Verantwortungsvolle KI und vertrauliche Inferenz - NetApp AI mit Protopia Image Transformation

NetApp Solutions

NetApp
December 03, 2024

Inhalt

Verantwortungsvolle KI und vertrauliche Inferenz - NetApp AI mit Protopia Image Transformation 1
TR-4928: Verantwortungsvolle KI und vertrauliche Inferenz - NetApp AI mit Protopia Image und Data Transformation 1
Lösungsbereiche 3
Technologieüberblick 5
Test- und Validierungsplan 9
Testkonfiguration 9
Testverfahren 9
Vergleich der Inferenzgenauigkeit 24
Obfuskationsgeschwindigkeit 25
Schlussfolgerung 25
Weitere Informationen und Bestätigungen 26

Verantwortungsvolle KI und vertrauliche Inferenz - NetApp AI mit Protopia Image Transformation

TR-4928: Verantwortungsvolle KI und vertrauliche Inferenz - NetApp AI mit Protopia Image und Data Transformation

Sathish Thyagarajan, Michael Oglesby, NetApp Byung Hoon Ahn, Jennifer Cwagenberg, Protopia

Visuelle Interpretationen sind zu einem integralen Bestandteil der Kommunikation mit dem Aufkommen von Bildaufnahme und Bildverarbeitung geworden. Künstliche Intelligenz (KI) in der digitalen Bildverarbeitung bietet neue Geschäftsmöglichkeiten, beispielsweise im medizinischen Bereich zur Identifizierung von Krebs oder anderen Krankheiten, in geospatialen visuellen Analysen zur Untersuchung von Umweltgefahren, in der Mustererkennung, in der Videoverarbeitung zur Kriminalitätsbekämpfung usw. Diese Gelegenheit bringt aber auch außerordentliche Verantwortung mit sich.

Je mehr Entscheidungen Unternehmen mit KI in die Hände geraten, desto größer sind ihre Risiken in Bezug auf Datenschutz, -Sicherheit sowie rechtliche, ethische und regulatorische Probleme. Dabei ermöglicht verantwortungsvolle KI Unternehmen und Behörden, Vertrauen und Governance aufzubauen, die für skalierbare KI in Großunternehmen entscheidend sind. Dieses Dokument beschreibt eine von NetApp im Rahmen von drei verschiedenen Szenarien validierte Lösung zur KI-Inferenz, indem NetApp Datenmanagement-Technologien mit Protopia Datenobfuskationssoftware eingesetzt werden, um sensible Daten zu privatisieren und Risiken sowie ethische Bedenken zu verringern.

Jeden Tag werden Millionen von Bildern mit verschiedenen digitalen Geräten von Kunden und Geschäftseinheiten erzeugt. Die Folge der enormen Zunahme von Daten und Computing-Workloads sind Unternehmen, die zu Cloud-Computing-Plattformen für Skalierbarkeit und Effizienz wechseln. Gleichzeitig ergeben sich bei der Übertragung in eine Public Cloud Bedenken hinsichtlich der Vertraulichkeit der vertraulichen Informationen in Bilddaten. Der Mangel an Sicherheit und Datenschutz wird zum Haupthindernis für die Implementierung bildverarbeitender KI-Systeme.

Darüber hinaus gibt es das "Recht auf Löschung" Gemäß der DSGVO ist ein Unternehmen berechtigt, alle personenbezogenen Daten zu löschen. Da ist auch der "Datenschutzgesetz", Die einen Code fairer Informationspraktiken festlegt. Digitale Bilder wie Fotografien können als personenbezogene Daten gemäß der DSGVO gelten, die regelt, wie Daten erhoben, verarbeitet und gelöscht werden müssen. Andernfalls muss die DSGVO nicht eingehalten werden, was zu hohen Geldstrafen führen kann, um Compliance-Vorschriften zu widerstehen, die Unternehmen ernsthaft beschädigen können. Datenschutzgrundsätze sind eines der Rückgrat der Implementierung einer verantwortungsvollen KI, die für Gerechtigkeit bei den Modellprognosen für Machine Learning (ML) und Deep Learning (DL) sorgt und die Risiken im Zusammenhang mit der Verletzung von Datenschutz oder gesetzlicher Vorschriften verringert.

In diesem Dokument wird eine validierte Design-Lösung unter drei verschiedenen Szenarien mit und ohne Imageverschleierung beschrieben, die für den Datenschutz relevant ist und eine verantwortungsvolle KI-Lösung implementiert.

- **Szenario 1.** On-Demand Inferenz innerhalb von Jupyter Notebook.
- **Szenario 2.** Batch Inferenz auf Kubernetes.
- **Szenario 3.** NVIDIA Triton Inferenzserver.

Für diese Lösung verwenden wir den Face Detection Data Set und Benchmark (FDDB), einen Datensatz von Gesichtsregionen, die für die Untersuchung des Problems der uneingeschränkten Gesichtserkennung konzipiert wurden, kombiniert mit dem PyTorch Machine Learning Framework zur Implementierung von FaceBoxes. Dieser Datensatz enthält die Anmerkungen für 5171 Gesichter in einem Satz von 2845 Bildern verschiedener Auflösungen. Der technische Bericht enthält zudem einige Lösungsbereiche und relevante Nutzungsfälle, die von NetApp Kunden und Field Engineers in Situationen erfasst wurden, in denen diese Lösung sinnvoll ist.

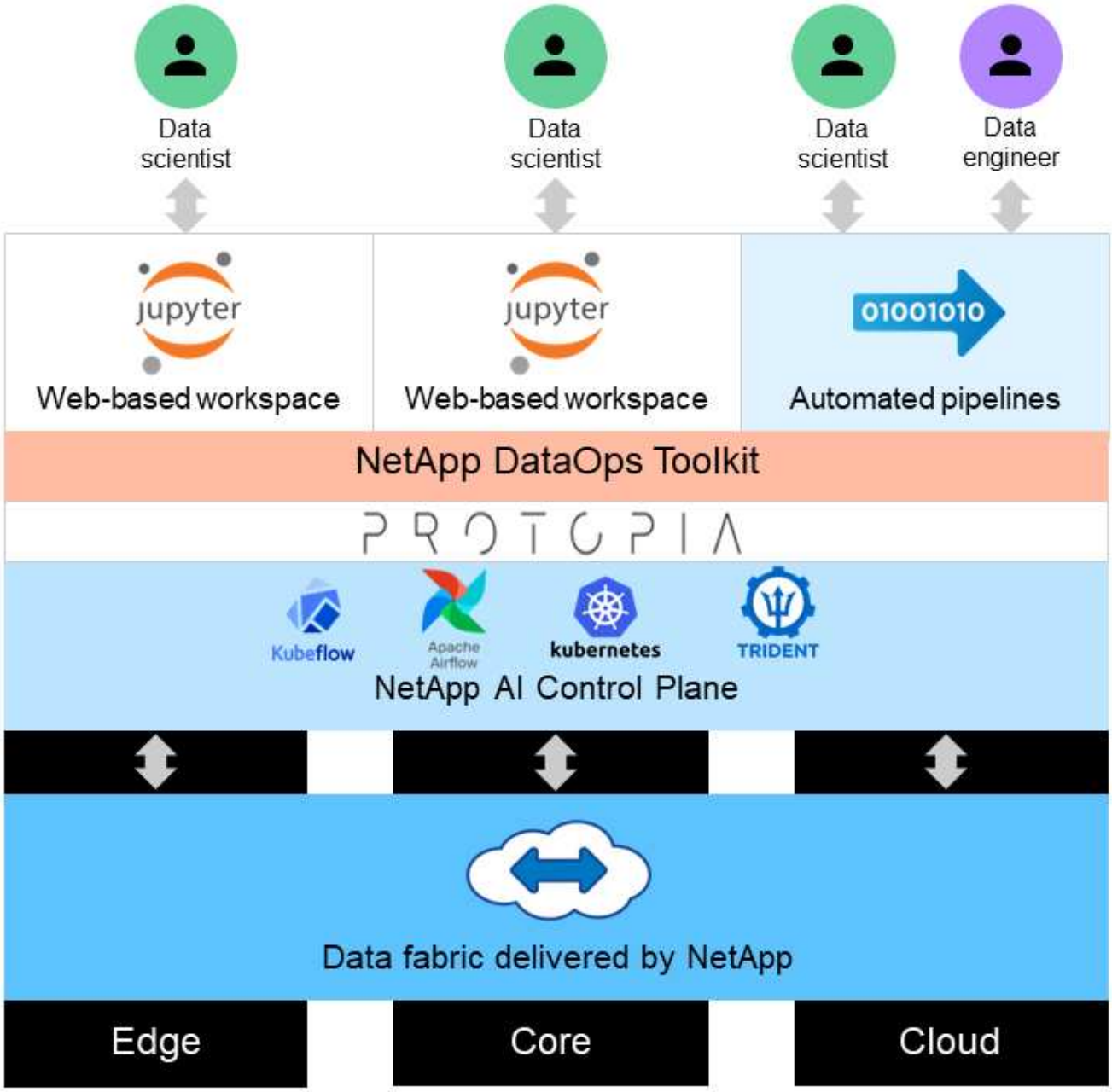
Zielgruppe

Dieser technische Bericht richtet sich an folgende Zielgruppen:

- Führungskräfte und Enterprise-Architekten, die eine verantwortungsvolle KI entwickeln und implementieren und Probleme hinsichtlich der Datensicherung und des Datenschutzes bei der Verarbeitung von Gesichts-Images in öffentlichen Bereichen lösen möchten.
- Data Scientists, Data Engineers, KI-/ml-Forscher (Machine Learning) und Entwickler von KI/ML-Systemen, die den Datenschutz schützen und bewahren möchten.
- Enterprise-Architekten, die Datenverschleisslösungen für KI/ML-Modelle und Applikationen entwerfen, die gesetzliche Standards wie DSGVO, CCPA oder den Datenschutzgesetz des US-Verteidigungsministeriums (DoD) und Regierungseinrichtungen erfüllen.
- Data Scientists und KI-Ingenieure suchen nach effizienten Möglichkeiten, Deep-Learning- (DL) und KI/ML/DL-Inferenzmodelle zu implementieren, die sensible Informationen sichern.
- Edge-Geräte-Manager und Edge-Serveradministratoren, die für die Implementierung und das Management von Edge-Inferenzmodellen verantwortlich sind

Lösungsarchitektur

Diese Lösung wurde entwickelt, um KI-Workloads in Echtzeit- und Batch-Inferenz auf großen Datensätzen zu verarbeiten. Dabei wird die Verarbeitungsleistung von GPUs neben herkömmlichen CPUs eingesetzt. Diese Validierung beweist, dass die Inferenz für ML geschützt ist und für Unternehmen, die einen verantwortlichen KI-Einsatz benötigen, optimales Datenmanagement erforderlich ist. Diese Lösung bietet eine Architektur für Kubernetes-Plattformen mit einzelnen oder mehreren Nodes für Edge- und Cloud-Computing und ist so mit NetApp ONTAP AI als zentrale On-Premises-Plattform, dem NetApp DataOps Toolkit und der Protopia Obfuscation Software über Jupyter Lab- und CLI-Schnittstellen verbunden. Die folgende Abbildung zeigt den Überblick über die logische Architektur der Data Fabric Vision von NetApp mit dem DataOps Toolkit und Protopia.



Die Protopia Obfuscation Software wird nahtlos auf das NetApp DataOps Toolkit ausgeführt und verändert die Daten, bevor der Storage Server verlassen wird.

Lösungsbereiche

Die digitale Bildverarbeitung bietet zahlreiche Vorteile, sodass viele Unternehmen das Beste aus ihren Daten in Verbindung mit visuellen Darstellungen machen können. Diese Lösung von NetApp und Protopia bietet ein einzigartiges Design zur KI-Inferenz zum Schutz und zur Privatisierung von KI/ML-Daten im ml/DL-Lebenszyklus. Kunden können damit die Kontrolle über sensible Daten behalten, Skalierungsmodelle und Effizienz in Public oder Hybrid-Cloud-Umgebungen nutzen, indem sie Bedenken im Zusammenhang mit dem Datenschutz ausräumen und die KI-Inferenz im Edge-Bereich implementieren.

Umweltbewusstsein

Es gibt viele Möglichkeiten, wie Branchen die Vorteile der Geospatialanalytik in den Bereichen der Umweltgefahren nutzen können. Regierungen und das Ministerium für öffentliche Arbeiten können nützliche Einblicke in die öffentliche Gesundheit und die Wetterbedingungen erhalten, um die Öffentlichkeit bei einer Pandemie oder einer Naturkatastrophe wie Waldbränden besser beraten zu können. So können Sie beispielsweise einen COVID-positiven Patienten in öffentlichen Räumen, wie Flughäfen oder Krankenhäusern, identifizieren, ohne die Privatsphäre der betroffenen Person zu beeinträchtigen und die jeweiligen Behörden und die Öffentlichkeit in der Umgebung auf notwendige Sicherheitsmaßnahmen aufmerksam zu machen.

Tragbare Geräte am Edge

Im Militär und auf Schlachtfeldern können Sie mithilfe der KI-Inferenz am Rand als tragbare Geräte den Status von Soldaten verfolgen, das Fahrverhalten überwachen und Behörden über die Sicherheit und die damit verbundenen Risiken im Zusammenhang mit der Annäherung an Militärfahrzeuge informieren, während die Privatsphäre von Soldaten geschützt und geschützt wird. Die Zukunft des Militärs wird mit dem Internet of Battlefield Things (IoBT) und dem Internet of Military Things (IoMT) High-Tech für tragbare Kampfausrüstung, die Soldaten helfen, Feinde zu identifizieren und besser im Kampf durch schnelle Edge-Computing. Der Schutz und der Schutz visueller Daten, die von Edge-Geräten wie Drohnen und tragbaren Getrieben erfasst werden, ist von entscheidender Bedeutung, um Hacker und Gegner vor Schach zu halten.

Nicht kämpferische Evakuierungsvorgänge

Nichtkämpferische Evakuierungsmaßnahmen (NEO) werden vom DoD durchgeführt, um bei der Evakuierung von US-Bürgern und Staatsangehörigen, ziviler DoD-Mitarbeiter und designierten Personen (HN) und Drittstaatsangehörigen (TCNs) zu helfen, deren Leben in Gefahr für einen geeigneten sicheren Hafen sind. Die vorhandenen administrativen Kontrollen nutzen weitgehend manuelle Evakuierungsverfahren. Jedoch könnte die Genauigkeit, Sicherheit und Geschwindigkeit der Evakuierung von ee-Identifizierung, Evakuierung von ee-Tracking und Bedrohungsabschirmung durch den Einsatz hochautomatisierter KI/ML-Tools in Kombination mit AI/ML-Videobfuskationstechnologien verbessert werden.

Gesundheitswesen und biomedizinische Forschung

Mithilfe der Bildverarbeitung werden Pathologien für die chirurgische Planung anhand von 3D-Bildern aus der Computertomographie (CT) oder der Magnetresonanztomographie (MRT) diagnostiziert. Die HIPAA-Datenschutzregeln regeln regeln die Art und Weise, in der Daten von Unternehmen für alle persönlichen Informationen und digitalen Bilder wie Fotos gesammelt, verarbeitet und gelöscht werden müssen. Damit die Daten gemäß den HIPAA Safe Harbor-Bestimmungen als teilbar gelten, müssen Fotos mit voller Fläche und vergleichbare Bilder entfernt werden. Automatisierte Techniken wie De-Identification oder Skull -Abisolieralgorithmen, die zur Obscrung der Gesichtszüge eines Einzelnen aus strukturellen CT/MR-Bildern verwendet werden, sind zu einem wesentlichen Bestandteil des Datenfreigabeprozesses für biomedizinische Forschungseinrichtungen geworden.

Cloud-Migration von KI/ML-Analysen

Enterprise-Kunden haben bisher KI/ML-Modelle lokal geschult und implementiert. Aus Skaleneffekten und Effizienzgründen erweitern diese Kunden zunehmend KI/ML-Funktionen in Public-, Hybrid- oder Multi-Cloud-Implementierungen. Sie sind jedoch an die gebunden, welche Daten anderen Infrastrukturen zugänglich sein können. NetApp Lösungen erfüllen die Anforderungen einer Reihe von Cybersicherheitsbedrohungen, die für erforderlich sind "Datensicherung" Die Sicherheitsbewertung und minimieren in Kombination mit der Protopia Datentransformation die Risiken, die mit der Migration von KI/ML-Workloads für die Bildverarbeitung in die Cloud verbunden sind.

Weitere Anwendungsfälle für Edge-Computing und KI-Inferenz in anderen Branchen finden Sie unter "TR-4886

KI-Inferenzierung am Edge" Mit dem NetApp AI Blog "Intelligenz und Datenschutz".

Technologieüberblick

Dieser Abschnitt bietet einen Überblick über die verschiedenen technischen Komponenten, die zum Abschließen dieser Lösung erforderlich sind.

Protopia

Protopia AI bietet heute eine unauffällige, rein softwarebasierte Lösung für vertrauliche Inferenz auf dem Markt. Die Protopia Lösung bietet unübertroffenen Schutz für Inferenz-Services, da die Gefahr von sensiblen Daten minimiert wird. Die KI ist nur aus den Informationen des Datensatzes gespeist, die für eine wirklich wichtige Aufgabe und nicht mehr zur Hand sind. Bei den meisten Inferenzaufgaben werden nicht alle in jedem Datensatz vorhandenen Informationen verwendet. Unabhängig davon, ob Ihre KI Bild-, sprach-, Video- oder sogar strukturierte tabellarische Daten verbraucht, steht Protopia nur bereit, was der Inferenz-Service benötigt. Die patentierte Kerntechnologie nutzt mathematisch kuratiertes Rauschen, um die Daten stochvoll zu transformieren und die Informationen zu erhalten, die von einem bestimmten ML-Service nicht benötigt werden. Diese Lösung maskiert die Daten nicht, sondern ändert die Datendarstellung durch kuratierte Zufallsgeräusche.

Die Protopia-Lösung formuliert das Problem der Änderung der Darstellung als gradienten-basierte Perturbationsmaximierungsmethode, die die relevanten Informationen im Eingabefeature-Raum hinsichtlich der Funktionalität des Modells behält. Dieser Erkennungsvorgang wird als Pass für die Feinabstimmung am Ende des Trainings DES ML-Modells ausgeführt. Nach dem Pass werden automatisch eine Reihe von Wahrscheinlichkeitsverteilungen generiert, bei der mit geringem Overhead Datenumwandlung werden Rauschproben aus diesen Distributionen an die Daten angewendet und vor dem Übergang zum Modell für die Inferenz verschleiert.

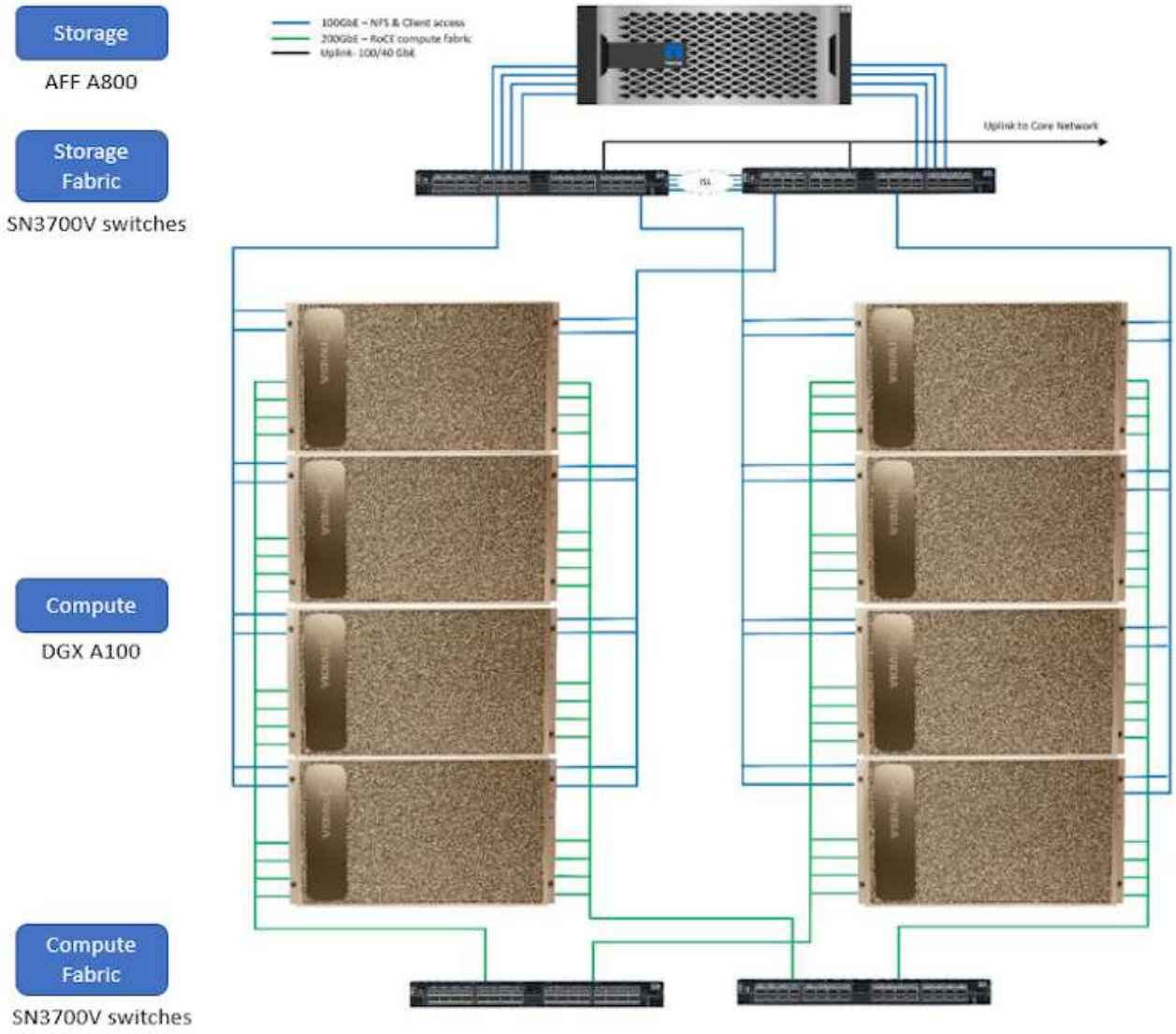
NetApp ONTAP AI

Die NetApp ONTAP AI Referenzarchitektur mit DGX A100-Systemen und Cloud-vernetzten NetApp Storage-Systemen wurde von NetApp und NVIDIA entwickelt und verifiziert. Sie bietet IT-Abteilungen eine Architektur mit folgenden Vorteilen:

- Sie vereinfacht das Design
- Computing und Storage können unabhängig voneinander skaliert werden
- Kunden können mit einer kleinen Konfiguration starten und nahtlos skalieren
- Zahlreiche Storage-Optionen für diverse Performance- und Kostenpunkte werden angeboten

ONTAP AI integriert DGX A100-Systeme und NetApp AFF A800 Storage-Systeme nahtlos in hochmoderne Netzwerke. Mithilfe von ONTAP AI können Komplexität und Unsicherheiten bei der Systemaufsetzung beseitigt werden, was den Einsatz von KI-Vorhaben vereinfacht. Kunden können mit einer kleinen Installation starten und unterbrechungsfrei wachsen. Gleichzeitig erhalten sie intelligente Datenmanagement-Funktionen, mit denen sich Daten zwischen Datenaufnahme, zentraler Datenplattform und Cloud frei verschieben lassen.

Die folgende Abbildung zeigt verschiedene Varianten der ONTAP AI Lösungsfamilie mit DGX A100-Systemen. Die Performance des AFF A800 Systems wurde mit bis zu acht DGX A100-Systemen verifiziert. Durch Hinzufügen von Storage-Controller-Paaren zum ONTAP Cluster kann die Architektur auf mehrere Racks skaliert werden, um viele DGX A100-Systeme und Petabyte an Storage-Kapazität mit linearer Performance zu unterstützen. Dieser Ansatz bietet die Möglichkeit, das Computing-zu-Storage-Verhältnis unabhängig voneinander anzupassen, je nach Größe der verwendeten DL-Modelle und der erforderlichen Performance-Kennzahlen.



Weitere Informationen zu ONTAP KI finden Sie unter "NVA-1153: NetApp ONTAP AI mit NVIDIA DGX A100-Systemen und Mellanox Spectrum Ethernet-Switches"

NetApp ONTAP

ONTAP 9.11, die jüngste Generation der Storage-Managementsoftware von NetApp, ermöglicht Unternehmen eine Modernisierung der Infrastruktur und den Übergang zu einem Cloud-fähigen Datacenter. Dank der erstklassigen Datenmanagementfunktionen lassen sich mit ONTAP sämtliche Daten mit einem einzigen Toolset managen und schützen, ganz gleich, wo sich diese Daten befinden. Zudem können Sie die Daten problemlos dorthin verschieben, wo sie benötigt werden: Zwischen Edge, Core und Cloud. ONTAP 9.11 umfasst zahlreiche Funktionen, die das Datenmanagement vereinfachen, geschäftskritische Daten beschleunigen und schützen und Infrastrukturfunktionen der nächsten Generation über Hybrid-Cloud-Architekturen hinweg ermöglichen.

NetApp DataOps Toolkit

NetApp DataOps Toolkit ist eine Python Library, mit der Entwickler, Data Scientists, DevOps Engineers und Data Engineers verschiedene Datenmanagement-Aufgaben ausführen können, zum Beispiel die nahezu sofortige Bereitstellung eines neuen Daten-Volumes oder einer JupyterLab Workspace, das nahezu sofortiges

Klonen eines Daten-Volumes oder einer JupyterLab Workspace. Und nahezu sofortige Snapshots von Daten-Volumes oder JupyterLab Workspace für Rückverfolgbarkeit oder Baselining. Diese Python-Bibliothek kann entweder als Befehlszeilen-Dienstprogramm oder als Bibliothek mit Funktionen verwendet werden, die Sie in jedes Python-Programm oder Jupyter-Notebook importieren können.

NVIDIA Triton Inferenz Server

NVIDIA Triton Inferenz Server ist eine Open-Source-Software für Inferenz-Server. Sie unterstützt die Standardisierung der Implementierung und Ausführung von Modellen, um schnelle und skalierbare KI in der Produktion zu ermöglichen. Triton Inference Server optimiert die KI-Inferenz, indem Teams trainierte KI-Modelle von jedem Framework auf jeder GPU- oder CPU-basierten Infrastruktur implementieren, ausführen und skalieren können. Der Triton Inference Server unterstützt alle wichtigen Frameworks wie TensorFlow, NVIDIA TensorRT, PyTorch, MXNet, OpenVINO und so weiter. Die Lösung lässt sich in Kubernetes integrieren und ermöglicht so die Orchestrierung und Skalierung, die Sie in allen wichtigen KI- und Kubernetes-Plattformen nutzen können. Es ist auch in viele MLOps Software-Lösungen integriert.

PyTorch

"PyTorch" Ist ein Open-Source-ML-Framework. Es ist eine optimierte Tensor-Bibliothek für Deep Learning, die GPUs und CPUs verwendet. Das PyTorch-Paket enthält Datenstrukturen für multidimensionale Tensor, die viele Dienstprogramme zur effizienten Serialisierung von Tensors unter anderen nützlichen Dienstprogrammen liefern. Es hat auch ein CUDA Pendant, mit dem Sie Ihre Tensor Berechnungen auf einer NVIDIA GPU mit Compute-Fähigkeit ausführen können. In dieser Validierung verwenden wir die OpenCV-Python (cv2) Bibliothek, um unser Modell zu validieren und dabei die intuitivsten Computer-Vision-Konzepte von Python zu nutzen.

Vereinfachtes Datenmanagement

Für den Enterprise IT-Betrieb und die Data Scientists spielt Datenmanagement eine zentrale Rolle, damit für KI-Applikationen die entsprechenden Ressourcen zum Training von KI/ML-Datensätzen verwendet werden. Die folgenden zusätzlichen Informationen über NetApp Technologien sind bei dieser Validierung nicht im Umfang enthalten, können jedoch je nach Ihrer Implementierung relevant sein.

Die ONTAP Datenmanagement-Software umfasst die folgenden Funktionen, um den Betrieb zu optimieren und zu vereinfachen und damit Ihre Gesamtbetriebskosten zu senken:

- Inline-Data-Compaction und erweiterte Deduplizierung: Data-Compaction reduziert den ungenutzten Speicherplatz in Storage-Blöcken, während Deduplizierung die effektive Kapazität deutlich steigert. Dies gilt für lokal gespeicherte Daten und für Daten-Tiering in die Cloud.
- Minimale, maximale und adaptive Quality of Service (AQoS): Durch granulare QoS-Einstellungen (Quality of Service) können Unternehmen ihre Performance-Level für kritische Applikationen auch in Umgebungen mit vielen unterschiedlichen Workloads garantieren.
- NetApp FabricPool: Bietet automatisches Tiering von „kalten“ Daten in Private- und Public-Cloud-Storage-Optionen, einschließlich Amazon Web Services (AWS), Azure und NetApp StorageGRID Storage-Lösung. Weitere Informationen zu FabricPool finden Sie unter "TR-4598: FabricPool Best Practices".

Beschleunigung und Sicherung von Daten

ONTAP bietet überdurchschnittliche Performance und Datensicherung, erweitert diese Funktionen auf folgende Weise:

- Performance und niedrige Latenz: ONTAP bietet höchstmöglichen Durchsatz bei geringstmöglicher Latenz.
- Datensicherung ONTAP verfügt über integrierte Funktionen für die Datensicherung mit zentralem

Management über alle Plattformen hinweg.

- NetApp Volume Encryption (NVE) ONTAP bietet native Verschlüsselung auf Volume-Ebene und unterstützt sowohl Onboard- als auch externes Verschlüsselungsmanagement.
- Multi-Faktor- und Multi-Faktor-Authentifizierung – ONTAP ermöglicht die gemeinsame Nutzung von Infrastrukturressourcen mit höchstmöglicher Sicherheit.

Zukunftssichere Infrastruktur

ONTAP bietet folgende Funktionen, um anspruchsvolle und sich ständig ändernde Geschäftsanforderungen zu erfüllen:

- Nahtlose Skalierung und unterbrechungsfreier Betrieb. Mit ONTAP sind das Hinzufügen von Kapazitäten zu bestehenden Controllern und das Scale-out von Clustern unterbrechungsfrei möglich. Kunden können Upgrades auf die neuesten Technologien wie NVMe und 32 GB FC ohne teure Datenmigrationen oder Ausfälle durchführen.
- Cloud-Anbindung: ONTAP ist die Storage-Managementsoftware mit der umfassendsten Cloud-Integration und bietet Optionen für softwaredefinierten Storage (ONTAP Select) und Cloud-native Instanzen (Google Cloud NetApp Volumes) in allen Public Clouds.
- Integration in moderne Applikationen: ONTAP bietet Datenservices der Enterprise-Klasse für Plattformen und Applikationen der neuesten Generation, wie autonome Fahrzeuge, Smart Citys und Industrie 4.0, auf derselben Infrastruktur, die bereits vorhandene Unternehmensanwendungen unterstützt.

NetApp Astra Control

Die NetApp Astra Produktfamilie bietet Storage und applikationsgerechte Datenmanagement-Services für Kubernetes-Applikationen On-Premises und in der Public Cloud auf der Basis von NetApp Storage- und Datenmanagement-Technologien. Sie ermöglicht einfaches Backup von Kubernetes-Applikationen, Migration von Daten in andere Cluster und die sofortige Erstellung von Klonen funktionierter Applikationen. Wenn Sie Kubernetes-Applikationen managen müssen, die in einer Public Cloud ausgeführt werden, finden Sie in der Dokumentation für "Astra Control Service". Astra Control Service ist ein von NetApp gemanagter Service, der applikationsgerechtes Datenmanagement für Kubernetes-Cluster in Google Kubernetes Engine (GKE) und Azure Kubernetes Service (AKS) ermöglicht.

NetApp Trident

Astra "Trident" NetApp ist ein Open-Source-Orchestrator für den dynamischen Storage von Docker und Kubernetes, das die Erstellung, das Management und die Nutzung von persistentem Storage vereinfacht. Trident, eine native Kubernetes-Applikation, wird direkt in einem Kubernetes Cluster ausgeführt. Trident ermöglicht Kunden die nahtlose Implementierung von DL-Container-Images auf NetApp Storage und bietet eine Erfahrung der Enterprise-Klasse für den Einsatz von KI-Containern. Kubernetes-Benutzer (ML-Entwickler, Data Scientists usw.) können die Orchestrierung und das Klonen erstellen, managen und automatisieren und so von erweiterten Datenmanagement-Funktionen der NetApp Technologie profitieren.

NetApp BlueXP Kopie und Synchronisierung

"BlueXP Copy und Sync" Ist ein NetApp Service für schnelle und sichere Datensynchronisierung. Unabhängig davon, ob Sie Dateien zwischen On-Premises-NFS- oder SMB-Dateifreigaben, NetApp StorageGRID, NetApp ONTAP S3, Google Cloud NetApp Volumes, Azure NetApp Files, Amazon Simple Storage Service (Amazon S3), Amazon Elastic File System (Amazon EFS), Azure Blob, Google Cloud Storage oder IBM Cloud-Objektspeicher übertragen müssen: BlueXP Copy and Sync verschiebt die Dateien schnell und sicher an den gewünschten Speicherort. Nach der Übertragung stehen die Daten an der Quelle und am Ziel vollständig zur Verfügung. BlueXP Copy und Syncc synchronisieren die Daten kontinuierlich basierend auf Ihrem

vordefinierten Zeitplan. Dabei werden nur die Deltas verschoben, sodass der Zeit- und Kostenaufwand für die Datenreplizierung minimiert wird. BlueXP Copy and Sync ist ein SaaS-Tool (Software-as-a-Service), das sich äußerst einfach einrichten und verwenden lässt. Datentransfers, die durch BlueXP Copy und Sync ausgelöst werden, erfolgen durch Datenmanager. Sie können Datenmanager von BlueXP Copy und Sync in AWS, Azure, Google Cloud Platform oder lokal implementieren.

NetApp BlueXP Klassifizierung

Unterstützt durch leistungsstarke KI-Algorithmen "NetApp BlueXP Klassifizierung" Automatisierte Kontrollmechanismen und Daten-Governance für den gesamten Datenbestand Hier können Sie mühelos Kosteneinsparungen ermitteln, Bedenken hinsichtlich Compliance und Datenschutz identifizieren und Möglichkeiten zur Optimierung finden. Das BlueXP Dashboard zur Klassifizierung bietet Ihnen Einblick in die Identifizierung mehrfach vorhandener Daten, um Redundanzen zu beseitigen, persönliche, nicht personenbezogene und sensible Daten zuzuordnen und Alarme für sensible Daten und Anomalien zu aktivieren.

Test- und Validierungsplan

Für dieses Lösungsdesign wurden die folgenden drei Szenarien validiert:

- Eine Inferenz mit und ohne Protopia obfuscation in einem JupyterLab Workspace, der mithilfe des NetApp DataOps Toolkit für Kubernetes orchestriert wurde.
- Einen Job zur Batch-Inferenz mit und ohne Protopia obfuscation auf Kubernetes mit einem Daten-Volume, das mithilfe des NetApp DataOps Toolkit für Kubernetes orchestriert wurde.
- Ein Inferenz-Task unter Verwendung einer NVIDIA Triton Inferenz Server-Instanz, die mit dem NetApp DataOps Toolkit für Kubernetes orchestriert wurde. Vor dem Aufruf der Triton-Inferenz-API haben wir Protopia-Obfuscation auf das Bild angewendet, um die allgemeine Anforderung zu simulieren, dass alle über das Netzwerk übertragenen Daten obfuscated werden müssen. Dieser Workflow eignet sich für Anwendungsfälle, in denen Daten in einer vertrauenswürdigen Zone erfasst, jedoch zur Inferenz außerhalb dieser vertrauenswürdigen Zone weitergeleitet werden müssen. Ohne Protopia Obfuscation ist es nicht möglich, diese Art von Workflow ohne sensible Daten zu implementieren, die die vertrauenswürdige Zone verlassen.

Testkonfiguration

In folgender Tabelle wird die Validierungsumgebung für das Lösungsdesign beschrieben.

Komponente	Version
Kubernetes	1.21.6
NetApp Trident-CSI-Treiber	22.01.0
NetApp DataOps Toolkit für Kubernetes	2.3.0
NVIDIA Triton Inferenz Server	21.11-py3

Testverfahren

In diesem Abschnitt werden die Aufgaben beschrieben, die zum Abschließen der Validierung erforderlich sind.

Voraussetzungen

Um die in diesem Abschnitt beschriebenen Aufgaben auszuführen, müssen Sie auf einen Linux- oder macOS-Host zugreifen können, auf dem die folgenden Tools installiert und konfiguriert sind:

- Kubectl (für Zugriff auf ein vorhandenes Kubernetes-Cluster konfiguriert)
  - Anweisungen zur Installation und Konfiguration finden Sie "Hier".
- NetApp DataOps Toolkit für Kubernetes
  - Installationsanweisungen finden Sie "Hier".

Szenario 1 – On-Demand-Inferenz in JupyterLab

1. Kubernetes-Namespace für KI/ML-Inferenz-Workloads erstellen

```
$ kubectl create namespace inference
namespace/inference created
```

2. Verwenden Sie das NetApp DataOps Toolkit, um ein persistentes Volume für die Daten bereitzustellen, auf denen Sie die Inferenz ausführen.

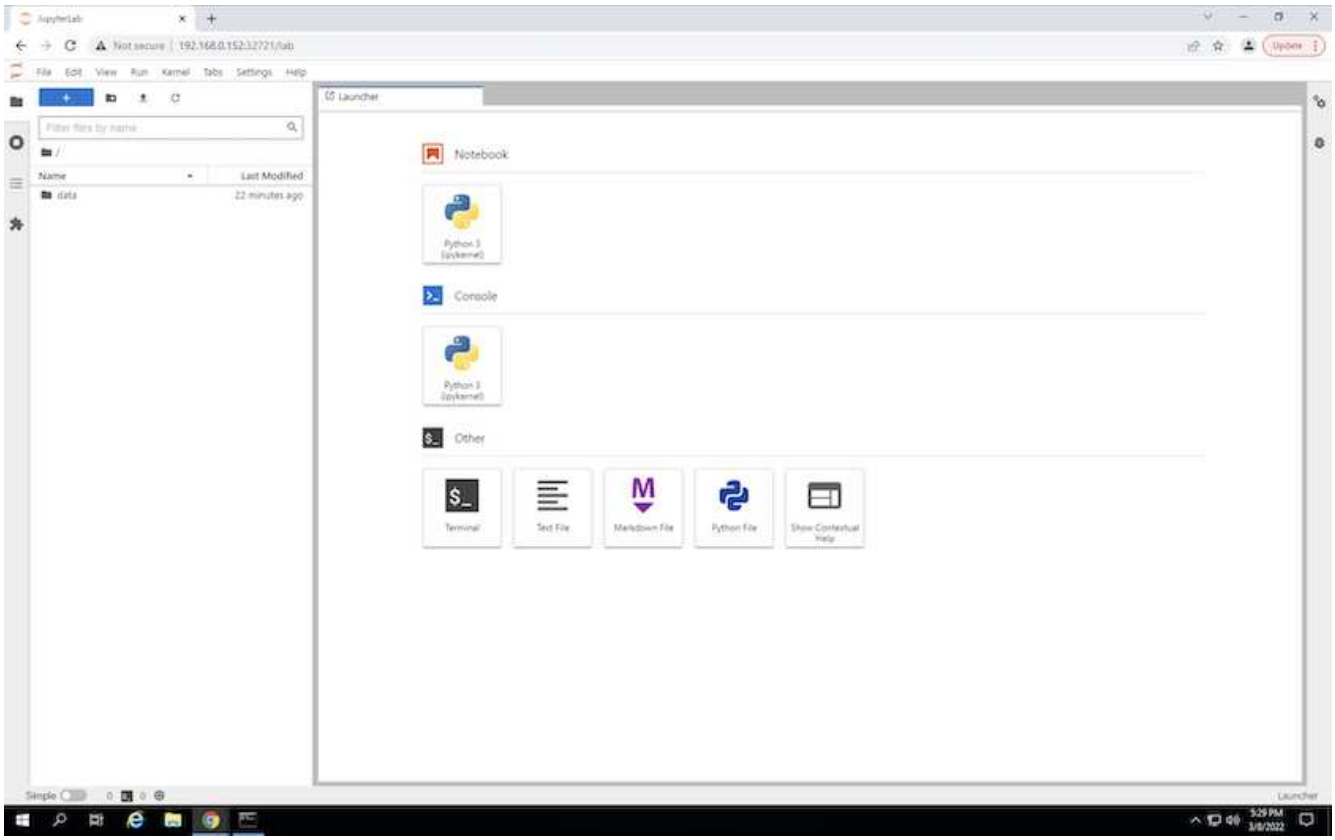
```
$ netapp_dataops_k8s_cli.py create volume --namespace=inference --pvc
-name=inference-data --size=50Gi
Creating PersistentVolumeClaim (PVC) 'inference-data' in namespace
'inference'.
PersistentVolumeClaim (PVC) 'inference-data' created. Waiting for
Kubernetes to bind volume to PVC.
Volume successfully created and bound to PersistentVolumeClaim (PVC)
'inference-data' in namespace 'inference'.
```

3. Verwenden Sie das NetApp DataOps Toolkit, um einen neuen JupyterLab Workspace zu erstellen. Mounten Sie das persistente Volume, das im vorherigen Schritt mit dem erstellt wurde `--mount- pvc` Option. Weisen Sie mit dem NVIDIA-GPUs dem Workspace nach Bedarf zu `-- nvidia-gpu` Option.

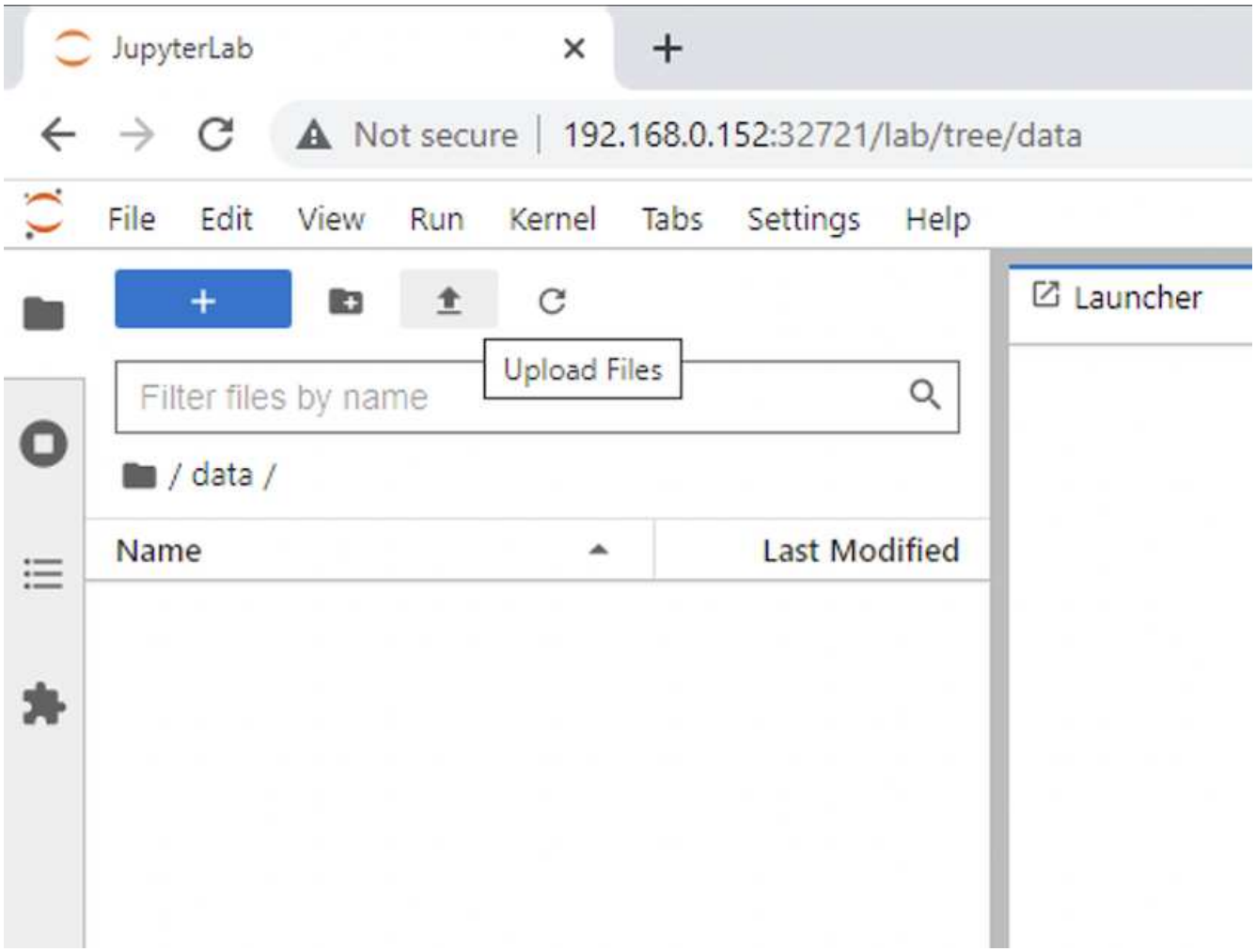
   Im folgenden Beispiel wird das persistente Volume beschrieben `inference-data` Wird im JupyterLab-Workspace-Container unter installiert `/home/jovyan/data`. Bei der Verwendung von offiziellen Projekt-Jupyter-Container-Images, `/home/jovyan` Wird als oberes Verzeichnis innerhalb der JupyterLab-Weboberfläche dargestellt.

```
$ netapp_dataops_k8s_cli.py create jupyterlab --namespace=inference
--workspace-name=live-inference --size=50Gi --nvidia-gpu=2 --mount
-pvc=inference-data:/home/jovyan/data
Set workspace password (this password will be required in order to
access the workspace):
Re-enter password:
Creating persistent volume for workspace...
Creating PersistentVolumeClaim (PVC) 'ntap-dsutil-jupyterlab-live-
inference' in namespace 'inference'.
PersistentVolumeClaim (PVC) 'ntap-dsutil-jupyterlab-live-inference'
created. Waiting for Kubernetes to bind volume to PVC.
Volume successfully created and bound to PersistentVolumeClaim (PVC)
'ntap-dsutil-jupyterlab-live-inference' in namespace 'inference'.
Creating Service 'ntap-dsutil-jupyterlab-live-inference' in namespace
'inference'.
Service successfully created.
Attaching Additional PVC: 'inference-data' at mount_path:
'/home/jovyan/data'.
Creating Deployment 'ntap-dsutil-jupyterlab-live-inference' in namespace
'inference'.
Deployment 'ntap-dsutil-jupyterlab-live-inference' created.
Waiting for Deployment 'ntap-dsutil-jupyterlab-live-inference' to reach
Ready state.
Deployment successfully created.
Workspace successfully created.
To access workspace, navigate to http://192.168.0.152:32721
```

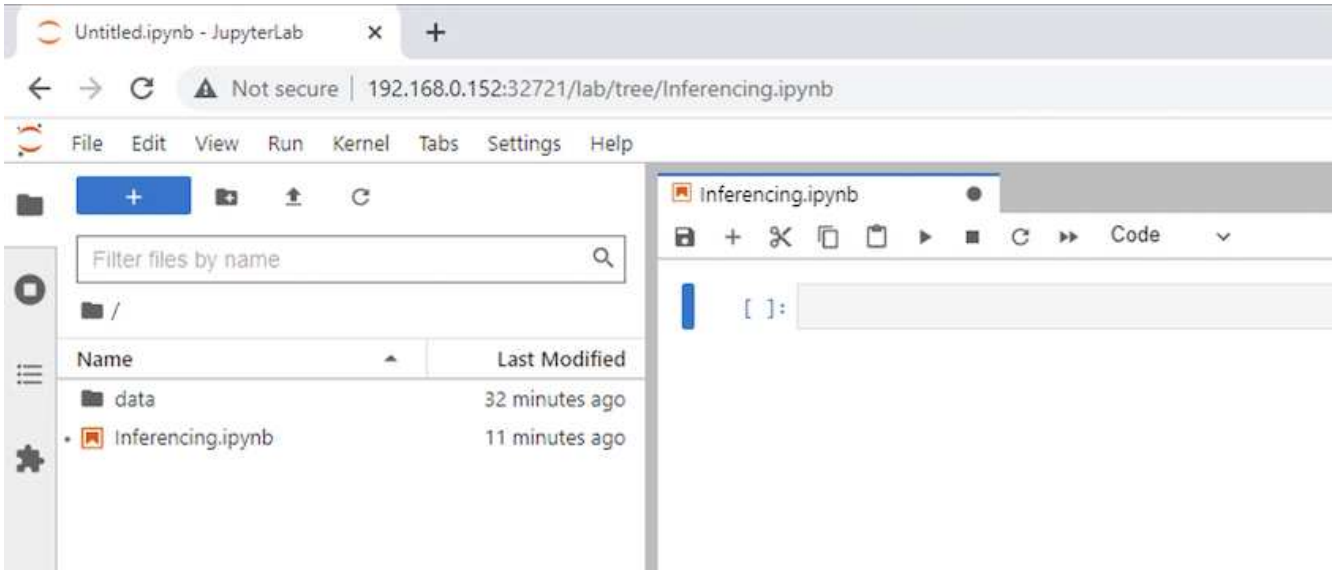
4. Greifen Sie auf den JupyterLab-Arbeitsbereich über die URL zu, die in der Ausgabe des angegeben ist `create jupyterlab` Befehl. Das Datenverzeichnis stellt das persistente Volume dar, das in den Arbeitsbereich gemountet wurde.



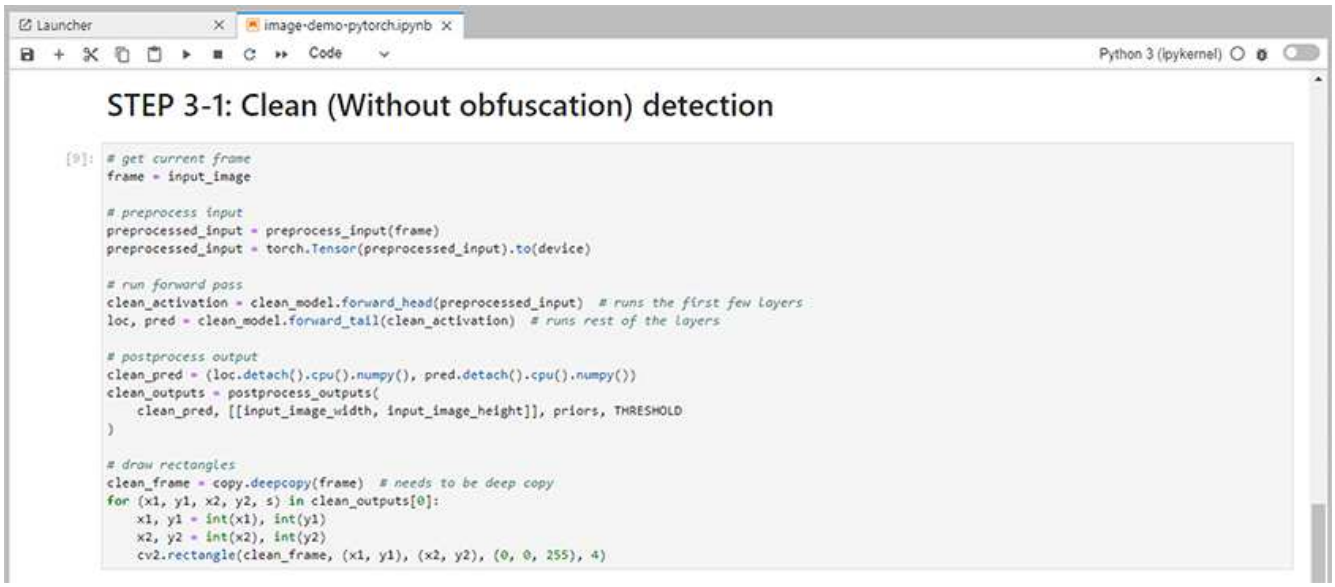
5. Öffnen Sie das `data` Verzeichnis erstellen und die Dateien hochladen, auf denen die Inferenz durchgeführt werden soll. Wenn Dateien in das Datenverzeichnis hochgeladen werden, werden sie automatisch auf dem persistenten Volume gespeichert, das in den Arbeitsbereich eingebunden wurde. Um Dateien hochzuladen, klicken Sie wie im folgenden Bild gezeigt auf das Symbol Dateien hochladen.

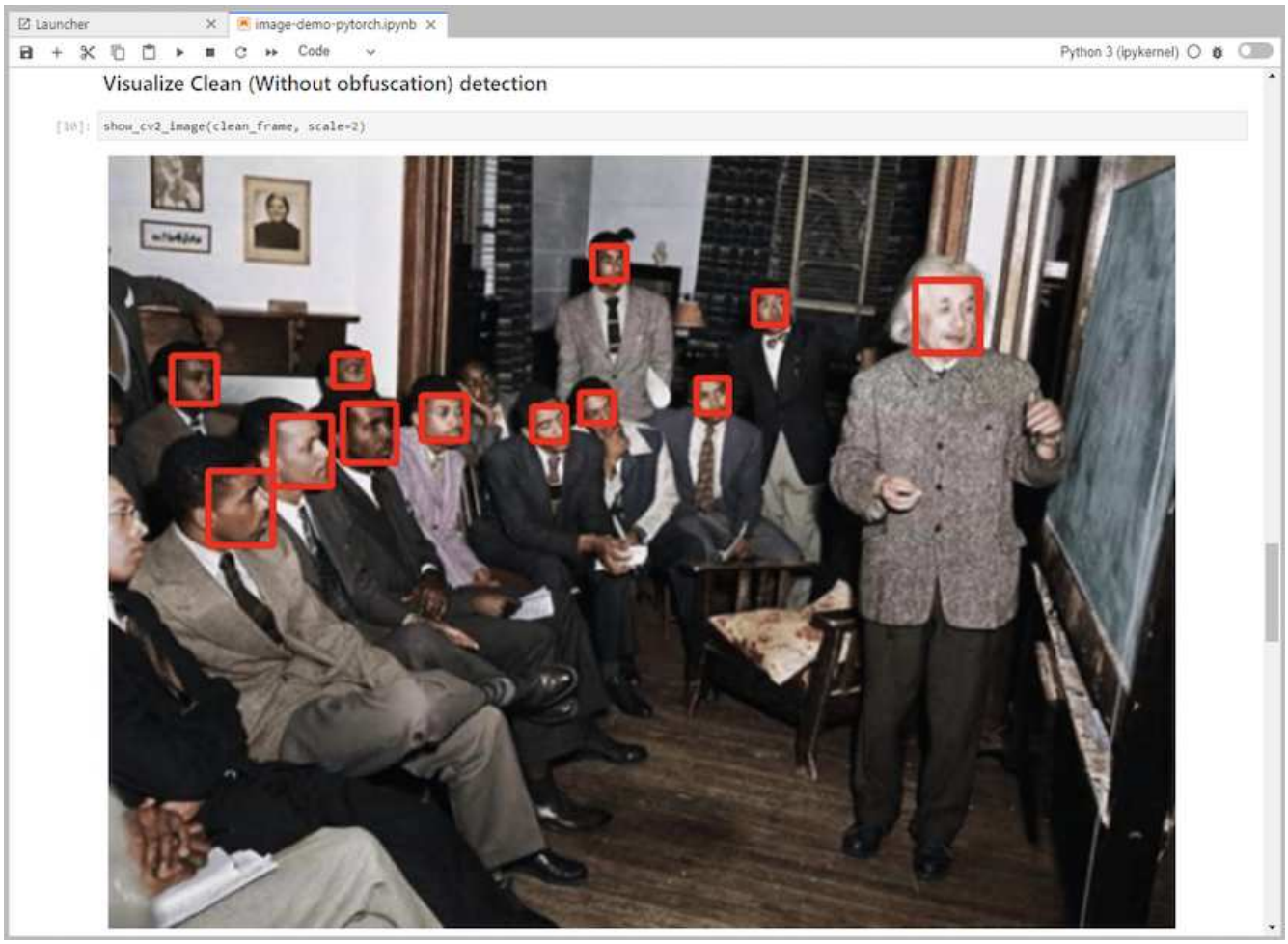


6. Kehren Sie zum Verzeichnis der obersten Ebene zurück und erstellen Sie ein neues Notebook.

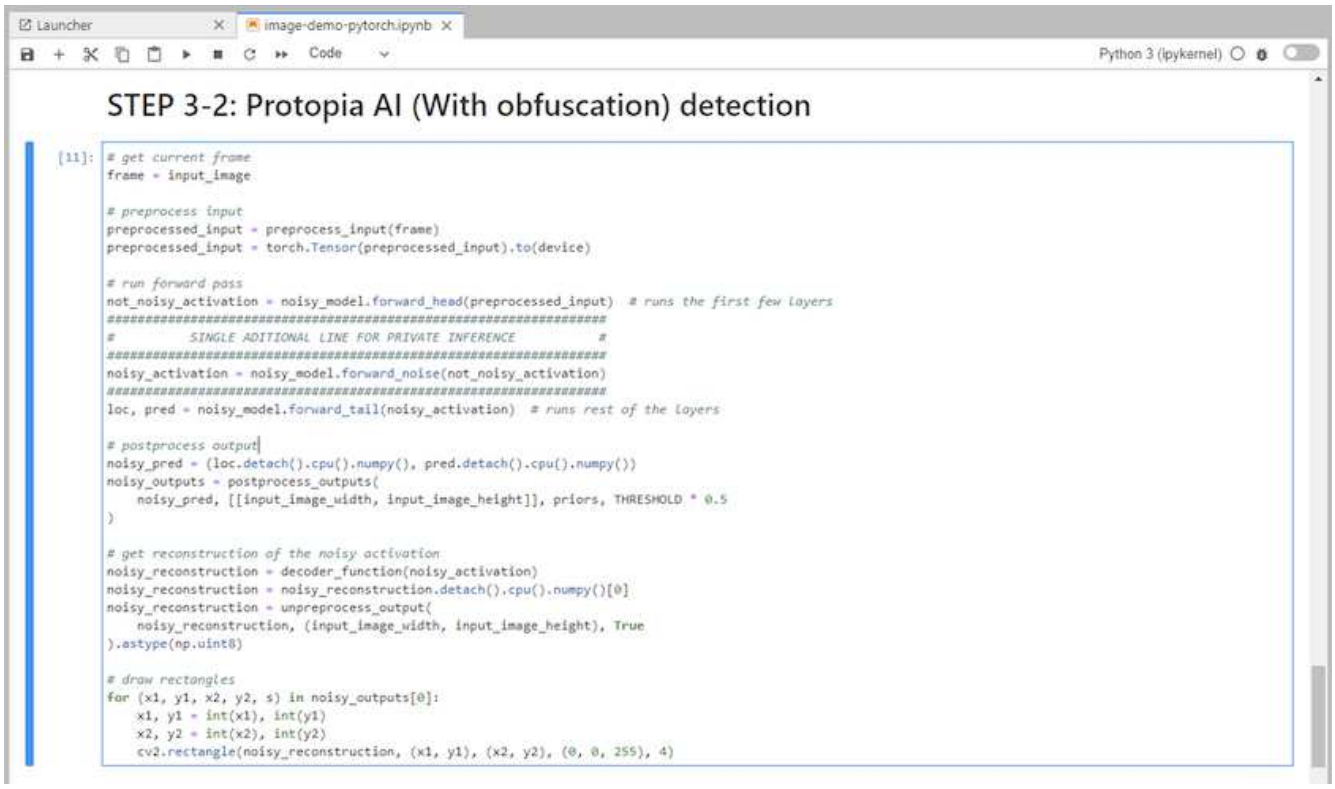


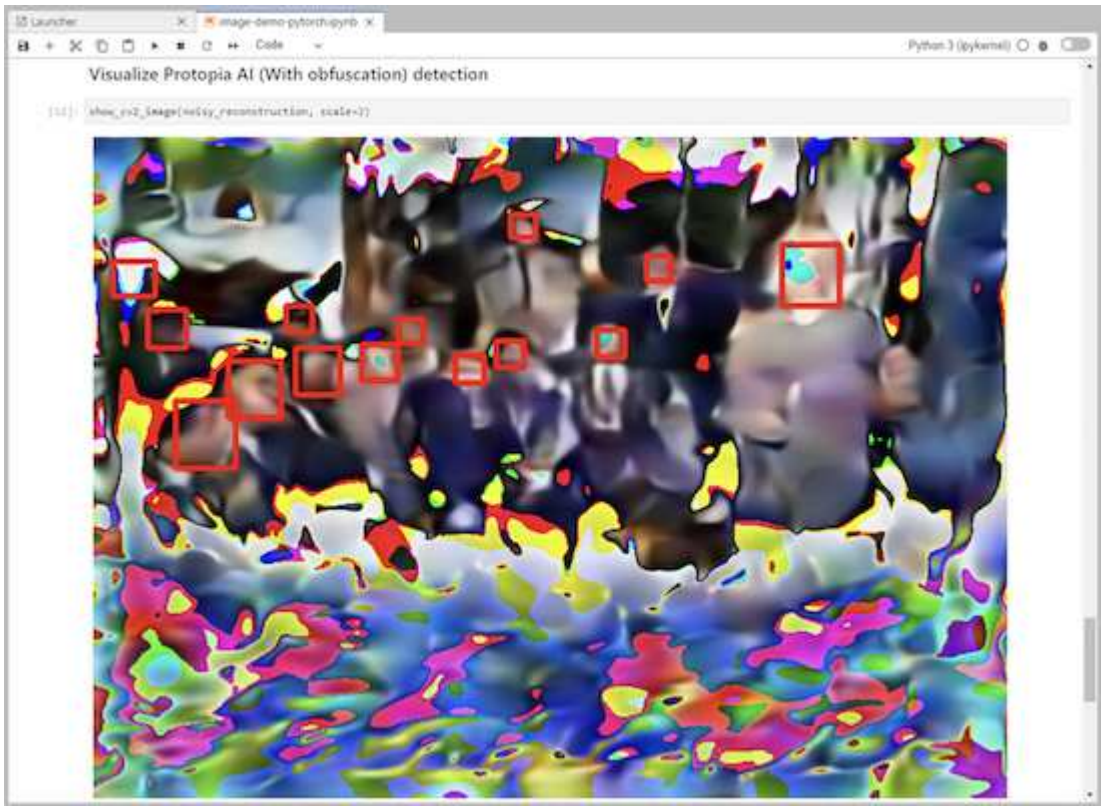
7. Fügen Sie dem Notebook den Inferenzcode hinzu. Im folgenden Beispiel wird der Inferenzcode für einen Anwendungsfall der Bilderkennung gezeigt.





8. Fügen Sie Protopia Obfuscation in Ihren Inferenzcode ein. Protopia arbeitet direkt mit Kunden zusammen, um eine anwendungsspezifische Dokumentation bereitzustellen und befindet sich außerhalb des Umfangs dieses technischen Berichts. Das folgende Beispiel zeigt den Inferenzcode für eine Bilderkennung, wobei Protopia obfuscation hinzugefügt wurde.





Szenario 2 – Batch-Inferenz auf Kubernetes

1. Kubernetes-Namespace für KI/ML-Inferenz-Workloads erstellen

```
$ kubectl create namespace inference
namespace/inference created
```

2. Verwenden Sie das NetApp DataOps Toolkit, um ein persistentes Volume für die Daten bereitzustellen, auf denen Sie die Inferenz ausführen.

```
$ netapp_dataops_k8s_cli.py create volume --namespace=inference --pvc
-name=inference-data --size=50Gi
Creating PersistentVolumeClaim (PVC) 'inference-data' in namespace
'inference'.
PersistentVolumeClaim (PVC) 'inference-data' created. Waiting for
Kubernetes to bind volume to PVC.
Volume successfully created and bound to PersistentVolumeClaim (PVC)
'inference-data' in namespace 'inference'.
```

3. Füllen Sie das neue persistente Volume mit den Daten, auf denen Sie die Inferenz durchführen möchten.

   Es gibt verschiedene Methoden zum Laden von Daten auf ein PVC. Wenn Ihre Daten aktuell in einer S3-kompatiblen Objekt-Storage-Plattform wie NetApp StorageGRID oder Amazon S3 gespeichert sind, können Sie verwenden "Funktionen des NetApp DataOps Toolkit S3 Data Mover". Eine weitere einfache Methode ist es, einen JupyterLab-Arbeitsbereich zu erstellen und dann Dateien über die JupyterLab-Webschnittstelle hochzuladen, wie in den Schritten 3 bis 5 im Abschnitt “ beschriebenSzenario 1 – On-Demand-Inferenz in JupyterLab.“

4. Erstellen eines Kubernetes-Jobs für die Batch-Inferenz. Das folgende Beispiel zeigt einen Batch-Inferenzauftrag für einen Anwendungsfall zur Bilderkennung. Dieser Job führt die Inferenz auf jedem Bild in einem Satz von Bildern durch und schreibt Metriken zur Inferenzgenauigkeit.

```
$ vi inference-job-raw.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: netapp-inference-raw
  namespace: inference
spec:
  backoffLimit: 5
  template:
    spec:
      volumes:
      - name: data
        persistentVolumeClaim:
          claimName: inference-data
      - name: dshm
        emptyDir:
          medium: Memory
      containers:
      - name: inference
        image: netapp-protopia-inference:latest
        imagePullPolicy: IfNotPresent
        command: ["python3", "run-accuracy-measurement.py", "--dataset",
"/data/netapp-face-detection/FDDB"]
        resources:
          limits:
            nvidia.com/gpu: 2
        volumeMounts:
        - mountPath: /data
          name: data
        - mountPath: /dev/shm
          name: dshm
      restartPolicy: Never
$ kubectl create -f inference-job-raw.yaml
job.batch/netapp-inference-raw created
```

5. Bestätigen Sie, dass der Inferenzauftrag erfolgreich abgeschlossen wurde.

```
$ kubectl -n inference logs netapp-inference-raw-255sp
100%|██████████| 89/89 [00:52<00:00,  1.68it/s]
Reading Predictions : 100%|██████████| 10/10 [00:01<00:00,  6.23it/s]
Predicting ... : 100%|██████████| 10/10 [00:16<00:00,  1.64s/it]
==================== Results ====================
FDDB-fold-1 Val AP: 0.9491256561145955
FDDB-fold-2 Val AP: 0.9205024466101926
FDDB-fold-3 Val AP: 0.9253013871078468
FDDB-fold-4 Val AP: 0.9399781485863011
FDDB-fold-5 Val AP: 0.9504280149478732
FDDB-fold-6 Val AP: 0.9416473519339292
FDDB-fold-7 Val AP: 0.9241631566241117
FDDB-fold-8 Val AP: 0.9072663297546659
FDDB-fold-9 Val AP: 0.9339648715035469
FDDB-fold-10 Val AP: 0.9447707905560152
FDDB Dataset Average AP: 0.9337148153739079
--------------------------------------------------
mAP: 0.9337148153739079
```

6. Fügen Sie Protopia Obfuscation zu Ihren Inferenz Job. Die anwendungsspezifische Anleitung zum Hinzufügen von Protopia-Obfuskation kann direkt aus Protopia gefunden werden, die nicht im Rahmen dieses technischen Berichts liegt. Das folgende Beispiel zeigt einen Batch-Inferenzauftrag für eine Gesichtserkennung Anwendungsfall mit Protopia-Obfuscation, die durch die Verwendung eines ALPHAWERTS von 0.8 hinzugefügt wurde. Bei diesem Job wird die Protopia-Obfuskation vor der Durchführung der Inferenz für jedes Bild in einer Reihe von Bildern angewendet und dann Kenngrößen für die Inferenzgenauigkeit geschrieben.

   Wir haben diesen Schritt für ALPHA-Werte 0.05, 0.1, 0.2, 0.4, 0.6, wiederholt. 0.8, 0.9 und 0.95. Die Ergebnisse sehen Sie in "„Vergleich der Genauigkeit bei der Inferenzierung“."

```
$ vi inference-job-protopia-0.8.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: netapp-inference-protopia-0.8
  namespace: inference
spec:
  backoffLimit: 5
  template:
    spec:
      volumes:
      - name: data
        persistentVolumeClaim:
          claimName: inference-data
      - name: dshm
        emptyDir:
          medium: Memory
      containers:
      - name: inference
        image: netapp-protopia-inference:latest
        imagePullPolicy: IfNotPresent
        env:
        - name: ALPHA
          value: "0.8"
        command: ["python3", "run-accuracy-measurement.py", "--dataset",
"/data/netapp-face-detection/FDDB", "--alpha", "$(ALPHA)", "--noisy"]
        resources:
          limits:
            nvidia.com/gpu: 2
        volumeMounts:
        - mountPath: /data
          name: data
        - mountPath: /dev/shm
          name: dshm
      restartPolicy: Never
$ kubectl create -f inference-job-protopia-0.8.yaml
job.batch/netapp-inference-protopia-0.8 created
```

7. Bestätigen Sie, dass der Inferenzauftrag erfolgreich abgeschlossen wurde.

```
$ kubectl -n inference logs netapp-inference-protopia-0.8-b4dkz
100%|██████████| 89/89 [01:05<00:00,  1.37it/s]
Reading Predictions : 100%|██████████| 10/10 [00:02<00:00,  3.67it/s]
Predicting ... : 100%|██████████| 10/10 [00:22<00:00,  2.24s/it]
=================== Results ===================
FDDB-fold-1 Val AP: 0.8953066115834589
FDDB-fold-2 Val AP: 0.8819580264029936
FDDB-fold-3 Val AP: 0.8781107458462862
FDDB-fold-4 Val AP: 0.9085731346308461
FDDB-fold-5 Val AP: 0.9166445508275378
FDDB-fold-6 Val AP: 0.9101178994188819
FDDB-fold-7 Val AP: 0.8383443678423771
FDDB-fold-8 Val AP: 0.8476311547659464
FDDB-fold-9 Val AP: 0.8739624502111121
FDDB-fold-10 Val AP: 0.8905468076424851
FDDB Dataset Average AP: 0.8841195749171925
================================================
mAP: 0.8841195749171925
```

Szenario 3 – NVIDIA Triton Inferenz Server

1. Kubernetes-Namespace für KI/ML-Inferenz-Workloads erstellen

```
$ kubectl create namespace inference
namespace/inference created
```

2. Verwenden Sie das NetApp DataOps Toolkit, um ein persistentes Volume bereitzustellen, das als Modell-Repository für den NVIDIA Triton Inference Server verwendet werden kann.

```
$ netapp_dataops_k8s_cli.py create volume --namespace=inference --pvc
-name=triton-model-repo --size=100Gi
Creating PersistentVolumeClaim (PVC) 'triton-model-repo' in namespace
'inference'.
PersistentVolumeClaim (PVC) 'triton-model-repo' created. Waiting for
Kubernetes to bind volume to PVC.
Volume successfully created and bound to PersistentVolumeClaim (PVC)
'triton-model-repo' in namespace 'inference'.
```

3. Sie können Ihr Modell auf dem neuen persistenten Volume in einem speichern "Formatieren" Das wird vom NVIDIA Triton Inferenz Server erkannt.

   Es gibt verschiedene Methoden zum Laden von Daten auf ein PVC. Eine einfache Methode ist es, einen JupyterLab-Arbeitsbereich zu erstellen und dann Dateien über die JupyterLab-Webschnittstelle hochzuladen, wie in den Schritten 3 bis 5 in “ beschriebenSzenario 1 – On-Demand-Inferenz in

JupyterLab. „

4. Verwenden Sie das NetApp DataOps Toolkit, um eine neue NVIDIA Triton Inferenz Server-Instanz zu implementieren.

```
$ netapp_dataops_k8s_cli.py create triton-server --namespace=inference
--server-name=netapp-inference --model-repo-pvc-name=triton-model-repo
Creating Service 'ntap-dsutil-triton-netapp-inference' in namespace
'inference'.
Service successfully created.
Creating Deployment 'ntap-dsutil-triton-netapp-inference' in namespace
'inference'.
Deployment 'ntap-dsutil-triton-netapp-inference' created.
Waiting for Deployment 'ntap-dsutil-triton-netapp-inference' to reach
Ready state.
Deployment successfully created.
Server successfully created.
Server endpoints:
http: 192.168.0.152: 31208
grpc: 192.168.0.152: 32736
metrics: 192.168.0.152: 30009/metrics
```

5. Verwenden Sie ein Triton Client SDK zur Durchführung einer Inferenz. Im folgenden Python-Code-Auszug wird das Triton Python-Client-SDK verwendet, um eine Inferenzaufgabe für einen Anwendungsfall zur Gesichtserkennung durchzuführen. Dieses Beispiel nennt die Triton API und führt ein Bild zur Inferenz durch. Der Triton Inference Server erhält dann die Anfrage, ruft das Modell auf und gibt die Inferenzausgabe als Teil der API-Ergebnisse zurück.

```
# get current frame
frame = input_image
# preprocess input
preprocessed_input = preprocess_input(frame)
preprocessed_input = torch.Tensor(preprocessed_input).to(device)
# run forward pass
clean_activation = clean_model_head(preprocessed_input)  # runs the
first few layers
##########################################################################
##############
#          pass clean image to Triton Inference Server API for
inferencing           #
##########################################################################
##############
triton_client =
httpclient.InferenceServerClient(url="192.168.0.152:31208",
verbose=False)
model_name = "face_detection_base"
```

```
inputs = []
outputs = []
inputs.append(httpclient.InferInput("INPUT__0", [1, 128, 32, 32],
"FP32"))
inputs[0].set_data_from_numpy(clean_activation.detach().cpu().numpy(),
binary_data=False)
outputs.append(httpclient.InferRequestedOutput("OUTPUT__0",
binary_data=False))
outputs.append(httpclient.InferRequestedOutput("OUTPUT__1",
binary_data=False))
results = triton_client.infer(
    model_name,
    inputs,
    outputs=outputs,
    #query_params=query_params,
    headers=None,
    request_compression_algorithm=None,
    response_compression_algorithm=None)
#print(results.get_response())
statistics =
triton_client.get_inference_statistics(model_name=model_name,
headers=None)
print(statistics)
if len(statistics["model_stats"]) != 1:
    print("FAILED: Inference Statistics")
    sys.exit(1)

loc_numpy = results.as_numpy("OUTPUT__0")
pred_numpy = results.as_numpy("OUTPUT__1")
##########################################################################
##############
# postprocess output
clean_pred = (loc_numpy, pred_numpy)
clean_outputs = postprocess_outputs(
    clean_pred, [[input_image_width, input_image_height]], priors,
THRESHOLD
)
# draw rectangles
clean_frame = copy.deepcopy(frame)  # needs to be deep copy
for (x1, y1, x2, y2, s) in clean_outputs[0]:
    x1, y1 = int(x1), int(y1)
    x2, y2 = int(x2), int(y2)
    cv2.rectangle(clean_frame, (x1, y1), (x2, y2), (0, 0, 255), 4)
```

6. Fügen Sie Protopia Obfuscation in Ihren Inferenzcode ein. Die anwendungsspezifische Anleitung zum Hinzufügen von Protopia-Obfuskation kann direkt aus Protopia gefunden werden; dieser Vorgang liegt

jedoch außerhalb des Geltungsbereichs dieses technischen Berichts. Das folgende Beispiel zeigt denselben Python-Code, der im vorhergehenden Schritt 5, jedoch mit Protopia Obfuscation hinzugefügt wird.

Beachten Sie, dass die Protopia-Obfuskation auf das Bild angewendet wird, bevor es an die Triton-API übergeben wird. So verlässt das nicht-verschleierte Bild nie die lokale Maschine. Nur das obfuscated Image wird über das Netzwerk übertragen. Dieser Workflow eignet sich für Anwendungsfälle, in denen Daten in einer vertrauenswürdigen Zone erfasst und dann zur Inferenz außerhalb der vertrauenswürdigen Zone weitergeleitet werden müssen. Ohne Protopia Obfuscation ist es nicht möglich, diesen Workflow ohne sensible Daten zu implementieren, die jemals die vertrauenswürdige Zone verlassen.

```
# get current frame
frame = input_image
# preprocess input
preprocessed_input = preprocess_input(frame)
preprocessed_input = torch.Tensor(preprocessed_input).to(device)
# run forward pass
not_noisy_activation = noisy_model_head(preprocessed_input)  # runs the
first few layers
#####################################################################
#          obfuscate image locally prior to inferencing             #
#          SINGLE ADITIONAL LINE FOR PRIVATE INFERENCE               #
#####################################################################
noisy_activation = noisy_model_noise(not_noisy_activation)
#####################################################################
#########################################################################
###################
#          pass obfuscated image to Triton Inference Server API for
inferencing           #
#########################################################################
###################
triton_client =
httpclient.InferenceServerClient(url="192.168.0.152:31208",
verbose=False)
model_name = "face_detection_noisy"
inputs = []
outputs = []
inputs.append(httpclient.InferInput("INPUT__0", [1, 128, 32, 32],
"FP32"))
inputs[0].set_data_from_numpy(noisy_activation.detach().cpu().numpy(),
binary_data=False)
outputs.append(httpclient.InferRequestedOutput("OUTPUT__0",
binary_data=False))
outputs.append(httpclient.InferRequestedOutput("OUTPUT__1",
binary_data=False))
results = triton_client.infer(
    model_name,
```

```
    inputs,
    outputs=outputs,
    #query_params=query_params,
    headers=None,
    request_compression_algorithm=None,
    response_compression_algorithm=None)
#print(results.get_response())
statistics =
triton_client.get_inference_statistics(model_name=model_name,
headers=None)
print(statistics)
if len(statistics["model_stats"]) != 1:
    print("FAILED: Inference Statistics")
    sys.exit(1)

loc_numpy = results.as_numpy("OUTPUT__0")
pred_numpy = results.as_numpy("OUTPUT__1")
########################################################################
###################

# postprocess output
noisy_pred = (loc_numpy, pred_numpy)
noisy_outputs = postprocess_outputs(
    noisy_pred, [[input_image_width, input_image_height]], priors,
THRESHOLD * 0.5
)
# get reconstruction of the noisy activation
noisy_reconstruction = decoder_function(noisy_activation)
noisy_reconstruction = noisy_reconstruction.detach().cpu().numpy()[0]
noisy_reconstruction = unpreprocess_output(
    noisy_reconstruction, (input_image_width, input_image_height), True
).astype(np.uint8)
# draw rectangles
for (x1, y1, x2, y2, s) in noisy_outputs[0]:
    x1, y1 = int(x1), int(y1)
    x2, y2 = int(x2), int(y2)
    cv2.rectangle(noisy_reconstruction, (x1, y1), (x2, y2), (0, 0, 255),
4)
```

Vergleich der Inferenzgenauigkeit

Für diese Validierung haben wir mithilfe eines Satzes an RAW-Bildern Inferenz für einen Anwendungsfall der Bilderkennung durchgeführt. Dann führten wir dieselbe Inferenzaufgabe auf demselben Bildersatz durch, wobei vor der Inferenz die Protopia-Obfuskation hinzugefügt wurde. Wir haben die Aufgabe mit unterschiedlichen Werten

VON ALPHA für die Komponente Protopia obfuscation wiederholt. Im Zusammenhang mit der Protopia-Obfuskation stellt der ALPHA-Wert die Menge der aufgetragenen Obfuskation dar, wobei ein höherer ALPHA-Wert ein höheres Maß an Obfuskation darstellt. Dann haben wir die Genauigkeit der Inferenz in diesen verschiedenen Durchläufen verglichen.

Die folgenden beiden Tabellen enthalten Details zu unserem Anwendungsfall und geben die Ergebnisse an.

Protopia arbeitet direkt mit den Kunden zusammen, um den passenden ALPHA-Wert für einen bestimmten Anwendungsfall zu ermitteln.

Komponente	**Details**
Modell	FaceBoxes (PyTorch) -
Datensatz	FDDB-Datensatz

Protopia obfuscation	**ALPHA**	**Genauigkeit**
Nein	K. A.	0.9337148153739079
Ja.	0.05	0.9028766627325002
Ja.	0.1	0.9024301009661478
Ja.	0.2	0.9081836283186224
Ja.	0.4	0.9073066107482036
Ja.	0.6	0.8847816568680239
Ja.	0.8	0.8841195749171925
Ja.	0.9	0.8455427675252052
Ja.	0.95	0.8455427675252052

Obfuskationsgeschwindigkeit

Für diese Validierung haben wir die Protopia-Obfuskation fünfmal auf ein 1920 x 1080 Pixel-Bild angewendet und die Zeit gemessen, die für den obfuscation-Schritt erforderlich war, um jedes Mal abzuschließen.

Wir verwendeten PyTorch, der auf einer einzelnen NVIDIA V100 GPU ausgeführt wurde, um den Verschleierung anzuwenden, und wir löchten den GPU-Cache zwischen den Durchläufen. Der obfuscation-Schritt dauerte 5,47 ms, 5,27 ms, 4,54 ms, 5,24 ms bzw. 4,84 ms, um in den fünf Durchläufen abzuschließen. Die durchschnittliche Geschwindigkeit betrug 5,072 ms.

Schlussfolgerung

Die Daten sind in drei Bundesstaaten unterwegs: Im Ruhezustand, während der Übertragung und im Computing. Ein wichtiger Bestandteil jedes KI-Inferenz-Services sollte der Schutz der Daten vor Bedrohungen während des gesamten Prozesses sein. Der Schutz von Daten während der Inferenz ist von großer Bedeutung, da der Prozess

private Informationen über externe Kunden und das Unternehmen, das den Inferenzservice bereitstellt, offenlegen kann. Protopia AI ist eine nicht aufdringliche rein softwarebasierte Lösung für vertrauliche KI-Inferenz auf dem heutigen Markt. Mit Protopia werden KI nur die transformierten Informationen in den Datensätzen gespeist, die für die Durchführung der KI/ML-Aufgabe am wichtigsten sind – und nicht mehr. Diese stochastische Transformation ist keine Form der Maskierung und basiert auf mathematisch veränderten Darstellung der Daten durch die Verwendung kuratierter Geräusche.

NetApp Storage-Systeme mit ONTAP-Funktionen bieten dieselbe oder bessere Performance wie lokaler SSD-Storage und bieten in Kombination mit dem NetApp DataOps Toolkit Data Scientists, Data Engineers, KI/ML-Entwickler und Entscheidungsträger in Unternehmen oder IT die folgenden Vorteile:

- Problemlose gemeinsame Nutzung von Daten zwischen KI-Systemen, Big-Data-Analysen und anderen geschäftskritischen Systemen Diese gemeinsame Nutzung von Daten verringert den Infrastruktur-Overhead, verbessert die Performance und optimiert das Datenmanagement im gesamten Unternehmen.
- Unabhängig skalierbare Computing- und Storage-Ressourcen minimieren Kosten und verbessern die Ressourcenauslastung.
- Optimierte Entwicklungs- und Implementierungs-Workflows mithilfe integrierter Snapshot Kopien und Klone für sofortige und platzsparende Benutzer-Workspaces, integrierte Versionskontrolle und automatisierte Implementierung
- Datensicherung und Data Governance der Enterprise-Klasse für Disaster Recovery, Business Continuity und gesetzliche Vorschriften.
- Vereinfachtes Aufrufen von Datenmanagement-Prozessen; schnelle Erstellung von Snapshot Kopien von Arbeitsbereichen für Data Scientists zur Sicherung und Rückverfolgbarkeit über das NetApp DataOps Toolkit in Jupyter Notebooks.

Die Lösung von NetApp und Protopia bietet eine flexible Scale-out-Architektur, die sich ideal für KI-Inferenz-Implementierungen der Enterprise-Klasse eignet. Sie ermöglicht Datensicherung und Datenschutz für sensible Informationen, bei denen vertrauliche KI-Inferenz-Anforderungen mit verantwortlichen AI-Praktiken sowohl in On-Premises- als auch in Hybrid-Cloud-Implementierungen erfüllt werden können.

Weitere Informationen und Bestätigungen

Weitere Informationen zu den in diesem Dokument beschriebenen Daten finden Sie in den folgenden Dokumenten bzw. auf den folgenden Websites:

- NetApp ONTAP Datenmanagement-Software – ONTAP Informationsbibliothek

  http://mysupport.netapp.com/documentation/productlibrary/index.html?productID=62286

- NetApp persistenter Storage für Container – NetApp Trident

  "https://netapp.io/persistent-storage-provisioner-for-kubernetes/"

- NetApp DataOps Toolkit

  "https://github.com/NetApp/netapp-dataops-toolkit"

- NetApp persistenter Storage für Container – NetApp Trident

"https://netapp.io/persistent-storage-provisioner-for-kubernetes/"

- Protopia AI – vertrauliche Inferenz

  "https://protopia.ai/blog/protopia-ai-takes-on-the-missing-link-in-ai-privacy-confidential-inference/"

- NetApp BlueXP Kopie und Synchronisierung

  "https://docs.netapp.com/us-en/occm/concept_cloud_sync.html#how-cloud-sync-works"

- NVIDIA Triton Inferenz Server

  "https://developer.nvidia.com/nvidia-triton-inference-server"

- NVIDIA Triton Inference Server Documentation

  "https://docs.nvidia.com/deeplearning/triton-inference-server/index.html"

- FaceBoxes in PyTorch

  "https://github.com/zisianw/FaceBoxes.PyTorch"

Danksagungen

- Mark Cates, Principal Product Manager, NetApp
- Sufian Ahmad, Technical Marketing Engineer, NetApp
- Hadi Esmaeilzadeh, Chief Technology Officer und Professor, Protopia AI

Copyright-Informationen

Markeninformationen