. By analyzing performance metrics specific to each workload over time, administrators can optimize configurations and resource allocations to achieve better overall system efficiency.

- Fault detection and isolation

  Time series analysis can assist in identifying and isolating faults within the IBM Z infrastructure. By examining patterns in error logs, administrators can pinpoint the root causes of issues, streamline troubleshooting processes, and minimize the impact of faults on system performance.

- Resource usage forecasting

  Predicting future resource requirements is essential for planning and optimizing IBM Z system resources. Time series analysis allows organizations to forecast trends in resource usage, helping them allocate resources effectively and prevent potential bottlenecks during peak usage periods.

- Transaction monitoring

  Time series analysis helps banks monitor transaction patterns over time. Unusual spikes in transaction volumes, unexpected patterns in fund transfers, or irregularities in withdrawal amounts can be identified through time series analysis. This enables banks to promptly detect and respond to potential fraudulent activities, enhancing overall security.

- Interest rate and stock exchange forecasting

  Time series analysis is employed to forecast interest rate and stock exchange trends based on historical data. Banks can use this information for strategic decision-making, including setting loan interest rates, managing investment portfolios, and adjusting financial products in response to anticipated market changes.

These examples showcase the versatility of time series analysis on IBM Z, emphasizing its role in enhancing security, predicting maintenance needs, optimizing workloads, and ensuring the overall reliability and efficiency of the system, particularly with the advancements introduced by the IBM z16 architecture.

Analyzing Linux system logs using time series models

Time series analysis can help in monitoring and troubleshooting Linux systems by examining log data over time. It enables the detection of patterns, trends, and anomalies in log entries. For example, it can be used to identify and predict resource usage trends, system performance degradation, and security incidents. By analyzing time series data, you can proactively address issues and optimize system performance. Various techniques can be deployed, including the following:

- Seasonal-Trend Decomposition

  Utilizing decomposition techniques to break down system log data into seasonal, trend, and residual components. This enables a better understanding of periodic patterns and long-term behavior.

- Auto-Regressive Integrated Moving Average (ARIMA)

  Implementing ARIMA models to forecast future system performance based on historical log data. ARIMA models consider patterns in the data to predict potential future trends.

- LSTM (Long Short-Term Memory) Networks

  Deep learning models like LSTMs can capture intricate patterns in the data and are particularly useful for handling sequences of data, making them ideal for time series analysis.

For the various presented techniques, IBM z16 is equipped with various capabilities for data processing and analysis, offering a range of tools for time series analysis on Linux system logs in the IBM z/OS Management Facility (z/OSMF). IBM z/OSMF provides an interface for managing and analyzing system logs, aiding in log extraction and preprocessing. For more information about how IBM z/OSMF can help simplify the day-to-day operations and administration of your mainframe z/OS systems, see IBM z/OS Management Facility.

4.2 Workloads processing and security

In the dynamic landscape of modern computing, the efficient processing of workloads is paramount to the success of any enterprise system. In the context of IBM Z mainframes running Linux workloads, performance optimization, and robust security are pivotal to the overall efficacy of the system. This section discusses key aspects of workload processing and security, shedding light on critical considerations for enhancing the capabilities of Linux on IBM Z.

4.2.1 Batch processing with short batch window

IBM Z is renowned for its batch processing capabilities, making it ideal for organizations that require efficient and high-performance batch processing. It can handle large volumes of data with low latency, ensuring that batch jobs are completed within tight time frames.

As part of its batch processing capabilities, IBM Z supports parallel processing, enabling multiple batch jobs to run concurrently. This is crucial for organizations that have numerous batch jobs to process within short time windows.

The mainframe platform offers high availability and fault tolerance, ensuring that batch jobs are executed reliably and without interruptions.

4.2.2 Security

IBM Z offers a high level of securability to Linux workloads when requests cannot be sent off platform and for the data on IBM Z.

IBM Z is renowned for its robust security features. With secure boot processes[1] enabled, you can cancel loading for components that cannot be verified. Advanced authentication methods help protect data and applications from threats.

Additionally, the platform allows for strong isolation between workloads. This ensures that applications running on the mainframe cannot send requests or access data outside of their designated boundaries, thereby enhancing data security.

By providing secure data handling and access control features, IBM Z can help you meet regulatory compliance requirements, such as GDPR and HIPAA.

4.2.3 Data gravity

For data management IBM Z provides powerful capabilities, enabling organizations to manage and process large volumes of data efficiently.

The platform supports high-capacity data storage solutions, which is essential for data-intensive workloads. Data gravity is addressed and capitalized on by allowing data to be processed close to where it resides.

For data analytics, IBM Z can integrate with analytics tools to help organizations derive insights from their data, making it a suitable platform for addressing data gravity challenges.

[1] https://www.ibm.com/docs/en/linux-on-systems?topic=blilm-secure-boot-2

4.2.4 AI Toolkit for IBM Z

The AI Toolkit for IBM Z, illustrated in Figure 4-1, is a collection of open source AI frameworks with IBM Elite Support, adapted for IBM Z and IBM LinuxONE hardware. The Toolkit consists of IBM Elite Support and IBM Secure Engineering that vet and scan open source AI serving frameworks and IBM certified containers for security vulnerabilities and validate compliance with industry regulations. The introduction of the AI Toolkit provides the level of support needed to successfully implement these popular open source AI frameworks.

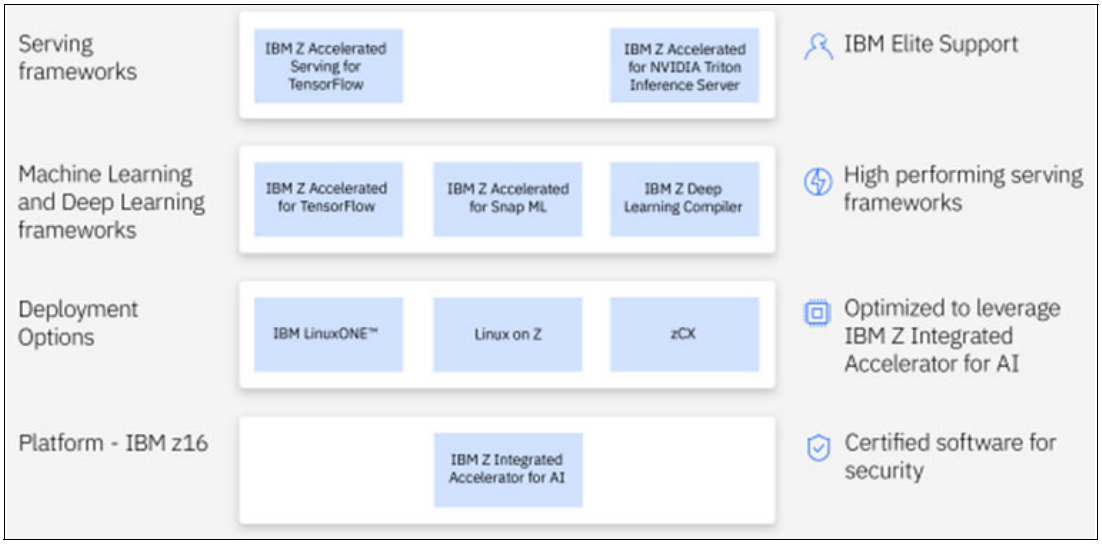

Figure 4-1 AI Toolkit for IBM Z and LinuxONE overview

The AI Toolkit for IBM Z and LinuxONE is discussed in more detail, including practical implementation, in 5.5.1, “AI Toolkit for IBM Z and IBM LinuxONE” on page 57.

For specific details about the different levels of IBM Selected Support, including IBM Elite Support, see IBM Selected Support.

4.3 Success story

As an example, a Swiss insurance and pension company depends on IBM Z to process high volumes of transactions and provide a secure environment for their most sensitive data. By leveraging SQL Data Insights on IBM z16 they were able to uncover patterns in their mission-critical data to improve their insurance offer predictions and reduce cost. They worked with IBM to implement NLP-based AI functions in near real-time, ensuring privacy and security with 94% accuracy in prediction results.[2]

[2] https://www.ibm.com/blog/announcement/ibm-accelerates-enterprise-ai-for-clients-with-new-watsonx-capabilities-on-ibm-z/#_ftn1

5

AI Model inferencing on IBM Z

In the previous chapters, we have discussed the IBM Z platform's functional capabilities in supervised, unsupervised, and deep learning. Now, let's explore the non-functional aspects of the IBM Z platform for AI model inferencing.

In this chapter, we will cover the performance and environmental benefits of co-locating AI inferencing on IBM Z with Online Transaction Processing (OLTP), use cases, the IBM technologies behind the benefits, implementation options, and performance tuning.

5.1 Performance and environmental benefits of co-locating AI inferencing on IBM Z with OLTP

Typical architecture uses a separate process or Virtual Machine (VM) for AI inferencing. The AI service consumer or client running in a different process invokes the inferencing servers through a network protocol such as https and web services. Network latency is unavoidable.

For production deployments, IBM Z and IBM LinuxONE enable clients to co-locate crucial business processes and data without replicating data to other environments for analysis.

IBM is developing optimizations in software and hardware to meet low latency requirements and to enable clients to integrate AI with IBM Z data and core business applications on IBM Z. These technologies help clients embed AI in their applications with minimal application changes. Target use cases include time-sensitive applications with high transaction volumes. In these transactional use cases, the main objective is to reduce latency for faster response time, delivering inference results back to the caller at high volume and speed. One example is fraud detection; for banks and financial markets, accurate detection of fraud during the transaction can result in significant savings.

The following examples illustrate the benefits of reducing network traffic, latency, and energy consumption[1]:

- Using one IBM z16 Integrated Accelerator for AI on an OLTP workload on IBM z16 matches the throughput of a compared remote x86 server running inferencing on 18 cores (see “Use case 1: Core count requirement for IBM Z in-transaction inferencing versus x86”).
- On IBM z16 utilizing the IBM z16 Integrated Accelerator for AI, run an OLTP workload with in-transaction inference operations with 3x lower response time versus remotely running inferencing on a compared x86 server leveraging a NVIDIA GPU (see “Use case 2: IBM Z in-transaction inferencing response time versus remote inferencing in x86 + GPU”).
- On IBM z16, reduce the energy consumption by 41x using the IBM z16 Integrated Accelerator for AI to process inference operations of an OLTP workload versus running inference operations remotely on a compared x86 server using an NVIDIA GPU (see “Use case 3: Energy consumption of IBM Z in-transaction inferencing versus remote inferencing in x86 + GPU”).
- Run prediction of customer transactions 3.5x faster by co-locating your application with the Snap ML library on IBM LinuxONE Emperor 4 versus running prediction remotely using the NVIDIA Forest Inference Library on a compared x86 server (see “Use case 4: Comparing Random Forest classifier of IBM Z in-transaction inferencing response time versus remote inferencing in x86 + GPU”).

[1] Based on IBM internal measurements. Results may vary by customer based on individual workload, configuration, and software levels.

5.2 Use case 1: Core count requirement for IBM Z in-transaction inferencing versus x86

The first use case we will discuss is an in-transaction inferencing deployment option of a fraud detection model with an OLTP application.

In this example, the fraud detection model is an LSTM model trained with 2.4 M samples of customer transactions, with 1.22% labeled as fraudulent. Given the customer's 6 previous transactions, this model was trained to detect whether the current transaction is likely fraudulent.

The first performance comparison using the setup illustrated in Figure 5-1 on page 46 is targeted to demonstrate the capabilities of the IBM z16 Integrated Accelerator for AI against Intel x86 cores.

For our example, we show that using 1 IBM z16 Integrated Accelerator for AI on an OLTP workload on IBM z16 matches the throughput of a comparable remote x86 server running inferencing on 18 cores.

DISCLAIMER: Performance results extrapolated from IBM internal tests running an OTLP workload with credit card transaction using the Credit Card Fraud Detection model on IBM z16 versus running the OLTP workload on IBM z16 and running inferencing on a remote x86 server running Tensorflow Serving.

IBM z16 configuration: Ubuntu 20.04 in an LPAR with 6 dedicated IFLs, 256 GB memory, and IBM FlashSystem 900 storage.

x86 configuration: Ubuntu 20.04 on 18 IceLake Intel Xeon Gold CPU @ 2.80GHz with Hyperthreading turned on, 1 TB memory and local SSDs.

Results may vary.

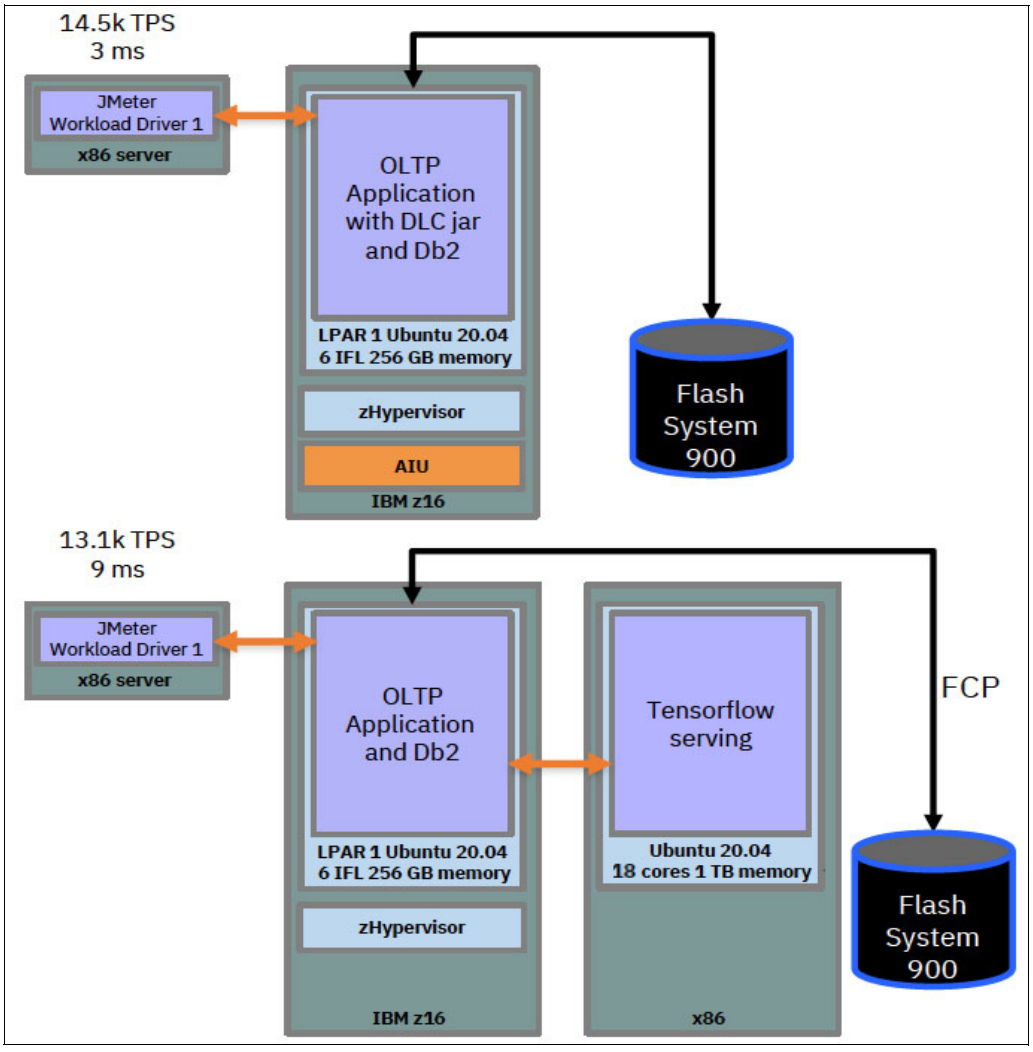

Figure 5-1 OLTP with fraud detection and IBM Db2 on IBM z16 versus IBM z16 and x86

5.2.1 Use case 2: IBM Z in-transaction inferencing response time versus remote inferencing in x86 + GPU

The second use case we will discuss is another in-transaction inferencing deployment option of the same fraud detection model as use case 1; however, this time the performance comparison will be the response times between IBM z16 and x86 + GPU.

The setup for this second performance comparison, illustrated in Figure 5-2, is targeted to demonstrate the capabilities of the IBM z16 Integrated Accelerator for AI against an NVIDIA A40 GPU.

For our example, we show the IBM z16 utilizing the IBM z16 Integrated Accelerator for AI, running an OLTP workload with in-transaction inference operations with 3x lower response time versus remotely running inferencing on a compared x86 server leveraging a NVIDIA GPU.

DISCLAIMER: Performance results are extrapolated from IBM internal tests running an OTLP workload with credit card transaction using the Credit Card Fraud Detection model on IBM z16 versus running the OLTP workload on IBM z16 and running inferencing on a remote x86 server running Tensorflow Serving.

IBM z16 configuration: Ubuntu 20.04 in an LPAR with 6 dedicated IFLs, 256 GB memory, and IBM FlashSystem 900 storage.

x86 configuration: Ubuntu 20.04 on 24 IceLake Intel Xeon Gold CPU @ 2.80GHz with Hyperthreading turned on, 1 TB memory, local SSDs, and NVIDIA A40 GPU.

Results may vary.

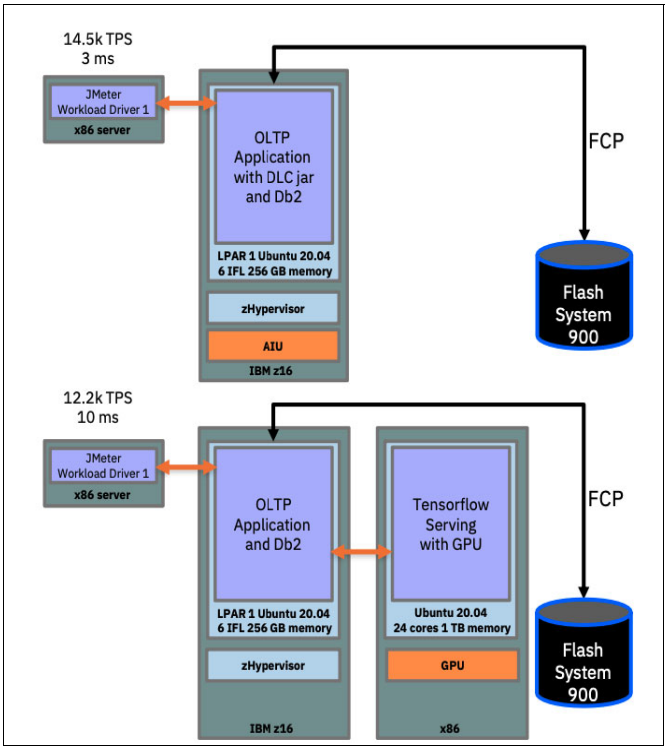

Figure 5-2 OLTP with fraud detection and IBM Db2 on IBM z16 versus IBM z16 and x86 with GPU

5.2.2 Use case 3: Energy consumption of IBM Z in-transaction inferencing versus remote inferencing in x86 + GPU

Our third performance comparison example using the setup illustrated in Figure 5-3 is targeted to demonstrate the power efficiency of the IBM z16 Integrated Accelerator for AI against an NVIDIA A40 GPU. Our example shows that IBM z16 reduces the energy consumption by 41x using the IBM z16 Integrated Accelerator for AI to process inference operations of an OLTP workload versus running inference operations remotely on a compared x86 server using an NVIDIA GPU.

DISCLAIMER: Results based on IBM internal tests running an OTLP workload with credit card transaction using the Credit Card Fraud Detection model on IBM z16 using the Integrated Accelerator for AI to process inference operations versus running the OLTP workload on IBM z16 with remote inferencing on a x86 server running Tensorflow serving.

IBM z16 configuration: Ubuntu 20.04 in an LPAR with 6 dedicated cores, 256 GB memory, and IBM FlashSystem 9200 storage.

x86 configuration: Ubuntu 22.04 on 2x 24 IceLake Intel Xeon Gold 6342 CPU @ 2.80GHz with Hyperthreading turned on, 1 TB memory, local SSDs, NVIDIA A40 GPU, UEFI maximum performance profile enabled, CPU P-State Control and C-States disabled.

Results may vary.

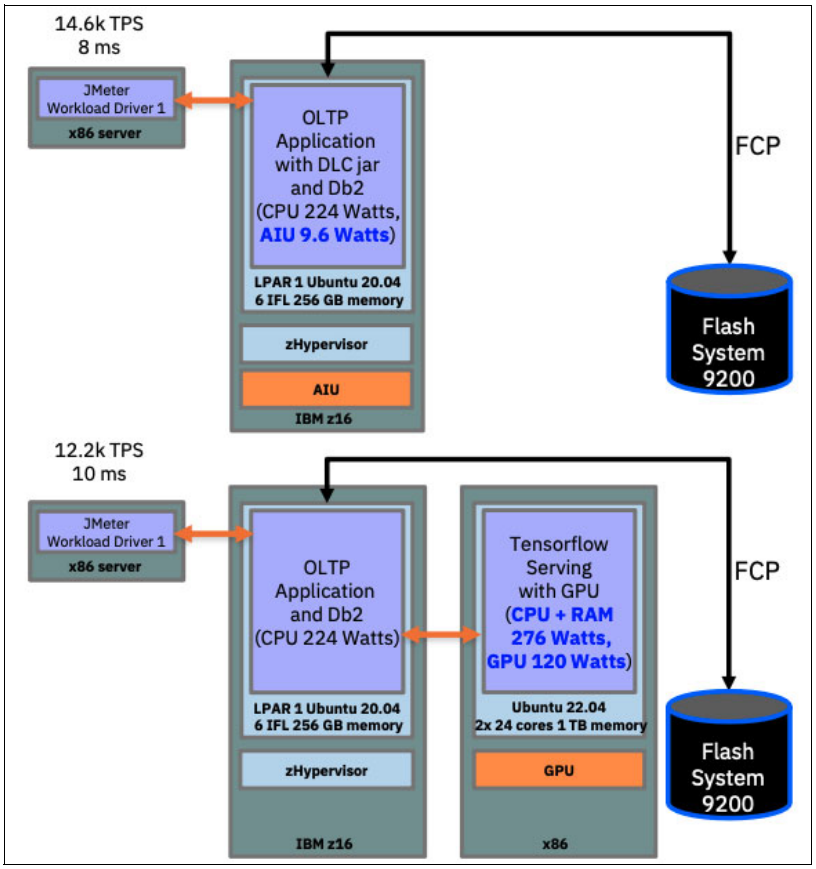

Figure 5-3 Comparing energy consumption of in-transaction inferencing of IBM z16 versus remote inferencing on x86 + GPU

5.2.3 Use case 4: Comparing Random Forest classifier of IBM Z in-transaction inferencing response time versus remote inferencing in x86 + GPU

Moving away from the in-transaction inferencing of our first examples, our next example utilizes a typical deployment option: inference in a separate process or container, by deploying a Snap ML Random Forest classifier model on NVIDIA Triton Inference Server.

In this example, the Random Forest model is trained based on the scikit-learn breast cancer dataset. For more information, see Serving SnapML models with Triton Inference Server on Linux on IBM zSystems.

Our comparison example using the setup illustrated in Figure 5-4 on page 50 is targeted to demonstrate the capabilities of the IBM z16 Integrated Accelerator for AI for a random forest use case in the financial domain. The setup uses IBM z16 using NVIDIA Triton, a serving infrastructure with IBM Snap ML as a compute back end.

On the compared x86 server, we also use NVIDIA Triton with the Forest Inference Library as the compute back end.

For our example, we demonstrate that running prediction of customer transactions is 3.5x faster by co-locating the OLTP application with the Snap ML library on IBM z16 A01 versus running prediction remotely using the NVIDIA Forest Inference Library on a compared x86 server.

DISCLAIMER: Performance results based on IBM internal tests doing inferencing using a Random Forest model with Snap ML v1.12.0 back end which uses the Integrated Accelerator for AI on IBM Machine Type 3931 versus the NVIDIA Forest Inference Library back end on compared x86 server. The model was trained on the following public dataset and NVIDIA Triton was used on both platforms as model serving framework. The workload was driven by the http benchmarking tool Hey.

- IBM Machine Type 3931 configuration: Ubuntu 22.04 in an LPAR with 6 dedicated IFLs, 256 GB memory.
- x86 configuration: Ubuntu 22.04 on 6 Ice Lake Intel Xeon Gold CPU @ 2.80GHz with Hyper-Threading turned on, 1 TB memory.

Results may vary.

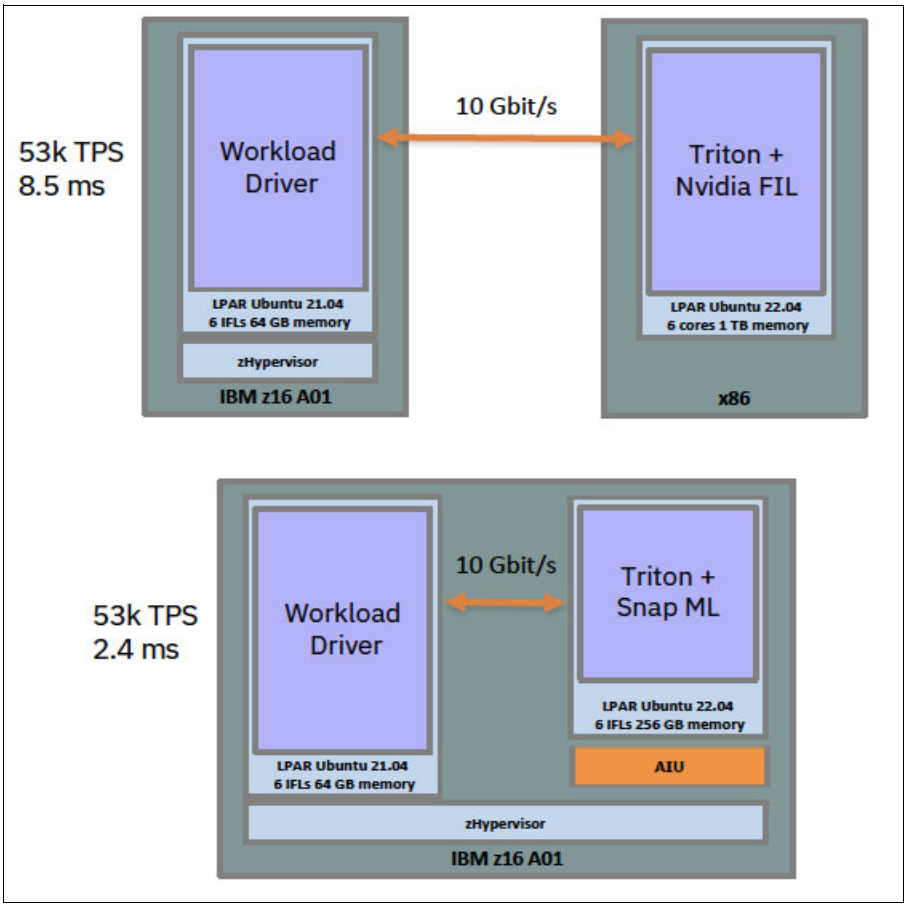

Figure 5-4 Compare Triton with Snap ML co-location on IBM LinuxONE Emperor 4 versus x86 with Forest Inference Library

5.3 The technologies behind the high performance of IBM Z

With the introduction of the IBM z16 Integrated Accelerator for AI, organizations can scale their inferencing of transactions running on IBM z16 and IBM LinuxONE Emperor 4 while continuing to meet latency and throughput requirements. AI can be directly embedded into mission-critical workloads to gain real-time insights.

AI functions, or macros, are more than just matrix multiplication, they are abstracted using NNPA instructions (see Table 5-1 on page 51).

Table 5-1 Functions of IBM z16 Integrated Accelerator for AI on IBM z16 and IBM LinuxONE Emperor 4

Function group	#	Function support in GA1
Elementwise ops	0x10	NNPA_EL_ADD
	0x11	NNPA_EL_SUB
	0x12	NNPA_EL_MUL
	0x13	NNPA_EL_DIV
	0x14	NNPA_EL_MIN
	0x15	NNPA_EL_MAX
Activation ops	0x20	NNPA_LOG
	0x21	NNPA_EXP
	0x31	NNPA_RELU
	0x32	NNPA_TANH
	0x33	NNPA_SIGMOID
Norm ops	0x34	NNPA_SOFTMAX
	0x40	NNPA_BATCHNORM
Pooling	0x50	NNPA_AVGPOOL2D
	0x51	NNPA_MAXPOOL2D
Systolic ops	0x70	NNPA_CONVOLUTION
	0x71	NNPA_MATMUL_OP
	0x72	NNPA_MATMUL_OP_BCAST23
RNN	0x60	NNPA_LSTMACT
	0x61	NNPA_GRUACT
	0X00	NNPA_QAF

The following section introduces how we can leverage the IBM z16 Integrated Accelerator for AI. There are generally 2 different approaches: Integrating the exploitation of the IBM z16 Integrated Accelerator for AI directly into your application using the IBM Z Deep Learning Compiler or using a popular framework or library that is either locally called by your application or used as a compute back end that is leveraged through an inference serving instance.

5.3.1 Single-instruction, multiple-data on IBM z16 Telum processor

IBM z16 includes a set of instructions called single-instruction, multiple-data (SIMD) that can improve the performance of complex mathematical models and analytics workloads. This improvement is realized through vector processing and complex instructions that can process a large volume of data by using a single instruction. SIMD is designed for parallel computing and can accelerate code that contains integer, string, character, and floating-point data types. This system enables better consolidation of analytics workloads and business transactions on the IBM Z platform.

Popular ML models using Scikit-learn, such as Linear Regression, Random Forest, and Boosting Machine could leverage SIMD through the IBM Snap ML library.

5.3.2 IBM Z Deep Learning Compiler

IBM is architecting solutions to enable model portability to IBM Z and IBM LinuxONE without requiring additional development efforts for deployment. For instance, ONNX technology, a standard format for representing AI models, allows data scientists to build and train a model in their framework of choice without worrying about the downstream inference implications. To enable the deployment of ONNX models, IBM has provided an ONNX model compiler that is specifically optimized for IBM Z and IBM LinuxONE.

IBM Z Deep Learning Compiler (zDLC) is a prebuilt s390x container; it compiles ONNX deep learning AI models into shared libraries. This lightweight shared object library has no dependencies on the framework or libraries in which the model was developed and trained. We can easily use it for inference from C++, Java, or Python programs directly.

The compiled models perform better than the Python-interpreted models and they exploit IBM Z technologies, including SIMD on IBM z13® and later and the IBM z16 Integrated Accelerator for AI available on IBM z16 without changes to the original model (see Figure 5-5).

ONNX is an open format for representing AI models. It is open-source and vendor-neutral. Some AI frameworks, such as PyTorch, directly support exporting to `.onnx` format. For other frameworks, open-source converters are readily available. For more information, see ONNX Supported Tools.

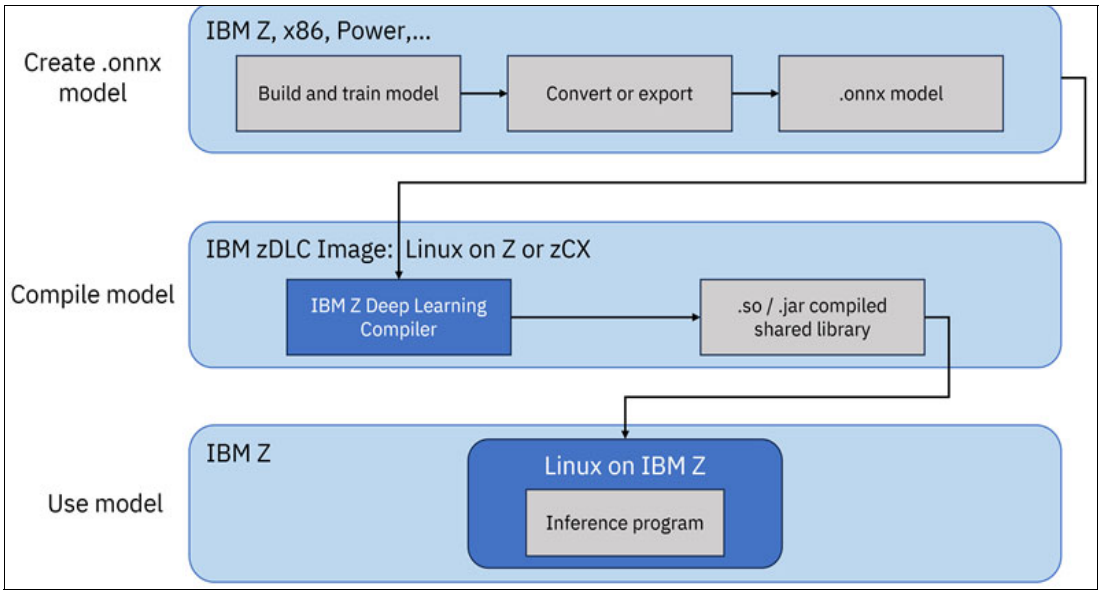

Figure 5-5 IBM Z Deep Learning Compiler workflow

To convert or export models in ONNX format, TensorFlow models can be converted using the following open-source tensorflow-onnx package:

```
python -m tf2onnx.convert --saved-model tensorflow-model-path --opset 19
--output model.onnx
```

PyTorch models can be exported directly to ONNX using `torch.onnx.export()` PyTorch APIs

Best practices

To avoid endian issues, we strongly recommend converting models to the ONNX format on the platform on which they were trained. For example, if a model was trained on an x86 environment, convert the model to ONNX on an x86 environment before transmitting it to IBM Z or IBM LinuxONE.

For zDLC usage

To determine which ONNX opset to specify for model conversion, see the following references:

- NNPA supported opset level
- CPU supported opset level

On IBM z16 or IBM LinuxONE 4 machines, we recommend using the highest opset level specified under NNPA operator support. This will generally be the first statement in the `SupportedONNXOps-NNPA.md` file linked above (see Figure 5-6).

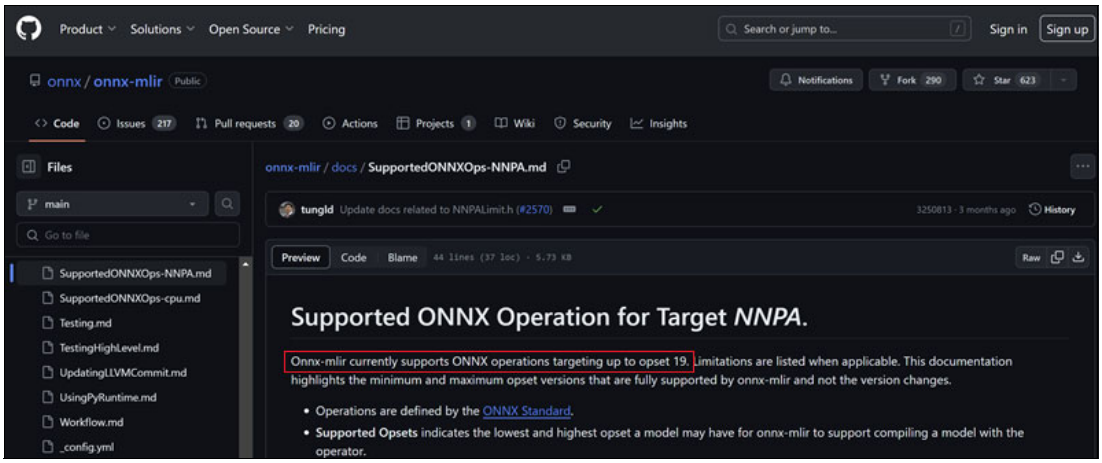

Figure 5-6 Highest opset supported by NNPA operator

5.3.3 IBM-zDNN-plug-in to TensorFlow

For models using TensorFlow, in addition to exporting them in ONNX format and compiling them with IBM Z Deep Learning Compiler, another option is to install the IBM-zDNN-plug-in, which leverages the IBM z Deep Neural Network, to TensorFlow in IBM Z and import the pre-trained TensorFlow models. It is possible to use the proto buffer format of TensorFlow and the open-source tool to convert the model from x86 to the s390x platform.[2]

[2] https://github.com/ambitus/zos-native/blob/master/tools/Endian_converter/pb_endian_convert_utility.py

This capability allows TensorFlow (and TensorFlow Serving) to target the IBM z16 accelerator seamlessly and transparently for compute-intensive operations (see Figure 5-7).

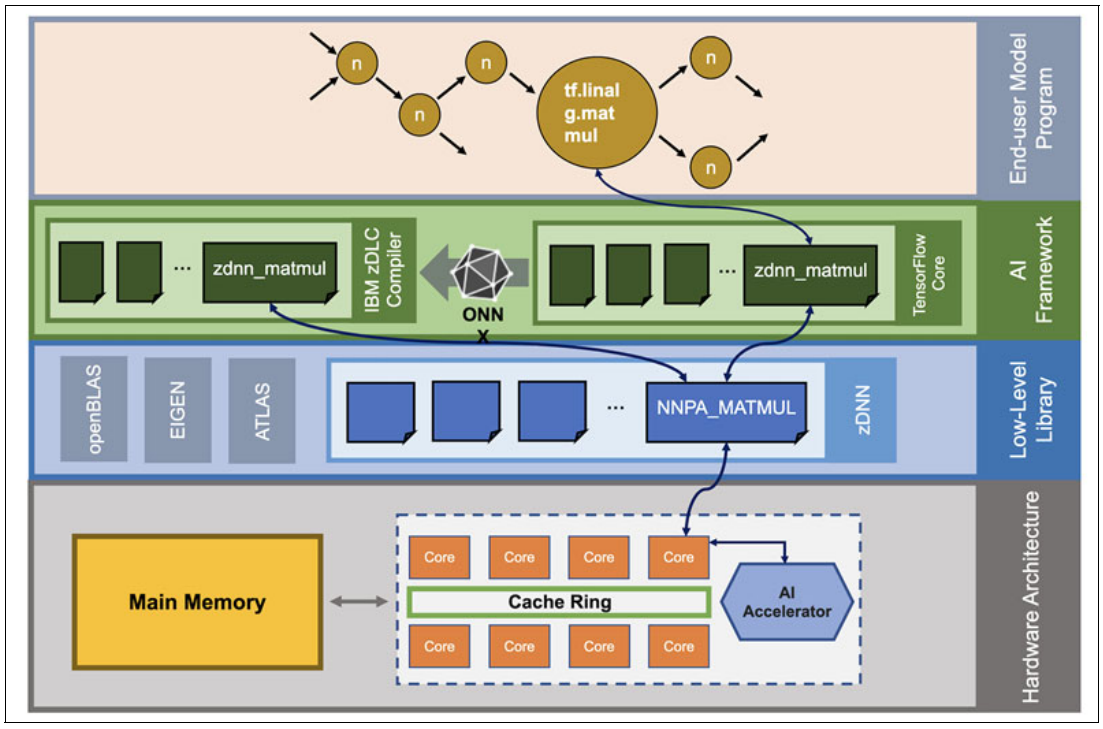

Figure 5-7 Two code paths to accelerate TensorFlow workload

The IBM-zdnn-plug-in for TensorFlow provides a more seamless integration with TensorFlow, while the export of the model into ONNX and compilation with the IBM zDLC toolkit offers additional optimization capabilities through the use of a high-performance compiler. The choice between these methods will depend on your specific use case and requirements.

5.3.4 IBM Snap ML

Snap ML is a library for training and scoring traditional ML models. It uses multi-threading to take advantage of multi-core, multi-socket CPUs for linear models. It has a memory-efficient breadth-first search algorithm for training decision trees, random forests, and gradient-boosting machines. Over 10x acceleration could be obtained, depending on the model and dataset.

To leverage the accelerated inference engine on IBM Z, the models trained with frameworks such as scikit-learn, XGBoost, and LightGBM can be saved in PMML format and imported into Snap ML for inferencing as shown in Figure 5-8 on page 55.

For more information, see Snap ML Model Import.

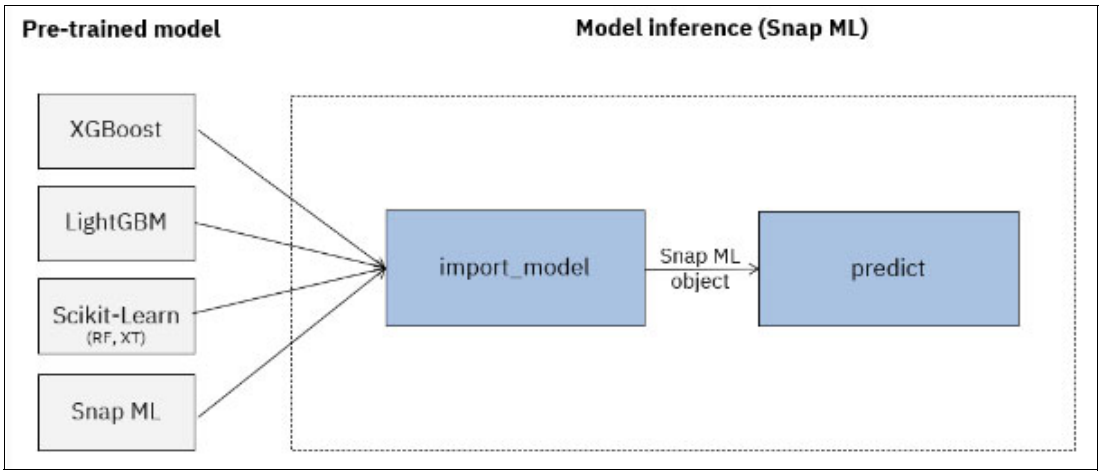

Figure 5-8 Importing traditional ML models for inferencing by Snap ML

5.4 Leveraging the IBM z16 Integrated Accelerator for AI in model inferencing server

In the previous section, we highlighted the advantages of in-transaction inferencing and the enabling technologies. There are options to deploy the models in separate inference servers/containers and they can be invoked using networking protocols (see Figure 5-9 and Table 5-2 on page 56). Although there may be some network latency, these models can still exploit the Integrated Accelerator in IBM Z.

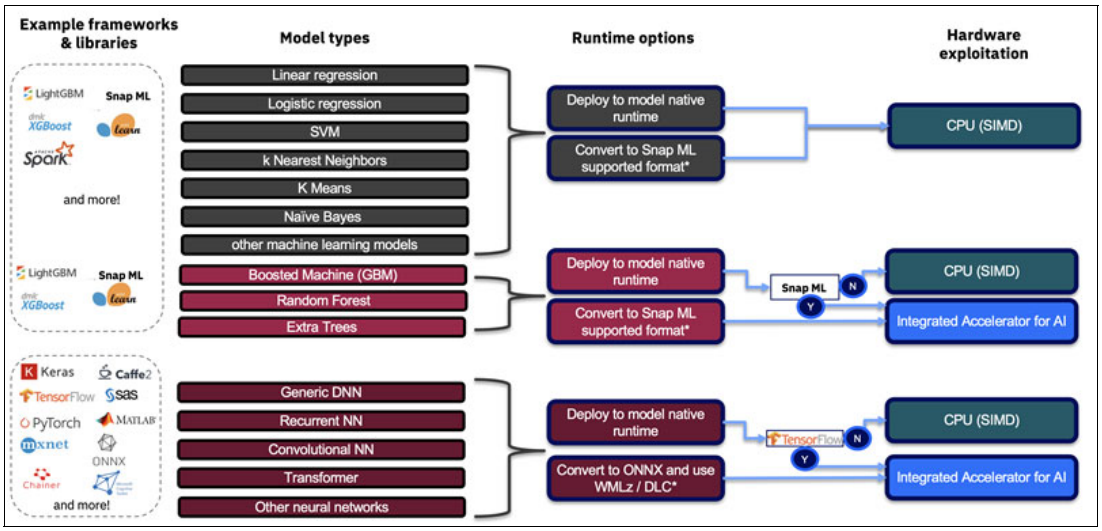

Figure 5-9 Model optimization on IBM z16 Processor

Table 5-2 Inferencing options on IBM Z Summary

Option	Inferencing option	Models	Relative performance	Deployment method
1	In Transaction Inference	Deep learning frameworks, for example, TensorFlow or Pytorch	High	Export the model in ONNX format. Compile it with the IBM Z Deep Learning Compiler into SO format and optionally a Java or Python wrapper, which can be invoked by the transaction directly.
2	Inference in separate Tensorflow Serving container or process	TensorFlow	Mid-High	Install Tensorflow Serving and TensorFlow from the IBM Cloud Container Registry (ICR) on IBM Z. Import the TensorFlow model for inferencing.
3	Inference in separate PyTorch container or process	PyTorch	Mid-High	Install Pytorch from Anaconda for IBM Z. Import the pre-trained PyTorch model and build a container for inferencing. Or, install NVIDIA Triton Inference Server from ICR and deploy the PyTorch model in it.
4	Inference in separate Python-SnapML container or process	Traditional tree based or boosting based ML models: GBM or Random Forest	High	Install Python and SnapML from ICR on IBM Z. Import the model in Predictive Model Markup Language (PMML) format. SnapML will exploit IBM z16 Integrated Accelerator for AI. Or, install NVIDIA Triton Inference Server from ICR and deploy the model in it.
5	Inference in separate Python-SnapML container or process	Traditional machine learning models: Linear regression, Logistic regression, SVM, KNN, K Means, or Naive Bayes	Mid-High	Install Python and SnapML from ICR on IBM Z. Import the model in PMML for inference. SnapML will exploit SIMD in IBM Z. Or, install NVIDIA Triton Inference Server FROM ICR and deploy the model in it.

5.5 Requirements of a model inference server

While the model compiled by zDLC can be invoked directly by the C++, Java, or Python applications without going through a network, this gives us the highest performance benefit. However, in some cases, it is necessary to deploy the model in a separate container or process to provide inference services for more than one application.

The requirement of a model inference server will be very different from that of a server for model training (see Figure 5-10). It should be able to expose model endpoints as RESTful or gRPC interfaces, support multiple AI frameworks, and have version control, server-side micro-batching, and performance monitoring capability. In addition, it must handle a high volume of requests with low latency.

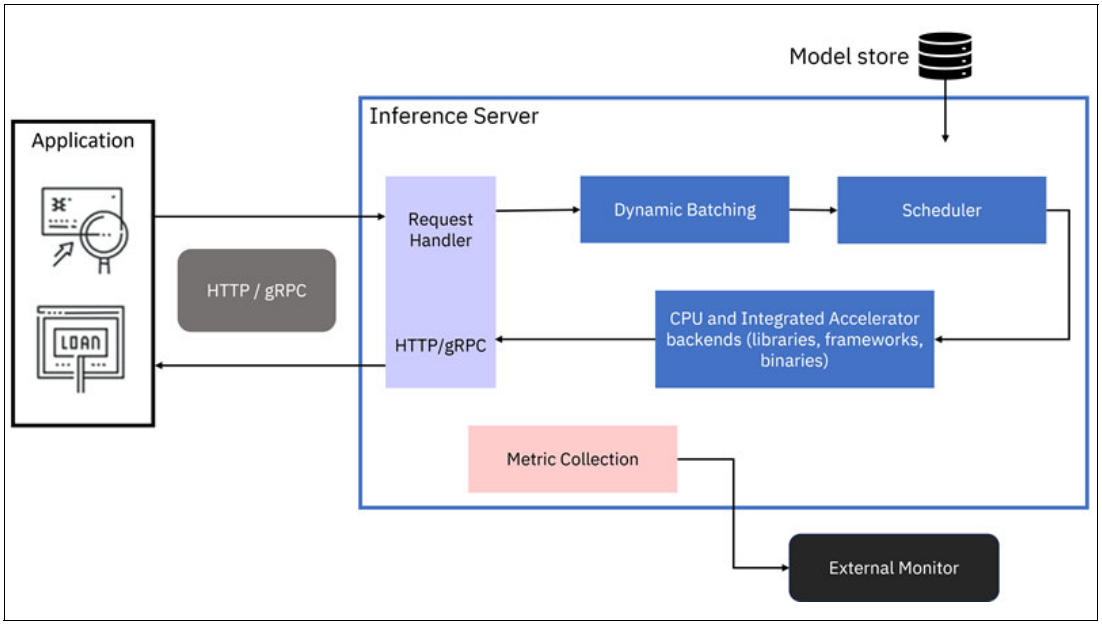

Figure 5-10 Inference Server Requirement

The following are entry points for leveraging the IBM z16 Integrated Accelerator for AI:

- Cloud Pak for Data on IBM Z: This enterprise solution has a fully integrated ML pipeline. For more details, see *IBM Cloud Pak for Data on IBM Z*, REDP-5695.
- AI Toolkit for IBM Z and IBM LinuxONE: Enables the customers to deliver AI solutions on IBM Z using open-source software with IBM support. For example, Z Deep Learning Compiler, SnapML, and NVIDIA Triton Inference Server are part of the AI Toolkit.
- Anaconda for IBM Z and IBM LinuxONE: Has leading-edge open-source frameworks and tools that leverage the IBM Z AI acceleration. A typical use case includes migration of existing Anaconda AI projects from another platform to IBM Z. For more information on installation of Anaconda for IBM Z and IBM LinuxONE, see Installing on Linux.

5.5.1 AI Toolkit for IBM Z and IBM LinuxONE

The required open-source software for AI inferencing on IBM Z is available at AI Toolkit for IBM Z and IBM LinuxONE.

The Toolkit includes IBM Elite Support and IBM Secure Engineering, which vet and scan open-source AI serving frameworks and IBM-certified containers for security vulnerabilities

and validate compliance with industry regulations. For more information about the AI Toolkit for IBM Z and IBM LinuxONE, see AI Toolkit for IBM Z and LinuxONE.

Feature technologies of the AI Toolkit for IBM Z and IBM LinuxONE include:

- IBM Container Registry
- ibmz-accelerated-for-snapml
- ibmz-accelerated-for-tensorflow
- ibmz-accelerated-serving-for-tensorflow
- ibmz-accelerated-for-nvidia-triton-inference-server

NVIDIA Triton Inference Server

The Triton Inference Server (Triton) is delivered as part of AI Toolkit for IBM Z and IBM LinuxONE. It is an open-source model server by NVIDIA that facilitates deploying AI models at scale.

The following are the Triton inference server features and advantages on inferencing in IBM Z:

- Support across devices:
  - Triton supports model inference across a wide variety of platforms due to the possibility of using different compute back ends, including the Telum processor of IBM Z and IBM LinuxONE.
- Integration with AI frameworks:
  - On IBM z16, Triton can leverage AI frameworks as back ends that exploit both the SIMD architecture and the IBM z16 Integrated Accelerator for AI. For more information, see Using the IBM Z Accelerated for NVIDIA Triton Inference Server Container Image.
- Features of Triton:
  - Triton supports server-side micro-batching.
  - Triton supports multiple frameworks, enabling customization for new frameworks and rule integrations.
  - Triton boasts model version control.
  - The platform supports concurrent model execution.
  - Triton facilitates seamless metrics and monitoring integration.
- Flexibility:
  - Triton supports various model types and can create custom model back ends, making it adaptable for different scenarios.
- Deployment on IBM z16 or IBM LinuxONE 4:
  - Focus on deploying models that leverage the IBM z16 Integrated Accelerator for AI on IBM z16 or IBM LinuxONE 4, specifically:
    - Traditional ML models in PMML, ONNX, or JSON format run using the IBM Snap ML library.
    - Deep learning models in the ONNX model format compiled with the IBM Z Deep Learning Compiler.

- Streamlined AI inferences:
  - The IBM Z Accelerated for NVIDIA Triton Inference Server helps streamline and standardize AI inferences by deploying ML or deep learning models from any framework on any CPU-based infrastructure.
- Concurrent execution:
  - Triton server runs models concurrently to maximize CPU-based inferencing, offering features like model ensembling and streaming inferencing.
- Deployment options:
  - The server can be deployed in cloud environments, on-premises data centers, or at the edge, providing inference service through an HTTP/REST or gRPC endpoint, allowing clients to request inferencing for any model managed by the server.
- Examples and guidance:
  - IBM has provided example container images and repositories for building and experimenting with Triton on a Linux on Z environment, including deploying models like a random forest classifier model and ONNX models compiled with the IBM Z Deep Learning Compiler.

This summary encapsulates the multifaceted advantages and features of the Triton Inference Server, primarily when utilized on IBM Z platforms, aiding in the efficient deployment and management of AI models.

Start using the AI Toolkit for IBM Z and IBM LinuxONE

First, obtain an IBM Cloud API key. Then, pull the AI Toolkit for IBM Z and IBM LinuxONE feature images from the ICR using the following steps:

1. Log in to IBM Cloud.
2. Use the **Manage** drop-down on the manual bar and select **Access (IAM)**.
3. On the navigation panel on the left, choose **API Keys** and create a new API key.
4. Copy the key or download a JSON file, which looks like:

```
{
   "name": "sample api key",
   "description": "sample api key",
   "createdAt": "2023-11-09T17:18+0000",
   "apikey": "fVn6NCD2aaQfZU4yNN4E21F1tPlItGIa49VAI3b-23gk"
}
```

Pulling container images from the ICR

You can find the list of container images in the IBM Containerized Workloads image registry.

To find the pull string, click on the image name. To pull the image of `ibmz-accelerated-for-nvidia-triton-inference-server`, complete the following steps:

1. First, sign on to the ICR:

```
podman login -u iamapikey icr.io
Password: <apikey>
```

2. Use the following pull string to pull the image of `ibmz-accelerated-for-nvidia-triton-inference-server`:

```
podman pull icr.io/ibmz-accelerated-for-nvidia-triton inference-server…
```

To show the image ID of Triton server, issue the following command (note that the image IDs shown here are examples):

```
podman images
REPOSITORY                      TAG                                IMAGE ID
CREATED      SIZE
icr.io/ibmz/zdlc                <none>                             53d7fdd53e43  7
weeks ago  538 MB
icr.io/ibmz/ibmz-accelerated-for-nvidia-triton-inference-server  3d1f7c309502  8
weeks ago  1.28 GB
```

5.6 Use case implementations

Since the supported flavors of Linux operating systems (Red Hat Enterprise Linux, SUSE Linux Enterprise Server, and Ubuntu) are optimized on IBM z16, the performance demonstrated in the Ubuntu environment shown earlier in this chapter should only have a minor difference by using Red Hat Enterprise Linux or SUSE-based setups. This section will show the steps to implement the use cases in the Podman container environment under Red Hat Enterprise Linux 8.

5.6.1 Implementing use cases 1 to 3 as in-transaction inferencing

Let us revisit our first three example use cases in 5.1, “Performance and environmental benefits of co-locating AI inferencing on IBM Z with OLTP” on page 44 and review the process of deploying a fraud detection model with an OLTP application as in-transaction inferencing.

Since the model is already exported in ONNX format, we can use the zDLC to create a JAR file directly invoked by the application using the commands outlined in Example 5-1.

Example 5-1 Creating a JAR file using the zDLC

```
export ZDLC_IMAGE=icr.io/ibmz/zdlc
export ZDLC_MODEL_NAME=ccf_lstm_static_tf2onnx_OS_new
export ZDLC_MODEL_DIR=<home>/zdlc_models
podman run --rm -v ${ZDLC_MODEL_DIR}:/workdir:z ${ZDLC_IMAGE} --EmitJNI --O3
--mcpu=z16 --mtriple=s390x-ibm-loz --maccel=NNPA ${ZDLC_MODEL_NAME}.onnx
```

A `ccf_lstm_static_tf2onnx_OS_new.jar` file is generated with a class signature as follows:

```
public static OMTensorList mainGraph(OMTensorList list)
```

The JAR file contains the LSTM model and the OLTP application can invoke it directly without going through the network.

5.6.2 Implementing inferencing in a separate container

This deployment example includes deploying a Random Forest classifier model on NVIDIA Triton Inference Server.

The process of deploying a random forest classifier model on NVIDIA Triton Inference Server includes the following steps:

1. Install required packages on Red Hat Enterprise Linux 8 on IBM Z or IBM LinuxONE:

   ```
   sudo yum install java openssl-devel
   ```

2. Clone the repository and navigate to the directory:

```
git clone https://github.com/IBM/ai-on-z-triton-is-examples.git
cd ai-on-z-triton-is-examples/snapml-examples/
```

3. Create and activate Conda environment:

```
conda env create -f ./environment.yml
conda activate snapenvz
```

4. Train the RF model and save it:

```
python train.py
cp model.pmml  <home>/triton/model-repo/rfc_model_py/1/
```

The model repository structure should look like:

```
model-repo
   rfc_model_py
      1
      model.pmml
      model.py
      config.pbtxt
```

5. Start Triton server with the model repository:

```
podman run --shm-size=1g --ulimit host --rm -p8000:8000 -p8001:8001 -p8002:8002
-v <home>/triton/model-repo:/models:z <tritonserver-s390x image id>
tritonserver --model-repository=/models
```

6. The following output indicates the model is running in Triton:

```
I1023 12:40:51.928297 1 server.cc:653]
+-------------------------------+---------+--------+
| Model                         | Version | Status |
+-------------------------------+---------+--------+
| rfc_model_py                  | 1       | READY  |
+-------------------------------+---------+--------+
```

7. Test the deployed model:

```
python test_service.py --model rfc_model_py
```

Here is the output:

```
{"model_name":"rfc_model_py","model_version":"1","outputs":[{"name":"OUT0","dat
atype":"FP64","shape":[1],"data":[1.0]}]}
```

We can apply the instructions above to various RF use cases. To demonstrate the performance capabilities of the IBM z16 Integrated Accelerator for AI against the competition, we used a different model and data.

5.7 Performance monitoring and tuning

To make best use of the available compute resources, identify bottlenecks in the running AI workload and apply proper tuning measures.

Mapping the compute operations in an AI model to the resources available in a system is a complex task covered by the AI frameworks and driver libraries involved. The IBM z16 Integrated Accelerator for AI supports certain operations particularly well, but also has

specific requirements on the input data. On the other hand, new AI models are constantly being developed and impose additional challenges on the existing accelerator hardware and software stacks.

5.8 Monitor IBM z16 Integrated Accelerator for AI usage

IBM z16 and IBM LinuxONE Emperor 4 provide two built-in facilities to monitor AI accelerator usage. Support for both the facilities has been added to the well-known Linux tool, **`perf`**.

Technical details regarding the Linux **`perf`** tool can be found in Linux on IBM Z and LinuxONE Device Driver, Features, and Commands Development stream (Kernel 6.2).

5.8.1 CPU Measurement Facility

The CPUMF provides a large set of CPU counters for various aspects of code execution on IBM Z. With IBM z16 and IBM LinuxONE, additional counters have been added that allow to usage monitoring of the IBM z16 Integrated Accelerator for AI. The accounting is done entirely by the machine and imposes no overhead on the running workload. The CPU MF counters are not virtualized and therefore are only available for workloads running in LPAR mode.

Enabling the CPU MF Facilities

Since the CPU MF Facilities allow the user to gather insights into all workloads running in the same LPAR, usage must be enabled explicitly in the LPAR activation profile. This is done by checking “Extended counter set authorization control” in the “Counter Facility Security Options” section in the LPAR profile.

Under Linux, the availability of the extended counter set can be checked using the **`lscpumf`** tool, as is shown in Example 5-2.

Example 5-2 Checking the availability of the extended counter set

```
$ lscpumf -i
CPU-measurement Counter Facility
--------------------------------------------------------------------
Version: 3.7

Authorized counter sets:
    Basic counter Set
    Crypto-Activity counter Set
    Extended counter Set
    MT-diagnostic counter Set
    Problem-State counter Set

Linux perf event support: Yes (PMU: cpum_cf)
…
```

Using the CPU MF facilities

The Linux **perf** tool supports the new counters added for the AI accelerator starting with these Linux distributions: Red Hat Enterprise Linux 8.7, Red Hat Enterprise Linux 9.1, SUSE Linux Enterprise Server 15SP5, and Ubuntu 22.04. The availability of the counters specific to the AI accelerator can be checked with the **lscpumf** tool as shown in Example 5-3.

Example 5-3 Checking availability of AI-specific counters on the IBM z16 Integrated accelerator for AI

```
$ lscpumf -c | grep NNPA
8:267   NNPA_INVOCATIONS
8:268   NNPA_COMPLETIONS
8:269   NNPA_WAIT_LOCK
8:270   NNPA_HOLD_LOCK
```

To record the counter increments triggered by a certain workload use the **perf** tool with the **stat** sub-command (see Example 5-4).

Example 5-4 Record the counter increments triggered by a specific workload

```
$ perf stat -e NNPA_INVOCATIONS,NNPA_COMPLETIONS,NNPA_WAIT_LOCK,NNPA_HOLD_LOCK
<workload>

 Performance counter stats for '<workload>':

             77      NNPA_INVOCATIONS
              3      NNPA_COMPLETIONS
          5,143      NNPA_WAIT_LOCK
      223,640,203        NNPA_HOLD_LOCK
```

Note: Additional **perf** counters can be added to the list to put the values into perspective. The number of cpu cycles **(-e cpu_cycles)** are of interest here. Operating system-provided counters can be recorded at the same time with the page-faults counter.

NNPA_INVOCATIONS

The number of total invocations of the Neural Network Processing Assist (NNPA) instruction. The NNPA instruction is a long running instruction that needs to be interrupted for handling page-faults or system interrupts. Therefore, NNPA might return with partial results indicated by a condition code greater than zero. The low-level driver library libzdnn automatically reissues the instruction in that case. These invocations result in additional increments of this counter. Because of that the total number of invocations is usually higher than the number of completions indicated by the next counter.

NNPA_COMPLETIONS

NNPA exiting with condition code zero indicates that the operation is complete. The ratio of this counter compared to NNPA_INVOCATIONS indicates how often the instruction had to be interrupted.

NNPA_WAIT_LOCK

This counter increments for every cycle the NNPA invocation was blocked due to the AI accelerator being occupied by a different thread. A high number indicates that too many threads are competing with the AI accelerator and is generally a bad sign.

NNPA_HOLD_LOCK

This counter increments for every cycle the NNPA invocation is running while holding the lock. A high number indicates a good utilization of the AI accelerator by the current thread and is generally a good sign.

For more information, see The CPU-Measurement Facility Extended Counters Definition for z10, z196/z114, zEC12/zBC12, z13/z13s, z14, z15, and z16.

5.8.2 Processor Activity Instrumentation Facility

The Processor Activity Instrumentation Facility (PAI) has been introduced with IBM z16 and IBM LinuxONE Emperor 4. PAI can gather statistics about the usage of special engines in the IBM Z system. For the IBM z16 Integrated Accelerator for AI it is possible to obtain information about how frequently the different operations of the NNPA instruction have been exercised. The helps with understanding the characteristics of running certain AI models on the accelerator.

In contrast to CPU MF, the PAI facility is fully virtualized and therefore can be used under KVM as well as under z/VM. With the help from the operating system the counter increments are recorded on a per-process basis. Due to overhead required in the operating system PAI imposes a minor impact on the performance of the running workload. PAI is enabled by default and is supported by these Linux distributions: Red Hat Enterprise Linux 9.2, SUSE Linux Enterprise Server 15 SP5, Ubuntu 22.04.

Using the PAI facility

The PAI facility counter is available with the **perf** tool. Use **`perf list`** to obtain a list of the available PAI NNPA counters as shown in Example 5-5.

Example 5-5 Using perf list

```
$ perf list pai_ext | grep NNPA_
  NNPA_1MFRAME
  NNPA_2GFRAME
  NNPA_ACCESSEXCEPT
  NNPA_ADD
  NNPA_ALL
  NNPA_AVGPOOL2D
  …
```

The PAI counters can be specified the same way as every other counter using the **-e option** to **perf** as shown in Example 5-6.

Important: Since the PAI counters are per-cpu counters, a CPU needs to be selected with the **-c** option. Alternatively, the **-a** option can be used to cumulate the counter values across all CPUs. Without **-c** and **-a** the current CPU is being recorded, which does not need to be the CPU the workload will run on.

Example 5-6 Specifying PAI counters

```
$ perf stat -a -e NNPA_MATMUL_OP,NNPA_RELU <workload>

 Performance counter stats for 'system wide':

                 3      NNPA_MATMUL_OP
                 3      NNPA_RELU
```

Note: PAI, CPUMF, other hardware counters as well as operating system counters can be recorded at the same time.

The following counters indicate how often the respective sub-function of NNPA has been used. Note that only full completions are counted, partial executions do not contribute to these increments.

NNPA_ADD, NNPA_AVGPOOL2D, NNPA_BATCHNORM, NNPA_CONVOLUTION, NNPA_DIV, NNPA_EXP, NNPA_GRUACT, NNPA_LOG, NNPA_LSTMACT, NNPA_MATMUL_OP, NNPA_MATMUL_OP_BCAST23, NNPA_MAX, NNPA_MAXPOOL2D, NNPA_MIN, NNPA_MUL, NNPA_RELU, NNPA_SIGMOID, NNPA_SOFTMAX, NNPA_SUB, and NNPA_TANH

The following counters increment for completed NNPA invocations and give indications on the sizes and dimensions of the input tensors. The exact definition of the counters depends on the sub-function executed with NNPA and be found in the Principles of Operation manual.

- NNPA_SMALLBATCH: Depending on the sub-function this indicates that either the height or the batch dimension is smaller than 32. A large number here indicates that the AI Accelerator does not operate efficiently.
- NNPA_LARGEDIM: Indicates that an input tensor dimension exceeds ¼ of the maximum dimension size.
- NNPA_SMALLTENSOR: Indicates that input tensors are smaller than 64 KB.
- NNPA_1MFRAME: Indicates that input tensor size is between 64 k and 1 MB. These are suitable for being covered by “large pages” and therefore might be handled more efficiently by the hardware.
- NNPA_2GFRAME: Indicates that input tensor size is between 1 MB and 2 GB. These are suitable for being covered by “huge pages” and therefore might be handled more efficiently by the hardware.
- NNPA_ACCESSEXCEPT: This counter indicates that access exceptions occurred while executing NNPA operation. This usually means that memory locations where not available and execution had to be interrupted to bring in the required memory areas. This can happen multiple times for an operation and is counted separately.
- NNPA_ALL: This is an artificial counter that sums up all the counters above.

5.9 Tuning

In this section, we give a few examples of how the gathered performance information can be used to improve the performance of the AI workload.

5.9.1 Thread-pinning to utilize multiple AI accelerators

The Integrated Accelerator is a compute engine residing on the chip alongside 8 IBM z16 cores. This means that threads running on the 8 cores of the same chip are competing over the same resource. Whenever the competing threads issue parallel NNPA invocations, only one of them can start right away with the computation on the accelerator, while the other is blocked by the hardware until the first operation is finished.

This situation is indicated by high numbers in the NNPA_WAIT_LOCK CPUMF counter.

When running in LPAR mode with more than 8 cores, performance can be significantly improved by pinning the AI compute threads to cores on different chips. In order to learn about the topology of the cores available in the system use the **`lscpu`** command as shown in Example 5-7.

Example 5-7 Using the lscpu command

```
$ lscpu -e -y
CPU NODE DRAWER BOOK SOCKET CORE L1d:L1i:L2 ONLINE CONFIGURED POLARIZATION ADDRESS
  0    0      1    2      1    0 0:0:0         yes yes        horizontal   0
  1    0      1    2      1    0 1:1:0         yes yes        horizontal   1
  2    0      1    2      1    1 2:2:0         yes yes        horizontal   2
  3    0      1    2      1    1 3:3:0         yes yes        horizontal   3
  4    0      1    2      1    2 4:4:0         yes yes        horizontal   4
  5    0      1    2      1    2 5:5:0         yes yes        horizontal   5
  6    0      1    2      1    3 6:6:0         yes yes        horizontal   6
  7    0      1    2      1    3 7:7:0         yes yes        horizontal   7
  8    0      1    2      1    4 8:8:0         yes yes        horizontal   8
  9    0      1    2      1    4 9:9:0         yes yes        horizontal   9
 10    0      1    2      1    5 10:10:0       yes yes        horizontal   10
 11    0      1    2      1    5 11:11:0       yes yes        horizontal   11
 12    0      1    2      2    6 12:12:0       yes yes        horizontal   12
 13    0      1    2      2    6 13:13:0       yes yes        horizontal   13
…
```

The interesting columns are the CPU column which indicates the CPU number as it is known by the Linux operating system, and the SOCKET column which indicates on which chip the CPU resides. In this case assigning the two AI compute threads to CPUs 0 and 12 instead of CPUs 0 and 1 allows two AI accelerators to be used in parallel resulting in a big performance boost.

To pin a thread to a certain CPU, the thread affinity mask, which by default contains all CPUs, needs to be adjusted. This can be done through the command line using the taskset tool, and the **`-c`** option. In order to do this programmatically, the **`sched_setaffinity`** function in the C library can be used.

5.9.2 Transparent Huge Pages

When dealing with large amounts of input and output data for an AI model we rely on this being handled efficiently by the hardware and the operating system. One feature provided by the Linux operating system is called Transparent Huge Pages (THP).

Memory allocations are broken down into smaller units (pages) by the operating system and the hardware. These pages are by default 4 BKB in size. However, for particularly large memory allocations it is more efficient to increase the page size.
This results in having smaller tables to maintain and to walk. With the THP feature, the Linux kernel can automatically bump up the page size when encountering large memory allocations. This feature is enabled by default on Red Hat and SUSE enterprise Linux distributions, but needs to be enabled manually on Ubuntu based systems.

Use the following command to check whether THP are enabled on the system:

```
$ cat /sys/kernel/mm/transparent_hugepage/enabled
always [madvise] never
```

The **madvise** setting indicates that THPs will only be used when explicitly requested using the **madvise C** library call. This is the default setting found on Ubuntu systems.

To benefit from the THP feature, either invoke **madvise** when doing the memory allocations or change this setting to `always` using the following command:

```
$ sudo /bin/bash -c "echo \"always\" >
/sys/kernel/mm/transparent_hugepage/enabled"
```

5.9.3 Touch memory of the output tensors

A minor performance improvement can be achieved by writing a dummy byte into every page of the output tensors. This triggers the copy-on-write mechanism in the Linux kernel to create a separate copy for this page right away instead of when the AI accelerator writes to it. This shows a performance benefit in particular for cases where no THPs can be used to back the memory.

5.9.4 Improving IBM z16 Integrated Accelerator for AI utilization

If there is a bad ratio between CPU and NNPA compute cycles for an AI model indicated by the **perf** counters (cpu_cycles versus nnpa_hold_lock) this usually means that the AI accelerator is under utilized and the CPU takes over compute tasks. This might have the following different root causes:

- The AI model operations in general are not supported by the AI accelerator.

  Although the operations implemented by the IBM z16 Integrated Accelerator for AI are carefully picked to support a wide range of AI models used in the market right now, there might be situations where a certain operation is not available. The AI frameworks and the compilers must fall back to CPU execution in these cases.

  There is unfortunately not much to be done except searching for other models with the same use cases. Please let your IBM representative know about this. IBM continues to work on improving the accelerator and adding new functionality.

- Operations exceed certain limits of the accelerator Maximum Design Input Signal (MDIS) or Mean Time to Saturation (MTS).

If either the tensors get too large or certain dimensions exceed the limits of the AI accelerator, the AI frameworks and compilers will fall back to executing the operation on the CPU. While single tensor dimensions cannot exceed 32 k, the overall size of tensors is limited to 4 GB. Splitting the tensors into smaller chunks helps here.

- Operations are replaced with unsupported constructs by the conversion process.

  Operations used in AI frameworks, compilers, and interchange formats do not always have the same semantics. When using ONNX as interchange format the translation might introduce either unnecessarily complex or unsupported constructs for otherwise supported AI models. If this cannot be circumvented by updating to the latest version of the conversion tools (like tf2onnx), reach out to an IBM representative for assistance.

5.10 Next steps

IBM has developed a suite of pre-built blueprints that guide you through hands-on experience to start your deployment of AI on IBM Z with various use cases and while leveraging a variety of technologies. To access these pre-built AI solution templates on IBM Z and IBM LinuxONE, see AI solution templates on IBM Z & LinuxONE.

Additionally, running your AI models using framework container images provided in the IBM Z and IBM LinuxONE Container Registry is a great place to start. Once you have running AI models, your next steps can include:

- Continuous monitoring and tuning of the AI models on IBM Z and IBM LinuxONE.
- Use of a container orchestration solution, such as Kubernetes or Red Hat OpenShift Container Platform, to meet the high availability and scalability requirements of your enterprise environment.

5.10.1 AI on IBM Z and IBM LinuxONE Discovery Workshop

This is a no charge discovery workshop designed to better understand the technology powering AI and can be conducted in-person or virtual depending on your needs.

The IBM Client Engineering team ensures you have the architecture, practices, and hands-on experience to begin the AI on IBM Z journey. The workshop includes specific use cases: designing solution architecture, developing a practical implementation plan, assisting with AI project implementation, and leveraging the right AI frameworks and tooling to solve your business problems to embed AI into your applications.

For more information, contact the AI on IBM Z and IBM LinuxONE Client Engineering team at ce4s@ibm.com.

6

Security and privacy

The integration of AI-enhanced workloads poses unique security and privacy challenges. This chapter delves into the intricacies of securing AI-enhanced Linux workloads on the IBM z16 mainframe. The emergence of generative AI and quantum computing has significantly impacted our lives and created a new set of challenges for privacy and security.
Linux on IBM Z solves these challenges by providing a computing platform for AI enhanced workloads, allowing you to focus on developing applications for your business goals.

6.1 Security

This section addresses security in 2 broad categories: system and workload. Addressing security at each layer is critical to ensure a fully trusted environment to run workloads. IBM Z was developed with security standards that are at or above the stringent standards set by the United States.

6.1.1 System security

The following are key features in Linux on IBM Z that help provide protection and assurance for business-critical workloads.

IBM Secure Execution for Linux

Introduced with the IBM z15 and IBM LinuxONE III systems, IBM Secure Execution is the process of creating encrypted Linux images capable of running on public, private, or hybrid cloud environments while ensuring the protection of their in-use memory.

IBM Secure Execution is based on hardware and firmware features introduced with the IBM z15, including a private host key, hardware memory protection, and an ultravisor that enforces memory protection. This ultravisor securely manages secret information passed from the guest using the public host key, and it is only accessible to trusted servers provided by your cloud provider. The boot image is encrypted, and integrity-protected, and secure memory isolates the guest from the Kernel-based Virtual Machine (KVM) hypervisor, preventing data manipulation and extraction. The ultravisor also safeguards the state information of a secure virtual server.

For more information, see IBM Secure Execution components.

Technical and operational assurance

IBM z15 and newer is one of the few systems in the world that offers a system that cannot be tampered with. Technical assurance is achieved when a chain of trust and set of encryption keys protect against the system or platform administrator and/or service provider. This form of assurance stands in contrast with operational assurance, a type of insurance offered by most service providers.

Operational assurance ensures service providers will not access client workloads, whereas technical assurance ensures that service providers cannot access client workloads.

Secure service container

Secure service container serves as the foundational infrastructure for combining the operating system, middleware, and software components into a self-sufficient appliance. This appliance prioritizes customizability and security while providing essential services. Appliances that integrate with IBM Z through Secure Service Container (SSC) technology, often referred to as SSC-based appliances, undergo encryption and signing to ensure confidentiality and integrity. They inherit the core characteristics of IBM Z, encompassing reliability and performance, and are inherently resistant to tampering. Access is exclusively achievable through well-defined REST interfaces using the secure hypertext transfer protocol (HTTPS). Appliance administrators can utilize these APIs to configure storage, network, or cryptographic resources for the appliances. All other forms of communication are blocked, including access to the operating system through secure shell (SSH).

There are three levels of protection is available for IBM Z systems with varying degrees of protection compared to Standard LPAR, all three are Evaluation Assurance Level (EAL) 5+ certified. To contract EAL levels, the US National Security Agency (NSA) only uses systems that are EAL 5+ and above.

The diagram in Figure 6-1 shows how a Standard LPAR compares to other security levels. The Hyper Protect platform shown in the figure is one of many IBM Cloud products, such as Universal Key Operator (UKO), that is part of the Hyper Protect Crypto Services, which is the only provider to achieve FIPS140-2 Level 4 certification. This is achieved by a special PCIe Crypto co-processor providing this functionality. With a design that is based on IBM Z, IBM offers a single tenancy Hardware Security Module (HSM) system available for customers called IBM Cloud Hyper Protect Crypto Services.

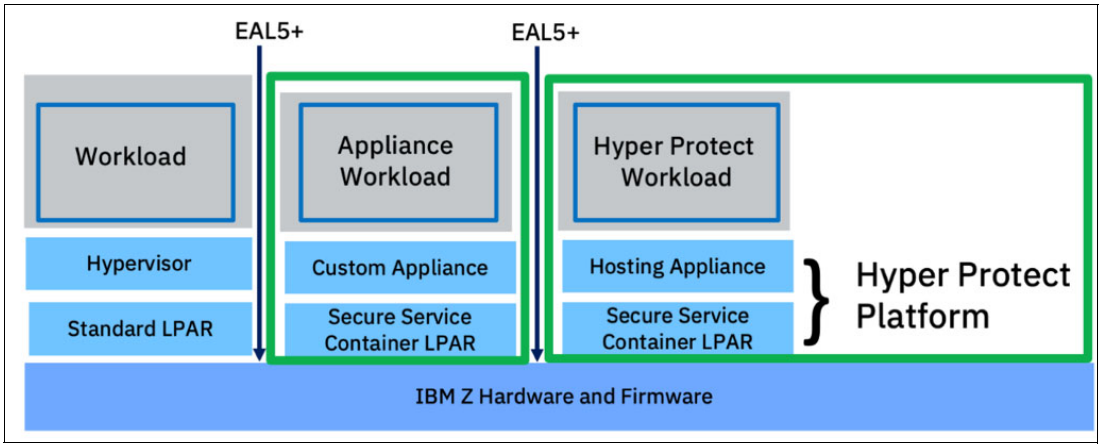

Figure 6-1 EAL security levels provided by IBM Z and its ecosystem

Confidential computing

Data protection is only as strong as the weakest link in end-to-end defense – meaning that data protection should be holistic. Confidential computing fueled by IBM Z Hyper Protect architecture is essential to provide a secure crucible to run enterprise AI.

In confidential computing, sensitive data is processed within a secure enclave or a trusted execution environment (TEE), where the data remains encrypted and isolated from the underlying infrastructure and any potentially malicious elements. As a result, confidential computing not only safeguards data during processing but also enhances trust in AI systems, making it a crucial component for various applications, including healthcare, finance, and more.

It empowers organizations to harness the full potential of AI while maintaining the highest standards of data security and privacy, instilling confidence in users and compliance with stringent regulatory requirements.

6.1.2 Workload security

While research is ongoing and commercial adoption will take some time, IBM is an in-use encryption thought leader. Quantum safe encryption can be implemented using IBM provided tools.

Triad of data protection

Quite often you hear about encryption when it comes to securing the data from prying hands of hackers. Let us dive into a few concepts of encryption.

The at-rest encryption is encrypting data where it resides in a storage subsystem. This encryption method that had been a staple for securing data has its flaws, for example when an AI model needs to process the data for training or inference, the data would need to be decrypted. This makes the data vulnerable to unauthorized access. The encryption and decryption are computing intensive operations so frequent operation could impact data processing velocity.

In-transit encryption is encrypted while data is in transit from location to other, this could be between two processes within the same system or between two global endpoints spanning 1000s of miles. The ubiquity of the internet has made Secure Sockets Layer (SSL) an essential part of communication channel encryption between any two endpoints.

Few system or software vendors venture into in-use encryption. With its fully homomorphic encryption (FHE), IBM customers can process data without needing to decrypt it. While current encryption techniques allow data to be protected during storage and in transit, data must be decrypted during computation, exposing it to greater security and privacy risks. Fully homomorphic encryption allows data to remain encrypted even while being computed upon, closing this gap in conventional encryption models, as illustrated in Figure 6-2.

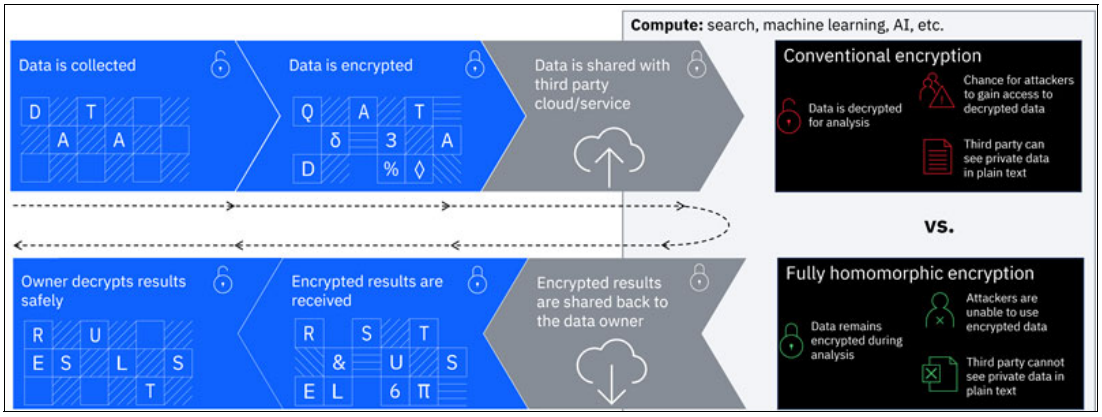

Figure 6-2 The IBM FHE Toolkit available for Linux on IBM Z customers

A top Brazilian bank piloted the IBM FHE toolkit in partnership with IBM Research and found that using FHE along with encrypted models performed without loss in accuracy compared to normal models. Since the use case is for batch processing, completion times are higher while the accuracy of results did not change.

For more information, see IBM Fully Homomorphic Encryption (HELayers) SDK for Linux.

For more details about FHE, see Fully Homomorphic Encryption.

Quantum safety

Current cryptography relies heavily on hard mathematical algorithms. With the current generation of compute, it is extremely difficult or impossible to break 2048-bit keys. The larger the key size, the more difficult it is to break. However, IBM Research and a few others have ventured into quantum computing. As quantum computers advance, they have the potential to break many of the cryptographic algorithms currently used to secure data. Quantum-safe encryption methods are designed to be resilient to quantum attacks, ensuring that sensitive information remains confidential in a post-quantum computing era.

Due to the potential risk to cryptographic algorithms resulting from the expansion of quantum computing, government bodies around the world have begun working to define post-quantum Quantum-safe cryptography standards for general encryption and digital signatures.

To address these growing security concerns, IBM introduced with the IBM z16 platform, with embedded quantum-level encryption in its Telum processor in addition to new features for Crypto Express8S hardware security modules (HSMs) offering quantum-safe key management.

For more technical details about the IBM Systems hardware and embedded quantum-level encryption, see IBM z16 family adds embedded AI and quantum-level encryption into new single frame and rack mount options.

IBM z16 family adds embedded AI and quantum-level encryption into new single frame and rack mount options

For a deeper dive into quantum safety, see Future-Proofed: The Journey Towards Quantum-Safe Security.

6.2 Privacy

This section introduces privacy in terms of growing regulations to govern AI technology and address the privacy concerns with generative AI. IBM has been a thought leader to provide enterprises with its open-source AI toolsets well before any specific regulations came into existence. In addition, with Linux on IBM Z, customers can access the power of the IBM Cloud Paks that include the state-of-the-art tools to protect Personally Identifiable Information (PII) data and govern data source lineage.

6.2.1 Compliance and regulation

IBM Z platform customers use the mainframe for their source of truth for transactional data. This data is entrusted to the IBM Z mainframe due to its unparalleled reliability, resilience, and robust security standards. In the era of data-driven decision-making and the relentless pursuit of enhanced computing power, the convergence of AI and mainframes is reshaping the landscape of enterprise computing. The power of enterprise grade Red Hat Enterprise Linux as an operating system enables enterprises to leverage cutting edge software stacks on a reliable hardware platform.

It is imperative to ensure that sensitive information remains confidential. The European Union's General Data Protection Regulation (GDPR), California Consumer Privacy Act (CCPA), and similar regulations mandate stringent data protection measures. To comply with these regulations, organizations must implement robust encryption, access controls, and data masking techniques, both at rest and in transit.

The United States White House Office of Science and Technology Policy has released a Blueprint of an AI Bill of Rights to protect the rights of American public.[1] Even well before the generative AI wave, since 2011, the US Department of the Treasury has been reviewing financial models used for decision making by banks and other financial institutions.[2]

Though countries across the world have some privacy laws and laws on individual’s rights, there are no regulatory bodies monitoring what goes into an AI model and implications on decisions made using the model. For more information, see Global AI Law and Policy Tracker.

[1] https://www.whitehouse.gov/ostp/ai-bill-of-rights/
[2] https://www.occ.treas.gov/news-issuances/bulletins/2011/bulletin-2011-12a.pdf

IBM has been at the forefront of AI governance with its software tools made available part of IBM Cloud Pak for Data and the IBM Research AI Factsheet 360, which provides a holistic governance model that covers four personas: Business Owner, Data Scientist, Model Validator, and AI Operations Engineer. For more information, see AI Lifecycle Governance.

6.2.2 Data privacy

To ensure privacy of individuals or entities in a dataset, comply with data residency requirements, often enterprises need a data governance model. This can be implemented using tools such as Watson Knowledge Catalog, which offers data lineage and masking PII and other private data before the data is used for AI models.

To further enhance privacy and security of the data, differential privacy is gaining ground. Differential privacy provides a rigorous mathematical algorithm that uses heuristics to preserve the privacy of individuals when their data is used in data sets. This is particularly helpful in cases where sensitive data is involved and performs exceptionally well in protecting user data for large data collections. For more information, see "What is Differential Privacy?"

IBM Z Data Privacy Passports provide a comprehensive Trusted Data Objects that replaces point to point encryption and eliminates many common vulnerabilities and risks for your data. For more information, see IBM Data Privacy Passports.

6.2.3 Conclusion

The integration of AI-enhanced workloads on IBM z16 mainframes presents unique security and privacy challenges, and this chapter discusses the key aspects of addressing these challenges.

Security challenges addressed by the platform include the following:

- System security: IBM z16 offers features like Secure Execution for Linux, Technical Assurance, and Secure Service Containers, ensuring the protection of in-use memory and providing levels of security that exceed standard LPAR configurations.
- Confidential computing: IBM Z Hyper Protect architecture enables confidential computing, where sensitive data is processed within secure enclaves or trusted execution environments, maintaining encryption and isolation to safeguard data during AI processing.
- Workload security: IBM Full Homomorphic Encryption Toolkit allows data to be processed without decryption, enhancing data security during AI operations. Quantum-safe encryption methods are also incorporated to prepare for post-quantum computing threats.

Privacy challenges addressed by the platform include the following:

- Compliance and regulation: To comply with data protection regulations such as GDPR and CCPA, organizations must implement robust encryption, access controls, and data masking techniques for AI workloads, both at rest and in transit. Emerging AI capabilities necessitate considerations for an AI Bill of Rights.
- Data privacy: Data governance models and tools like Watson Knowledge Catalog are essential for ensuring the privacy of individuals' data. Differential privacy offers a mathematical algorithm to prevent an individual or organization from identifying an individual within a protected dataset through comparison of the data to other data sets. IBM Z Data Privacy Passports provide comprehensive data protection solutions.

The convergence of AI and mainframes offers unprecedented computing power but requires a meticulous approach to ensure data security and privacy in the era of AI-driven decision-making. IBM's governance tools and technologies are instrumental in achieving these objectives.

The combination of robust security measures, compliance with data protection regulations, and advanced privacy technologies ensures the safe and responsible integration of AI workloads on IBM z16 mainframes.

7

Future trends

In this final chapter, we bring together the key insights from our exploration of the IBM Telum processor, focusing on its AI acceleration capabilities. We highlight the potential for seamless integration of AI acceleration throughout the enterprise system stack and discuss the future prospects for the IBM Telum processor.

The IBM Telum processor is engineered with a strong focus on vertical integration. This approach enables the chip's utilization and impact to be felt across various layers of the enterprise system stack. In this section, we delve into how the integration of the IBM Telum processor benefits each layer of the stack, from operating systems to application development.

As AI technology continues to evolve at a rapid pace, it is essential to consider the future prospects for the IBM Telum processor and its AI acceleration capabilities. In this section, we touch upon the potential enhancements and applications of the IBM z16 Integrated Accelerator for AI, ensuring that it remains at the forefront of AI and ML hardware innovation.

With the IBM Telum processor's AI acceleration capabilities, it is anticipated that more industries will embrace AI technologies in their operations. Enterprises can tap into AI-driven insights, automation, and enhanced decision-making, propelling their businesses forward.

7.1 Exploring emerging AI technologies for Linux

Exploring emerging AI technologies for Linux on IBM z16 entails the convergence of advanced enterprise-grade Linux systems and cutting-edge AI capabilities. This integration introduces new possibilities for data-driven, intelligent computing, particularly in the context of AI acceleration on IBM z16. The synergy between hardware and software components is paramount for successful AI integration in Linux workloads on IBM z16. The investment by IBM in AI libraries and tools specifically designed for the Linux ecosystem is pivotal to ensure seamless integration. Deep learning frameworks such as TensorFlow and PyTorch are at the forefront of AI research. Their efficient utilization on IBM z16 running Linux can empower organizations to leverage deep neural networks for a wide range of tasks, from image recognition to natural language processing AI-enhanced security is another facet that holds promise. AI-driven threat detection and security analytics can fortify the cybersecurity landscape on Linux systems. The implementation of AI-powered intrusion detection and real-time threat analysis can be instrumental in safeguarding critical data.

7.2 Predicting the future of AI in Linux workloads

Furthermore, AI-optimized workloads will become a common practice. As AI technologies continue to evolve, Linux workloads on IBM z16 can be optimized to leverage AI for tasks like data analysis, anomaly detection, and predictive maintenance, resulting in enhanced efficiency and responsiveness across various industries.

Predicting the future of AI in Linux workloads on IBM z16 reveals a landscape where AI is seamlessly integrated into the core of enterprise computing. It becomes an indispensable core component, relied upon for data-driven decision-making, automation, and enhanced system performance. Efficiency, real-time insights, and enhanced security are among the key anticipations for the future. AI-driven analytics will enable Linux workloads to provide real-time insights and support industries like finance, healthcare, and manufacturing. AI-driven security will actively monitor and respond to evolving threats, fostering system integrity. The commitment by IBM to open-source AI libraries will foster a collaborative ecosystem, where organizations and developers contribute to and benefit from a growing repository of AI-powered solutions for Linux workloads on IBM z16. The future of AI in Linux workloads on IBM z16 is marked by deep integration, efficiency, and proactive intelligence, setting the stage for a dynamic and intelligent future in enterprise computing. Future Applications and Potentialities

7.2.1 watsonx future applications

As the digital landscape evolves, the symbiosis of cutting-edge hardware and innovative software solutions becomes paramount. The IBM Z, with its advanced capabilities, is poised to redefine the future of AI mainframe computing. This section delves into the potential applications of the IBM Z and its synergy with watsonx.data, a ground breaking platform designed to revolutionize how enterprises store, access, and leverage data for analytics and AI.

In the realm of AI on the IBM Z mainframe, watsonx.data is a pivotal component, redefining how enterprises manage, access, and scale their data for AI workloads. Let us delve deeper into the practical use of watsonx.data, specifically tailored to the IBM Z ecosystem, and explore how it seamlessly integrates into the AI workflow.

When viewed through the lens of an IBM Z mainframe, watsonx.data serves as a fit-for-purpose data store built on an open lakehouse architecture. This architecture is a key enabler for scaling AI workloads across all data sources, both on-premises and in the hybrid cloud. It is designed to provide a unified environment for data engineers, data scientists, and business analysts, fostering collaboration and ensuring a single, authoritative copy of the data.

The open lakehouse architecture of watsonx.data transcends traditional data storage paradigms. It seamlessly integrates structured and unstructured data, breaking down silos and providing a unified view of the data landscape. This architecture empowers enterprises to store and manage diverse datasets, a critical requirement for the varied data inputs essential to the success of AI models.

As the volume of data continues to surge, watsonx.data's scalability ensures that enterprises can scale their analytics and AI workloads effortlessly. This scalability, combined with the computational prowess of IBM z16, creates a formidable synergy that supports the ever-expanding requirements of AI applications.

Ensuring the security and compliance of data is non-negotiable in the realm of AI. watsonx.data incorporates robust governance features, providing enterprises with the tools needed to maintain data integrity and security. Furthermore, support for open data formats facilitates seamless data sharing and collaboration, laying the foundation for a data-driven ecosystem.

watsonx.data will have support multiple query engines tailored for optimized workloads on the IBM Z mainframe. These include Presto, Spark, IBM DB2®, and IBM Netezza®. The versatility of these engines allows dynamic scaling, enabling the utilization of different engines for various workloads, optimizing both performance and cost. The storage component of watsonx.data seamlessly integrates with external data sources, facilitating direct access by the query engines. This streamlined access is instrumental in preventing data duplication, enhancing data security, and accelerating the time to gain actionable insights. The mainframe's computational capabilities, coupled with watsonx.data's support for dynamic scaling of query engines, allow for optimal resource utilization. Users can employ different query engines for distinct workloads, optimizing costs while ensuring that AI models can leverage the most suitable engine for a given task.

Users will be able to easily connect storage, sync metadata, and integrate analytics environments within minutes. The process involves registering existing data sources from a list of supported connectors, eliminating the need for migration and preventing data duplication. This accelerates the time to insight by providing a seamless connection between the mainframe and external data sources.

watsonx.data fosters collaboration among data engineers, data scientists, and business analysts in a single environment. The mainframe becomes a hub for collaboration, enabling these key stakeholders to work with a single copy of the data, promoting consistency and reducing redundancy.

watsonx.data will also introduce generative AI for the discovery, augmentation, and enrichment of data and metadata. This feature simplifies complex tasks. Users can leverage natural language queries to retrieve relevant tables, benefiting from generative AI models that provide meaningful and descriptive table and column names and that can benefit some complex data schemes deployed on IBM Z.

watsonx.data on IBM Z not only addresses current AI challenges but also paves the way for future possibilities. The platform's continuous evolution, coupled with the computational prowess of IBM Z, positions enterprises at the forefront of AI innovation.

As organizations strive to harness the full potential of AI on the mainframe, watsonx.data stands as a testament to the commitment by IBM to provide robust, scalable, and cutting-edge solutions for the AI-driven future. To explore these capabilities further, consider starting a free trial or reaching out to an IBM representative. The journey to AI excellence on the IBM Z mainframe begins with watsonx.data.

As future potentialities regarding watsonx, there will be also potentialities for further enhancement of IBM Z mainframe application modernization.
watsonx Code Assistant for IBM Z emerges as a transformative solution, offering unparalleled value to businesses seeking to accelerate their modernization efforts. This AI-driven tool is designed to streamline and enhance the entire application lifecycle on the IBM Z platform. One of its key advantages is its ability to accelerate code development, significantly boosting developer productivity throughout the application lifecycle. watsonx Code Assistant for IBM Z contributes to lowering the overall costs and complexities associated with mainframe application modernization initiatives. By automating code refactoring and assisting in the conversion from COBOL to Java, the tool not only reduces the risk but also minimizes the total cost of modernization compared to alternative or manual approaches.

Additionally, watsonx Code Assistant for IBM Z stands out in its support for developers, offering assistance from application discovery to validation testing. This comprehensive approach ensures a smooth transition throughout the modernization journey. The tool leverages a state-of-the-art Large Language Model (LLM) fine-tuned for both COBOL and Java transformations, providing an unprecedented level of accuracy and efficiency.

One notable feature is its applicability to support application off load strategies. While watsonx Code Assistant for IBM Z is primarily designed for modernizing applications on the IBM mainframe, it offers substantial value if an organization decides to off load an application to another platform as part of a fit-for-purpose, hybrid cloud strategy.

Importantly, watsonx Code Assistant for IBM Z does not replace existing mainframe application modernization offerings but rather complements and integrates seamlessly with existing IBM Z software offerings. The synergy with IBM Z DevOps offerings is particularly crucial, serving as a foundational element to achieve maximum value from watsonx Code Assistant for IBM Z.

Regarding cost considerations, while the tool is not explicitly designed to lower software-related costs, its impact on developer productivity, reduction in complexities, and overall modernization efficiency contribute to a compelling business case. IBM emphasizes that watsonx Code Assistant for IBM Z introduces new AI-assisted capabilities without replacing existing offerings, offering a forward-looking approach to mainframe application modernization. As businesses navigate the evolving landscape of AI and Generative AI, watsonx Code Assistant for IBM Z stands at the forefront, unlocking business advantages through its state-of-the-art code-only LLM with 20 billion parameters. With a focus on COBOL and Java transformation, this tool epitomizes the commitment by IBM to deliver cutting-edge solutions that resonate with the speed, scope, and scale of the generative AI impact.

IBM Z and watsonx.data herald a new era for AI on IBM Z. As organizations embark on their AI journeys, the combined power of these technologies positions them at the forefront of innovation. The computational prowess of IBM z16 and watsonx.data data storage and governance capabilities create a symbiotic relationship that not only meets current AI demands but also anticipates and addresses the future challenges and opportunities that lie ahead. This dynamic duo stands as a testament to IBM's commitment to empowering enterprises in their quest for AI-driven excellence. Furthermore, with the introduction of watsonx Code Assistant for IBM Z, IBM reaffirms its dedication to advancing mainframe application modernization. This comprehensive suite of technologies positions businesses at the forefront of transformative AI applications, empowering them to navigate the complexities of the digital era with unparalleled innovation and foresight.

7.2.2 Homomorphic encrypted AI

The future of IBM Z holds tremendous promise in the realm of homomorphic applications and their integration with AI. Homomorphic encryption, a ground breaking technology, enables computations on encrypted data without the need for decryption, thus preserving data privacy and security. IBM Z, with its robust architecture and advanced security features, is poised to play a pivotal role in the development and deployment of homomorphic applications. This has significant implications for AI on IBM Z, as it allows organizations to leverage sensitive data for training and inference without compromising confidentiality.

The potential applications span various industries, including healthcare, finance, and government, where the need for secure and privacy-preserving AI is paramount. It will enable to have Privacy-preserving machine and deep learning solutions. In fact, their ability to process encrypted input data through machine and deep learning models will allow to guarantee the privacy of users during the processing, hence allowing to match the stricter and stricter legislation and recommendations in terms of data protection and user privacy.

The relevance of running Homomorphic Encryption (HE) AI models on IBM Z extends beyond its robust architecture to encompass comprehensive framework support. IBM Z is designed to seamlessly integrate with popular AI and ML frameworks, providing a versatile environment for developing and deploying HE AI models. Furthermore, the scalability and performance capabilities of IBM Z make it an ideal choice for running computationally intensive tasks associated with HE AI models.

The inherent parallel processing capabilities of IBM Z, coupled with its ability to handle large datasets efficiently, contribute to the acceleration of homomorphic computations. This is particularly crucial for AI applications that involve complex model training or inference processes. IBM Z, with its emphasis on security, data integrity, and encryption technologies, aligns seamlessly with the evolving landscape of privacy-conscious AI applications.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this paper.

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.

- *Keeping Up With Security and Compliance on IBM Z*, SG24-8540
- *Machine Learning with Business Rules on IBM Z: Acting on Your Insights*, REDP-5502
- *Security in Development: The IBM Secure Engineering Framework*, REDP-4641
- *Turning Data into Insight with Machine Learning for IBM z/OS*, SG24-8552

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

Online resources

These websites are also relevant as further information sources:

- IBM Z and Cloud Modernization Stack with IBM Cloud Pak for Data for Regulatory compliance

  https://www.ibm.com/docs/en/cloud-paks/z-modernization-stack/2023.4?topic=planning-regulatory-compliance
- IBM Z and IBM LinuxONE Content Solutions

  https://www.ibm.com/support/z-content-solutions/fully-homomorphic-encryption/
- IBM Z and IBM LinuxONE Security and Compliance Center

  https://www.ibm.com/products/z-security-and-compliance-center

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

REDP-5712-00

ISBN 0738459429

Printed in U.S.A.