책임 **AI** 및 기밀 추론 **- NetApp AI** 및 **Protopia** 이미지 변환

NetApp Solutions

NetApp
April 20, 2024

This PDF was generated from https://docs.netapp.com/ko-kr/netapp-solutions/ai/ai-protopia-overview.html on April 20, 2024. Always check docs.netapp.com for the latest.

목차

책임 AI 및 기밀 추론 - NetApp AI 및 Protopia 이미지 변환 . . . 1
TR-4928: AI 및 기밀 추론 지원 - NetApp AI 및 Protopia 이미지 및 데이터 변환 . . . 1
솔루션 영역 . . . 3
기술 개요 . . . 4
테스트 및 검증 계획 . . . 9
구성을 테스트합니다 . . . 9
테스트 절차 . . . 9
추론 정확도 비교 . . . 24
난독화 속도 . . . 25
결론 . . . 25
추가 정보 및 승인 정보를 찾을 수 있는 위치 . . . 26

책임 AI 및 기밀 추론 - NetApp AI 및 Protopia 이미지 변환

TR-4928: AI 및 기밀 추론 지원 - NetApp AI 및 Protopia 이미지 및 데이터 변환

사티아가라잔, 마이클 오글즈비야, NetApp 병훈 안, 제니퍼 와겐버그, 프로토피아

시각적 해석은 이미지 캡처와 이미지 처리의 등장과 함께 커뮤니케이션의 핵심 요소가 되었습니다. 디지털 이미지 처리의 인공 지능(AI)은 암 및 기타 질병 식별을 위한 의료 분야, 환경 위험 연구, 패턴 인식, 범죄 퇴치를 위한 비디오 처리 등을 위한 지리공간 시각적 분석 등 새로운 비즈니스 기회를 제공합니다. 그러나 이 기회에는 특별한 책임이 있습니다.

AI에 대한 조직의 의사 결정이 내려질수록 데이터에 대한 개인 정보 보호, 보안, 법률, 윤리 및 규제 문제와 관련된 위험에 노출될 위험은 커집니다. 책임감 있는 AI를 통해 기업과 정부 조직이 대규모 기업에서 AI에 중요한 신뢰 및 거버넌스 구축을 지원할 수 있습니다. 이 문서에서는 Protopia 데이터 난독 처리 소프트웨어와 함께 NetApp 데이터 관리 기술을 사용하여 중요한 데이터를 민화하고 위험과 윤리적 문제를 줄임으로써 NetApp에서 검증한 AI 추론 솔루션에 대해 설명합니다.

소비자와 기업 모두 다양한 디지털 장치를 사용하여 매일 수백만 개의 이미지가 생성됩니다. 결과적으로 데이터가 폭발적으로 증가하고 컴퓨팅 작업 부하가 발생함에 따라 기업은 규모와 효율성을 위해 클라우드 컴퓨팅 플랫폼으로 전환하게 됩니다. 한편, 이미지 데이터에 포함된 민감한 정보에 대한 개인 정보 보호는 퍼블릭 클라우드로 이전될 때 발생합니다. 보안 및 개인 정보 보호 보장이 없기 때문에 이미지 처리 AI 시스템을 배포하는 데 주요 장애물이 되고 있습니다.

또한 이 있습니다 "삭제권" GDPR에 따라, 조직에서 모든 개인 데이터를 삭제하도록 요청할 수 있는 개인의 권리. 또한 도 있습니다 "개인정보보호법"이것은 공정 정보 관리 코드를 설정합니다. 사진과 같은 디지털 이미지는 데이터를 수집, 처리 및 지우는 방법을 제어하는 GDPR의 개인 데이터를 구성할 수 있습니다. 그렇지 않으면 GDPR을 준수하지 않아 조직에 심각한 피해를 줄 수 있는 규정 준수 위반에 대한 무거운 벌금이 부과될 수 있습니다. 개인 정보 보호 원칙은 머신 러닝(ML) 및 딥 러닝(DL) 모델 예측의 공정성을 보장하고 개인 정보 보호 또는 규정 준수 위반과 관련된 위험을 줄이는 책임 있는 AI 구현의 근간입니다.

이 문서에서는 개인 정보 보호 유지 및 책임 있는 AI 솔루션 배포와 관련된 이미지 난독 처리를 사용하는 경우와 사용하지 않는 세 가지 시나리오에서 검증된 설계 솔루션을 설명합니다.

- * 시나리오 1. * Jupyter 노트북 내 필요 시 추론.
- * 시나리오 2. * Kubernetes의 배치 추론
- * 시나리오 3. * NVIDIA Triton 추론 서버

이 솔루션에서는 FDDB(얼굴 인식 데이터 세트 및 벤치마크)를 사용합니다. FDDB는 페이스 보스의 구현을 위한 PyTorch 기계 학습 프레임워크와 함께, 구속되지 않은 얼굴 감지 문제를 연구하도록 설계된 얼굴 영역 데이터 세트입니다. 이 데이터 세트에는 다양한 해상도의 2845 이미지 세트에 있는 5171면에 대한 주석이 포함되어 있습니다. 또한 이 기술 보고서에서는 이 솔루션이 적용 가능한 상황에 대해 NetApp 고객 및 현장 엔지니어를 통해 수집된 일부 솔루션 영역 및 관련 사용 사례를 소개합니다.

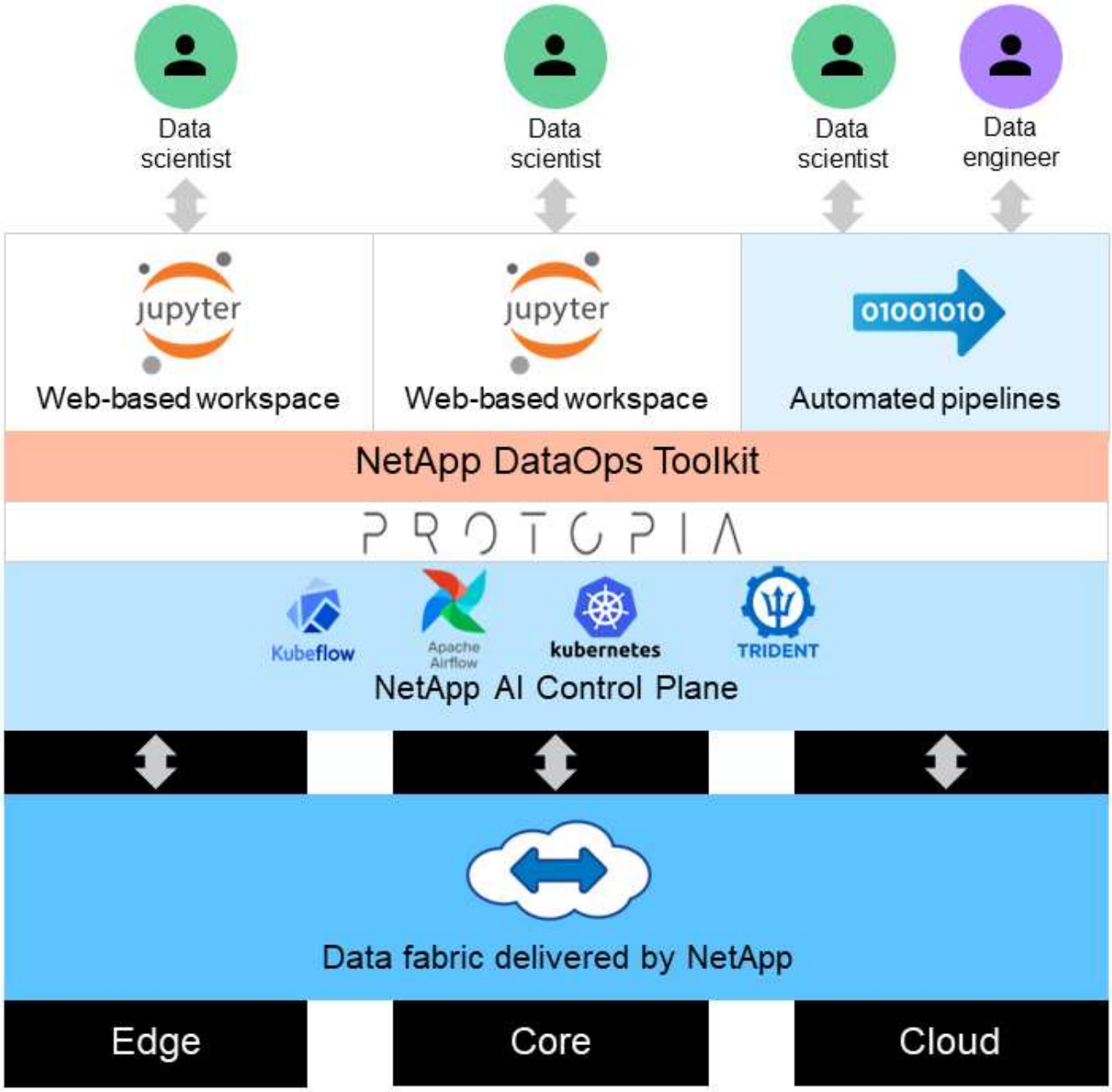
대상

이 기술 보고서는 다음 대상자를 대상으로 합니다.

- 책임 있는 AI를 설계 및 구축하고 공공 장소에서의 얼굴 이미지 처리와 관련된 데이터 보호 및 개인 정보 문제를 처리하고자 하는 비즈니스 리더 및 엔터프라이즈 설계자
- 개인 정보를 보호하고 유지하고자 하는 데이터 과학자, 데이터 엔지니어, AI/기계 학습(ML) 연구자 및 AI/ML 시스템 개발자.
- GDPR, CCPA 또는 국방부(DoD) 및 정부 조직의 개인정보 보호법과 같은 규정 표준을 준수하는 AI/ML 모델 및 애플리케이션에 대한 데이터 난독 처리 솔루션을 설계하는 엔터프라이즈 설계자
- 중요한 정보를 보호하는 딥 러닝(DL) 및 AI/ML/DL 추론 모델을 효율적으로 구축하는 방법을 찾고 있는 데이터 과학자 및 AI 엔지니어
- 에지 장치 관리자 및 에지 서버 관리자는 에지 추론 모델의 구축과 관리를 담당합니다.

솔루션 아키텍처

이 솔루션은 GPU의 처리 성능과 기존 CPU를 함께 사용하여 대규모 데이터 세트에서 AI 워크로드를 실시간으로 일괄 처리하고 추론하도록 설계되었습니다. 이 검증에서는 책임 있는 AI 배포를 원하는 조직에 필요한 ML에 대한 개인 정보 보호 추론과 최적의 데이터 관리를 보여줍니다. 이 솔루션은 Jupyter Lab 및 CLI 인터페이스를 사용하여 코어 사내 NetApp ONTAP AI와 상호 연결된 에지 및 클라우드 컴퓨팅을 위한 단일 또는 다중 노드 Kubernetes 플랫폼, NetApp DataOps 툴킷, Protopia 난독화 소프트웨어에 적합한 아키텍처를 제공합니다. 다음 그림에서는 DataOps Toolkit 및 Protopia를 지원하는 NetApp 기반의 Data Fabric에 대한 논리적 아키텍처 개요를 보여 줍니다.



Protopia 난독 처리 소프트웨어는 NetApp DataOps Toolkit에서 원활하게 실행되며 스토리지 서버를 떠나기 전에 데이터를 변환합니다.

솔루션 영역

디지털 이미지 프로세싱은 많은 이점을 제공하므로 많은 조직에서 시각적 표현과 관련된 데이터를 최대한 활용할 수 있습니다. 이 NetApp 및 Protopia 솔루션은 ML/DL 수명 주기 동안 AI/ML 데이터를 보호하고 민영화 할 수 있는 고유한 AI 추론 설계를 제공합니다. 고객은 이 모델을 통해 기밀 데이터에 대한 소유권을 유지하고, 개인 정보 보호와 관련된 문제를 완화하여 규모에 맞게 퍼블릭 또는 하이브리드 클라우드 구축 모델을 사용하거나, 에지에 AI 추론을 배포할 수 있습니다.

환경 인텔리전스

업계가 환경적 위험 영역에서 지리 공간 분석을 활용할 수 있는 방법은 여러 가지가 있습니다. 정부 및 공공사업 부서는 전염병 또는 산불과 같은 자연 재해 발생 시 공공 보건 및 기상 조건에 대한 실행 가능한 통찰력을 제공하여 일반에게 보다 나은 조언을 제공할 수 있습니다. 예를 들어, 영향을 받는 개인의 사생활을 침해하지 않고 공항 또는 병원과 같은 공공 장소에서 COVID-양성 환자를 식별하고 관련 당국과 인근 직원에게 필요한 안전 조치를 알릴 수 있습니다.

에지 장치 웨어러블

군대와 전쟁터에서 군인의 사생활을 보호하고 보호하면서 군인의 건강을 추적하고, 운전자의 행동을 모니터링하고, 군용 차량에 접근하는 안전 및 관련 위험에 대해 당국에게 경고하기 위해 엣지에서 AI 추론을 웨어러블 장치로 사용할 수 있습니다. 군대의 미래는 전투지 사물 인터넷(IoT)과 웨어러블 전투기용 사물 인터넷(IoMT)을 통해 첨단 기술로 나아가고 있습니다. 이 장비는 신속한 에지 컴퓨팅을 통해 군인들이 적들을 식별하고 전투에서 더 나은 성과를 달성할 수 있도록 지원합니다. 드론과 웨어러블 기어와 같은 에지 장치에서 수집된 시각적 데이터를 보호하고 보존하는 것은 해커와 적에게 항상 보안을 유지하는 데 매우 중요합니다.

비전투원 대피 작전

비전투원 대피 작전(Neos)은 미 국방부가 미국 시민과 국적을 대피시키기 위해, 국방부 민간인 직원, 그리고 적절한 안전한 피난처로 목숨을 걸고 있는 지정된 사람(HN)과 제3국 국민(TCN)을 대피시키기 위해 실시합니다. 현재 행정적 통제는 대부분 수작업을 통해 선별검사 프로세스를 비우는 데 사용됩니다. 그러나 AI/ML 비디오 난독 처리 기술과 결합된 고도로 자동화된 AI/ML 도구를 사용하면 비우는 ID, 비우는 방법 추적 및 위협 선별의 정확성, 보안 및 속도를 개선할 수 있습니다.

의료 및 생의학 연구

영상 처리는 컴퓨터 단층촬영(CT) 또는 자기공명영상(MRI)에서 얻은 3D 영상에서 외과적 계획을 위한 병리를 진단하는 데 사용됩니다. HIPAA 개인 정보 보호 규칙은 모든 개인 정보 및 사진과 같은 디지털 이미지에 대해 조직에서 데이터를 수집, 처리 및 지우는 방법을 규정합니다. HIPAA 세이프 하버 규정에 따라 데이터를 공유할 수 있는 자격을 갖추려면 전체 사진 이미지와 비교 이미지를 제거해야 합니다. 구조적 CT/MR 영상에서 개인의 얼굴 특징을 모호하게 하는 데 사용되는 식별 해제 또는 두개골 스트리핑 알고리즘과 같은 자동화 기술은 생물 의학 연구 기관의 데이터 공유 프로세스에서 필수적인 부분이 되었습니다.

AI/ML 분석의 클라우드 마이그레이션

엔터프라이즈 고객은 전통적으로 AI/ML 모델을 온프레미스에 교육 및 배포했습니다. 규모의 경제 및 효율성 측면에서 이러한 고객은 AI/ML 기능을 퍼블릭, 하이브리드 또는 멀티 클라우드 클라우드 구현 환경으로 이전하기 위해 확장하고 있습니다. 그러나 다른 인프라에 노출될 수 있는 데이터에 바인딩됩니다. NetApp 솔루션은 에 필요한 모든 범위의 사이버 보안 위협을 해결합니다 "데이터 보호" 또한, 보안 평가 및 프로토피아 데이터 변환과 결합하면 이미지 처리 AI/ML 워크로드를 클라우드로 마이그레이션하는 데 따른 위험을 최소화할 수 있습니다.

다른 업계에서의 에지 컴퓨팅 및 AI 추론의 추가 사용 사례는 를 참조하십시오 "TR-4886 Ai Inferencing at the Edge(가장자리 기준 TR-4886 AI 추론" 그리고 NetApp AI 블로그, "인텔리전스와 개인 정보 보호 비교".

기술 개요

이 섹션에서는 이 솔루션을 완료하는 데 필요한 다양한 기술 구성 요소에 대해 간략하게 설명합니다.

프로토피아

Protopia AI는 오늘날 시장에서 기밀 추론을 위한 방해되지 않는 소프트웨어 전용 솔루션을 제공합니다. Protopia 솔루션은 중요한 정보의 노출을 최소화하여 추론 서비스를 위한 탁월한 보호 기능을 제공합니다. AI는 현재 작업을 수행하는 데 꼭 필요한 데이터 레코드의 정보만 제공합니다. 대부분의 추론 작업에서는 모든 데이터 레코드에 있는 모든 정보를 사용하지 않습니다. AI가 이미지, 음성, 비디오 또는 구조화된 표 형식 데이터를 소비하고 있는지 여부와 관계없이 Protopia는 추론 서비스에 필요한 것만 제공합니다. 특허를 획득한 코어 기술은 수학적으로 선별된 소음을 사용하여 데이터를 변환하고 지정된 ML 서비스에서 필요하지 않은 정보를 제공합니다. 이 솔루션은 데이터를 마스킹하지 않고 선별된 랜덤 노이즈를 사용하여 데이터 표현을 변경합니다.

프로토피아 솔루션은 모델의 기능과 관련하여 입력 기능 공간에 관련 정보를 계속 유지하는 경사 기반 퍼터버레이션 최대화 방법으로 표현을 변경하는 문제를 공식화합니다. 이 검색 프로세스는 ML 모델 교육을 마칠 때 미세 조정 통과로 실행됩니다. 통과가 자동으로 확률 분포 세트를 생성한 후 오버헤드가 낮은 데이터 변환은 이러한 분포의 노이즈 샘플을 데이터에 적용하고 추론을 위해 모델에 전달하기 전에 난독 처리 합니다.

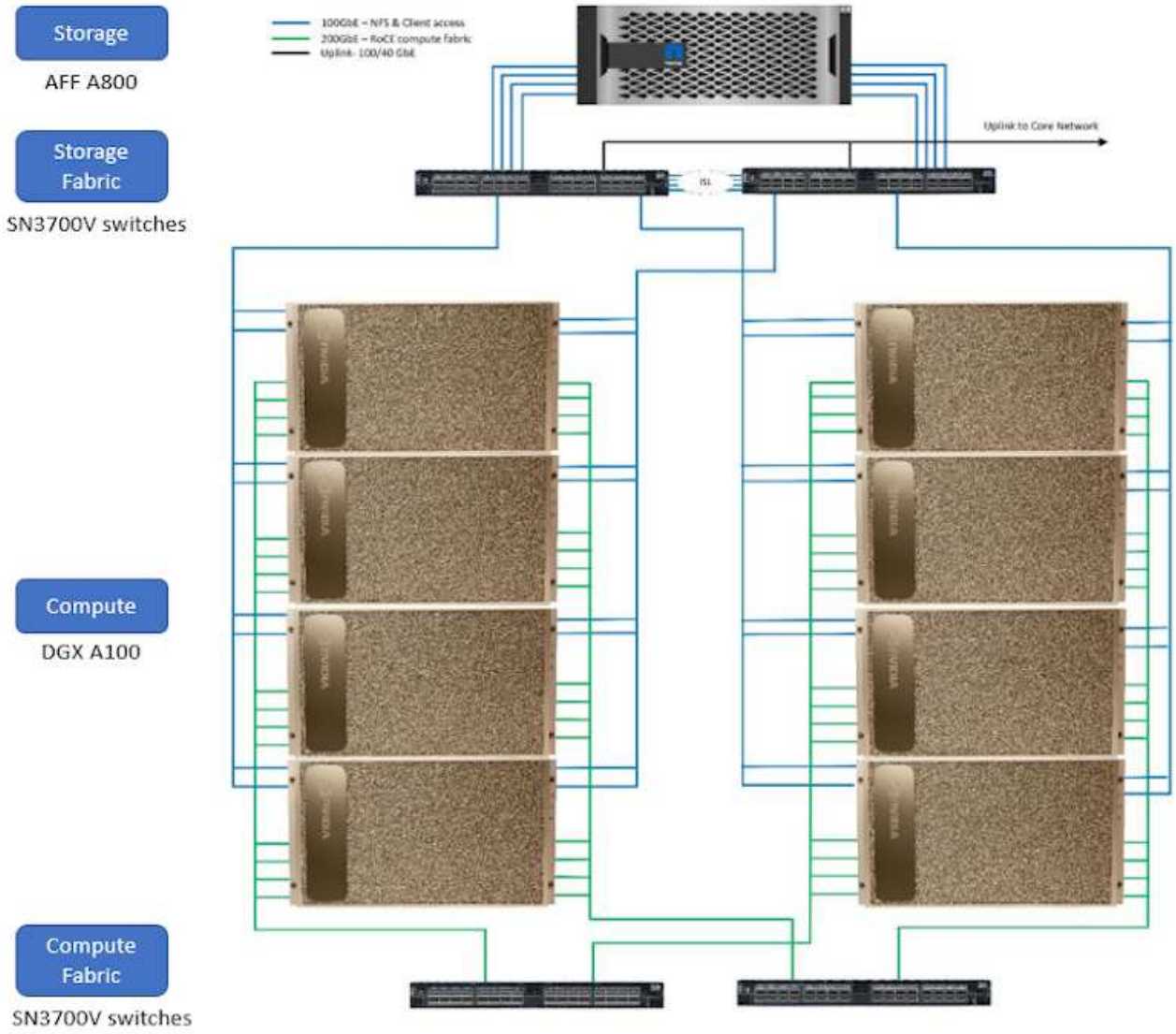
NetApp ONTAP AI를 참조하십시오

DGX A100 시스템과 NetApp 클라우드 연결형 스토리지 시스템에 기반한 NetApp ONTAP AI 참조 아키텍처는 NetApp과 NVIDIA가 개발 및 검증했습니다. 이 아키텍처는 IT 조직에 다음과 같은 이점을 제공하는 아키텍처를 제공합니다.

- 설계 복잡성 제거
- 컴퓨팅과 스토리지를 독립적으로 확장할 수 있습니다
- 고객이 작은 규모로 시작한 후 원활하게 확장할 수 있도록 지원
- 다양한 성능 및 비용 요소에 부합하는 폭넓은 스토리지 옵션을 제공합니다

ONTAP AI는 DGX A100 시스템과 NetApp AFF A800 스토리지 시스템을 최첨단 네트워킹과 긴밀하게 통합합니다. ONTAP AI는 설계 복잡성과 추측을 제거함으로써 AI 배포를 단순화합니다. 고객은 작은 규모로 시작한 후 에지에서 코어 및 클라우드까지 포괄하여 데이터를 지능적으로 관리하면서 중단 없이 확장할 수 있습니다.

다음 그림은 DGX A100 시스템과 ONTAP AI 솔루션 제품군의 다양한 변형을 보여줍니다. 최대 8개의 DGX A100 시스템에서 AFF A800 시스템 성능 검증 ONTAP 클러스터에 스토리지 컨트롤러 쌍을 추가하면 아키텍처를 여러 개의 랙으로 확장하여 선형 성능으로 많은 DGX A100 시스템과 페타바이트급 스토리지 용량을 지원할 수 있습니다. 이 접근 방식은 사용되는 DL 모델의 크기와 필요한 성능 메트릭을 기준으로 컴퓨팅 대 스토리지 비율을 독립적으로 변경할 수 있는 유연성을 제공합니다.



ONTAP AI에 대한 자세한 내용은 를 참조하십시오 "NVA-1153: NVIDIA DGX A100 시스템 및 Mellanox Spectrum 이더넷 스위치를 포함하는 NetApp ONTAP AI"

NetApp ONTAP를 참조하십시오

NetApp의 최신 세대 스토리지 관리 소프트웨어인 ONTAP 9.11을 통해 기업은 인프라를 현대화하고 클라우드 지원 데이터 센터로 전환할 수 있습니다. ONTAP는 업계 최고 수준의 데이터 관리 기능을 활용하여 데이터가 상주하는 위치와 상관없이 단일 툴셋으로 데이터를 관리하고 보호할 수 있습니다. 필요에 따라 에지, 코어, 클라우드 등 어느 위치로도 데이터를 자유롭게 이동할 수 있습니다. ONTAP 9.11에는 데이터 관리를 단순화하고, 중요 데이터를 가속화하고, 보호하며, 하이브리드 클라우드 아키텍처 전반에 걸쳐 차세대 인프라 기능을 지원하는 다양한 기능이 포함되어 있습니다.

NetApp DataOps 툴킷

NetApp DataOps Toolkit은 개발자, 데이터 과학자, DevOps 엔지니어 및 데이터 엔지니어가 새로운 데이터 볼륨의 거의 즉각적인 프로비저닝 또는 JupyterLab 작업 공간, 데이터 볼륨의 거의 즉각적인 클론 복제 또는 JupyterLab 작업 공간과 같은 다양한 데이터 관리 작업을 간단하게 수행할 수 있는 Python 라이브러리입니다. 추적 기능 또는 베이스라인 기능을 위해 데이터 볼륨의 스냅샷 또는 JupyterLab 작업 공간을 거의 즉각적으로 생성합니다. 이 Python 라이브러리는 명령행 유틸리티 또는 Python 프로그램이나 Jupyter 노트북으로 가져올 수 있는 기능의 라이브러리로 작동할 수

있습니다.

NVIDIA Triton Inference Server를 참조하십시오

NVIDIA Triton Inference Server는 모델 구축 및 실행을 표준화하여 운영 환경에서 신속하고 확장 가능한 AI를 제공하는 오픈 소스 추론 제공 소프트웨어입니다. Triton Inference Server는 팀이 GPU 또는 CPU 기반 인프라의 모든 프레임워크에서 훈련된 AI 모델을 구축, 실행 및 확장할 수 있도록 지원하여 AI 추론을 간소화합니다. Triton Inference Server는 TensorFlow, NVIDIA TensorRT, PyTorch, MXNet, OpenVINO 등 Triton은 모든 주요 퍼블릭 클라우드 AI 및 Kubernetes 플랫폼에서 사용할 수 있는 오케스트레이션 및 확장을 위해 Kubernetes와 통합됩니다. 또한 많은 MLOps 소프트웨어 솔루션과 통합됩니다.

PyTorch

"PyTorch" 는 오픈 소스 ML 프레임워크입니다. GPU 및 CPU를 사용하는 딥 러닝용으로 최적화된 텐서 라이브러리입니다. PyTorch 패키지에는 여러 유용한 유틸리티 간에 효율적인 서너 직렬화를 위한 여러 유틸리티를 제공하는 다차원 Tensor용 데이터 구조가 포함되어 있습니다. 또한 컴퓨팅 기능이 있는 NVIDIA GPU에서 텐서 컴퓨팅을 실행할 수 있는 CUDA 상대가 있습니다. 이 검증에서는 OpenCV-Python(CV2) 라이브러리를 사용하여 모델을 검증하고 Python의 가장 직관적인 컴퓨터 비전 개념을 활용합니다.

데이터 관리를 단순화하십시오

데이터 관리는 AI 애플리케이션에 적합한 리소스를 사용하고 AI/ML 데이터 세트를 교육할 수 있도록 엔터프라이즈 IT 운영 및 데이터 과학자에게 매우 중요합니다. NetApp 기술에 대한 다음 추가 정보는 이 검증의 범위에 포함되지 않지만, 배포에 따라 달라질 수 있습니다.

ONTAP 데이터 관리 소프트웨어에는 운영을 간소화 및 단순화하고 총 운영 비용을 절감하는 다음과 같은 기능이 있습니다.

- 인라인 데이터 컴팩션 및 확대된 중복제거: 데이터 컴팩션은 스토리지 블록 내부의 낭비되는 공간을 줄이고, 중복제거는 실제 용량을 상당히 늘려줍니다. 이는 로컬에 저장된 데이터와 클라우드로 계층화된 데이터에 적용됩니다.
- 최소, 최대 및 적응형 서비스 품질(AQoS): 세부적인 서비스 품질(QoS) 제어로 고도의 공유 환경에서 중요 애플리케이션의 성능 수준을 유지할 수 있습니다.
- NetApp FabricPool를 참조하십시오. AWS(Amazon Web Services), Azure, NetApp StorageGRID 스토리지 솔루션을 포함한 퍼블릭 클라우드 및 프라이빗 클라우드 스토리지에 콜드 데이터를 자동으로 계층화합니다. FabricPool에 대한 자세한 내용은 를 참조하십시오 "TR-4598: FabricPool 모범 사례".

데이터 가속화 및 보호

ONTAP는 탁월한 수준의 성능과 데이터 보호를 제공하며 다음과 같은 방법으로 이러한 기능을 확장합니다.

- 성능 및 짧은 지연 시간: ONTAP는 가장 짧은 지연 시간으로 가장 높은 처리량을 제공합니다.
- 데이터 보호: ONTAP는 모든 플랫폼에서 공통 관리를 지원하는 내장 데이터 보호 기능을 제공합니다.
- NVE(NetApp 볼륨 암호화). ONTAP는 온보드 및 외부 키 관리를 모두 지원하는 기본 볼륨 레벨 암호화를 제공합니다.
- 멀티테넌시 및 다단계 인증. ONTAP를 사용하면 인프라 리소스를 최고 수준의 보안으로 공유할 수 있습니다.

미래 지향형 인프라

ONTAP은 다음과 같은 기능을 통해 끊임없이 변화하는 까다로운 비즈니스 요구사항을 충족할 수 있도록 지원합니다.

- 원활한 확장 및 무중단 운영: ONTAP은 운영 중단 없이 기존 컨트롤러 및 스케일아웃 클러스터에 용량을 추가할 수 있도록 지원합니다. 고객은 고비용이 따르는 데이터 마이그레이션이나 운영 중단 없이 NVMe 및 32Gb FC와 같은 최신 기술로 업그레이드할 수 있습니다.
- 클라우드 연결: ONTAP은 클라우드에 가장 많이 연결된 스토리지 관리 소프트웨어로, 모든 퍼블릭 클라우드에서 ONTAP Select(소프트웨어 정의 스토리지) 및 NetApp Cloud Volumes Service(클라우드 네이티브 인스턴스)에 대한 옵션을 제공합니다.
- 새로운 애플리케이션과 통합: ONTAP은 기존 엔터프라이즈 앱을 지원하는 인프라와 동일한 인프라를 사용하여 자율주행 차량, 스마트 시티, Industry 4.0과 같은 차세대 플랫폼 및 애플리케이션을 위한 엔터프라이즈급 데이터 서비스를 제공합니다.

NetApp Astra Control

NetApp Astra 제품군은 온프레미스 및 퍼블릭 클라우드에서 Kubernetes 애플리케이션을 위한 스토리지 및 애플리케이션 인식 데이터 관리 서비스를 제공하며, NetApp 스토리지 및 데이터 관리 기술을 기반으로 합니다. Kubernetes 애플리케이션을 쉽게 백업하고, 데이터를 다른 클러스터로 마이그레이션하고, 작업 중인 애플리케이션 클론을 즉시 생성할 수 있습니다. 퍼블릭 클라우드에서 실행 중인 Kubernetes 애플리케이션을 관리해야 하는 경우에는 의 문서를 참조하십시오 "Astra 제어 서비스". Astra Control Service는 GKE(Google Kubernetes Engine) 및 AKS(Azure Kubernetes Service)에서 Kubernetes 클러스터의 애플리케이션 인식 데이터 관리를 제공하는 NetApp 관리 서비스입니다.

NetApp Astra Trident

아스트라 "트라이던트" NetApp은 Docker 및 Kubernetes용 오픈 소스 동적 스토리지 오케스트레이터로서 영구 스토리지의 생성, 관리 및 사용을 단순화합니다. Kubernetes 네이티브 애플리케이션인 Trident는 Kubernetes 클러스터 내에서 직접 실행됩니다. Trident를 사용하면 고객이 DL 컨테이너 이미지를 NetApp 스토리지에 원활하게 배포하고 AI 컨테이너 배포를 위한 엔터프라이즈급 경험을 제공할 수 있습니다. Kubernetes 사용자(ML 개발자, 데이터 과학자 등)는 오케스트레이션 및 클론 복제를 생성, 관리 및 자동화하여 NetApp 기술이 제공하는 고급 데이터 관리 기능을 활용할 수 있습니다.

NetApp BlueXP 복사 및 동기화

"BlueXP 복사 및 동기화" 는 빠르고 안전한 데이터 동기화를 제공하는 NetApp 서비스입니다. 온프레미스 NFS 또는 SMB 파일 공유 간에 파일을 전송해야 하는 경우, NetApp StorageGRID, NetApp ONTAP S3, NetApp Cloud Volumes Service, Azure NetApp Files, Amazon Simple Storage Service(Amazon S3), Amazon Elastic File System(Amazon EFS), Azure Blob, Google Cloud Storage, 또는 IBM Cloud Object Storage인 BlueXP Copy and Sync는 필요한 파일을 빠르고 안전하게 이동합니다. 데이터가 전송되면 소스와 타겟 모두에서 사용할 수 있습니다. BlueXP Copy 및 Syncc 는 미리 정의된 일정에 따라 데이터를 지속적으로 동기화하므로 변경된 부분만 이동하므로 데이터 복제에 소비되는 시간과 비용이 최소화됩니다. BlueXP Copy and Sync는 설정 및 사용이 매우 간편한 서비스형 소프트웨어(SaaS) 툴입니다. BlueXP Copy 및 Sync에 의해 트리거되는 데이터 전송은 데이터 브로커에 의해 수행됩니다. AWS, Azure, Google Cloud Platform 또는 사내에 BlueXP Copy 및 Sync 데이터 브로커를 배포할 수 있습니다.

NetApp BlueXP 분류

강력한 AI 알고리즘을 기반으로 "NetApp BlueXP 분류" 전체 데이터 자산에 걸쳐 자동화된 제어 및 데이터 거버넌스를 제공합니다. 비용 절감 효과를 쉽게 파악하고 규정 준수 및 개인 정보 보호에 대한 우려 사항을 파악하며 최적화 기회를 찾을 수 있습니다. BlueXP Classification 대시보드를 통해 중복 데이터를 식별하여 중복을 제거하고 개인, 비개인 및

중요 데이터를 매핑하고 기밀 데이터 및 이상 상황에 대한 알림을 설정할 수 있습니다.

테스트 및 검증 계획

이 솔루션 설계의 경우 다음 세 가지 시나리오를 검증했습니다.

- Kubernetes용 NetApp DataOps Toolkit을 사용하여 조율된 JupyterLab 작업 공간 내의 프로토피아 난독 처리 기능을 사용하거나 사용하지 않는 추론 작업.
- Kubernetes용 NetApp DataOps Toolkit을 사용하여 오케스트레이션된 데이터 볼륨을 사용하는 Kubernetes에서 프로토피아 난독 처리를 사용하거나 사용하지 않는 배치 추론 작업
- Kubernetes용 NetApp DataOps 툴킷을 사용하여 조정된 NVIDIA Triton Inference Server 인스턴스를 사용한 추론 작업 네트워크를 통해 전송되는 모든 데이터가 난독 처리되어야 한다는 일반적인 요구 사항을 시뮬레이션하기 위해 Triton 추론 API를 호출하기 전에 이미지에 Protopia 난독 처리를 적용했습니다. 이 워크플로는 신뢰할 수 있는 영역 내에서 데이터를 수집하지만 추론을 위해 신뢰할 수 있는 영역 외부로 전달해야 하는 사용 사례에 적용됩니다. Protopia 난독 처리를 사용하지 않으면 중요한 데이터가 신뢰할 수 있는 영역을 벗어나지 않으면 이러한 유형의 워크플로를 구현할 수 없습니다.

구성을 테스트합니다

다음 표에서는 솔루션 설계 검증 환경을 간략하게 보여 줍니다.

구성 요소	버전
쿠버네티스	1.21.6
NetApp Astra Trident CSI 드라이버	22.01.0
Kubernetes용 NetApp DataOps 툴킷	2.3.0
NVIDIA Triton Inference Server를 참조하십시오	21.11-3장

테스트 절차

이 섹션에서는 검증을 완료하는 데 필요한 작업에 대해 설명합니다.

필수 구성 요소

이 섹션에 설명된 작업을 실행하려면 다음 도구가 설치 및 구성되어 있는 Linux 또는 macOS 호스트에 대한 액세스 권한이 있어야 합니다.

- Kubectl(기존 Kubernetes 클러스터에 액세스하도록 구성)
  - 설치 및 구성 지침을 찾을 수 있습니다 "여기".
- Kubernetes용 NetApp DataOps 툴킷
  - 설치 지침을 찾을 수 있습니다 "여기".

시나리오 1 – JupyterLab의 온디맨드 추론

1. AI/ML 추론 워크로드를 위한 Kubernetes 네임스페이스를 생성합니다.

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

2. NetApp DataOps 툴킷을 사용하여 추론을 수행할 데이터를 저장할 영구 볼륨을 프로비저닝합니다.

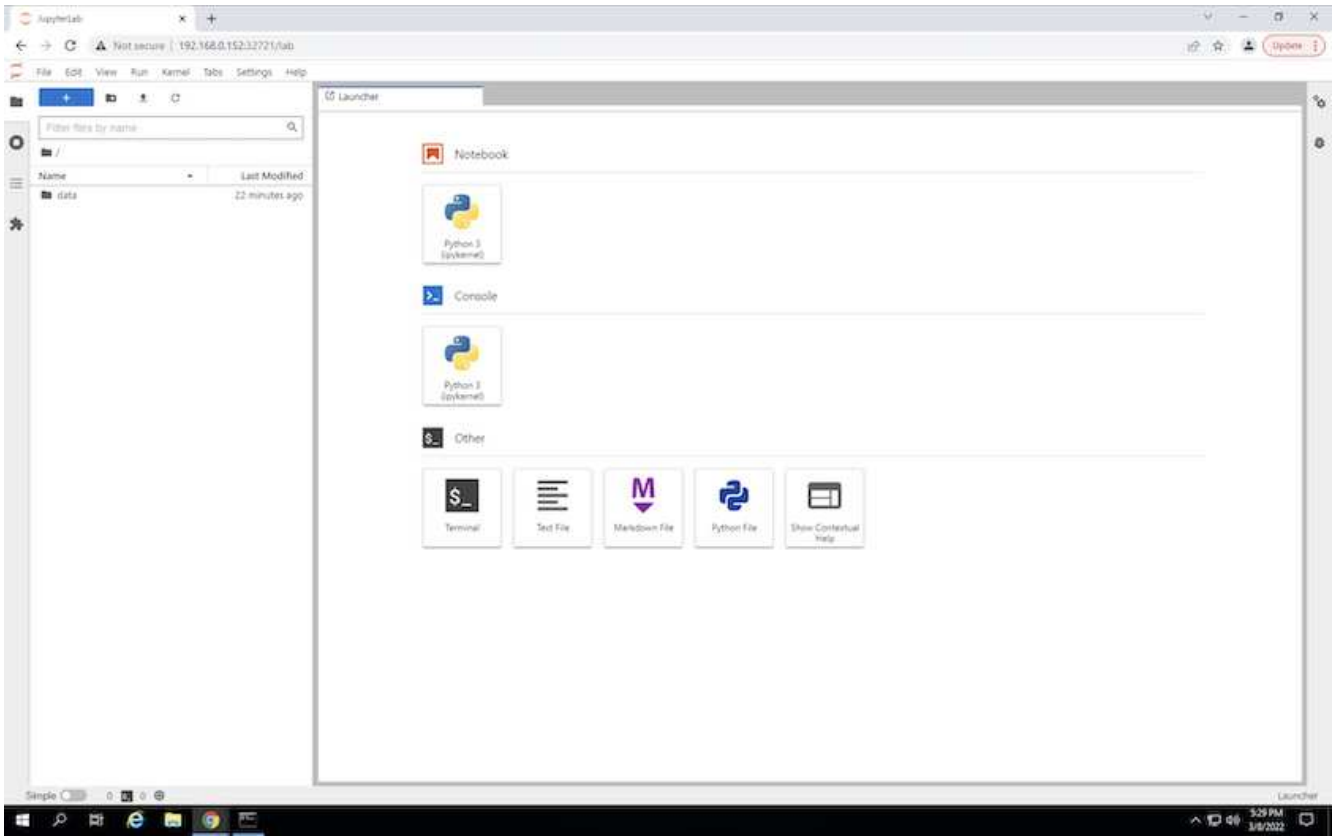
```
$ netapp_dataops_k8s_cli.py create volume --namespace=inference --pvc
-name=inference-data --size=50Gi
Creating PersistentVolumeClaim (PVC) 'inference-data' in namespace
'inference'.
PersistentVolumeClaim (PVC) 'inference-data' created. Waiting for
Kubernetes to bind volume to PVC.
Volume successfully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

3. NetApp DataOps Toolkit을 사용하여 새로운 JupyterLab 작업 공간을 생성합니다. '--mount-PVC' 옵션을 사용하여 이전 단계에서 생성한 영구 볼륨을 마운트합니다. 필요한 경우 '--nVidia-GPU' 옵션을 사용하여 NVIDIA GPU를 작업 공간에 할당합니다.

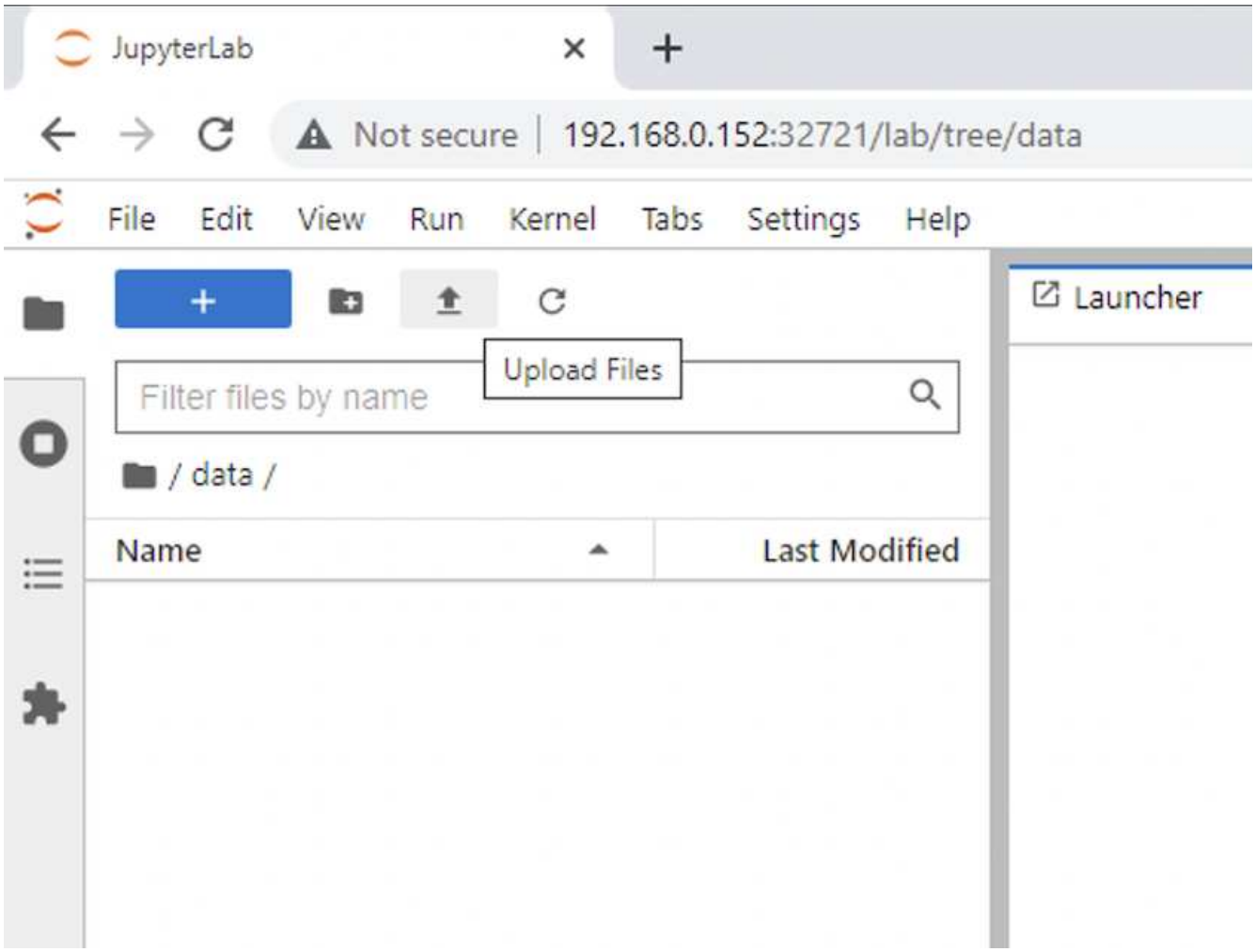
   다음 예에서는 영구 볼륨 '추론 데이터'가 '/home/jovyan/data'의 JupyterLab 작업 영역 컨테이너에 마운트됩니다. Jupytter의 공식 컨테이너 이미지를 사용할 때 JupyterLab 웹 인터페이스 내의 최상위 디렉토리로 /home/jovyan이 표시됩니다.

```
$ 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
```

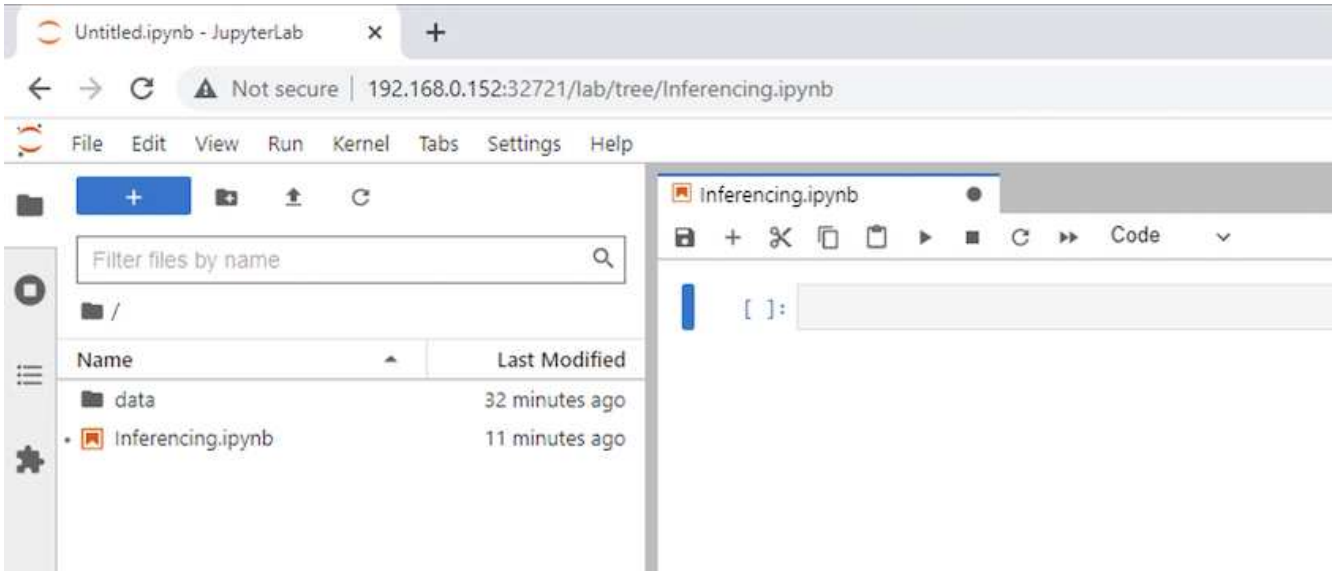
4. 'jupyterlab 생성' 명령의 출력에 지정된 URL을 사용하여 JupyterLab 작업 영역에 액세스합니다. 데이터 디렉토리는 작업 공간에 마운트된 영구 볼륨을 나타냅니다.



5. "ata" 디렉토리를 열고 추론을 수행할 파일을 업로드합니다. 파일이 데이터 디렉토리에 업로드되면 작업 공간에 마운트된 영구 볼륨에 자동으로 저장됩니다. 파일을 업로드하려면 다음 이미지와 같이 파일 업로드 아이콘을 클릭합니다.



6. 최상위 디렉토리로 돌아가서 새 전자 필기장을 만듭니다.



7. 노트북에 추론 코드를 추가합니다. 다음 예에서는 이미지 감지 사용 사례에 대한 추론 코드를 보여 줍니다.

Launcher | image-demo-pytorch.ipynb

Code | Python 3 (ipykernel)

STEP 3-1: Clean (Without obfuscation) detection

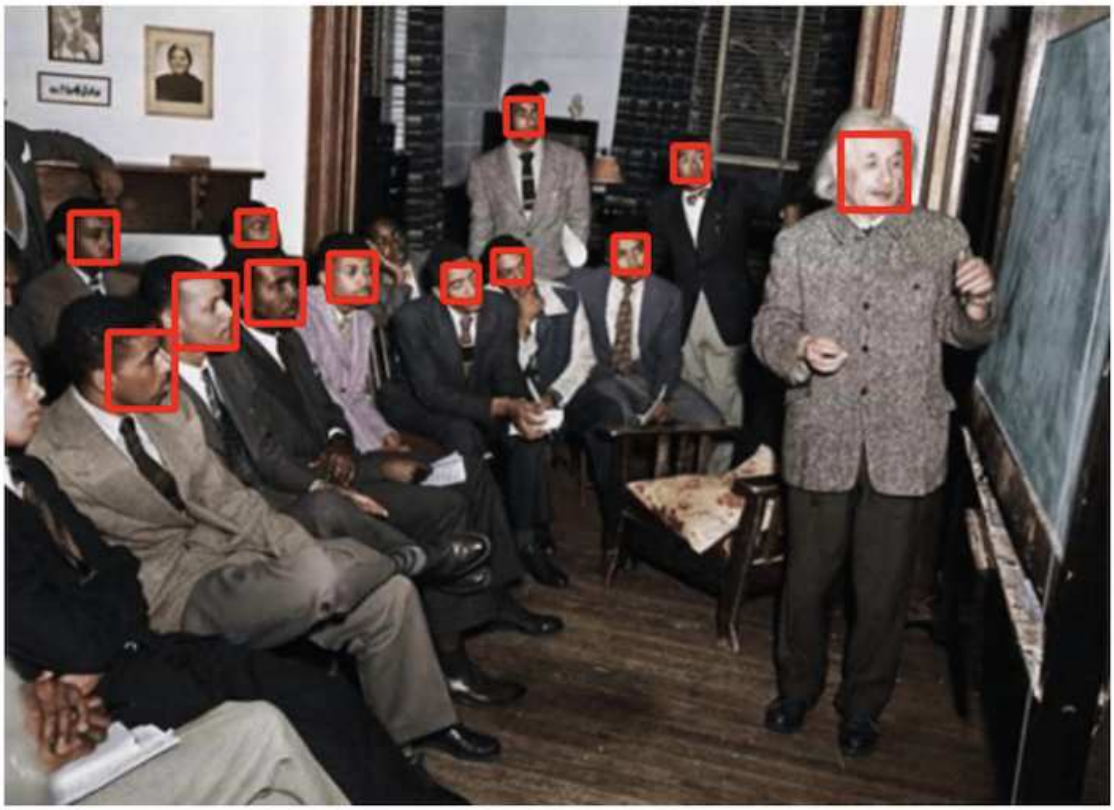
```
[9]: # get current frame
frame = input_image

# preprocess input
preprocessed_input = preprocess_input(frame)
preprocessed_input = torch.Tensor(preprocessed_input).to(device)

# run forward pass
clean_activation = clean_model.forward_head(preprocessed_input)  # runs the first few layers
loc, pred = clean_model.forward_tail(clean_activation)  # runs rest of the layers

# postprocess output
clean_pred = (loc.detach().cpu().numpy(), pred.detach().cpu().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)
```

Launcher | image-demo-pytorch.ipynb

Code | Python 3 (ipykernel)

Visualize Clean (Without obfuscation) detection

```
[10]: show_cv2_image(clean_frame, scale=2)
```



8. 추론 코드에 Protopia 난독 처리를 추가합니다. Protopia는 고객과 직접 협력하여 사용 사례별 문서를 제공하며 이 기술 보고서의 범위를 벗어납니다. 다음 예제에서는 Protopia 난독 처리를 추가한 이미지 검색 사용 사례에 대한 추론 코드를 보여 줍니다.

Launcher | image-demo-pytorch.ipynb

Code | Python 3 (ipykernel)

STEP 3-2: Protopia AI (With obfuscation) detection

[11]:
```
# 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.forward_head(preprocessed_input)  # runs the first few layers
##################################################################
#          SINGLE ADITIONAL LINE FOR PRIVATE INFERENCE           #
##################################################################
noisy_activation = noisy_model.forward_noise(not_noisy_activation)
##################################################################
loc, pred = noisy_model.forward_tail(noisy_activation)  # runs rest of the layers

# postprocess output
noisy_pred = (loc.detach().cpu().numpy(), pred.detach().cpu().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)
```

Launcher | image-demo-pytorch.ipynb

Code | Python 3 (ipykernel)

Visualize Protopia AI (With obfuscation) detection

[12]:
```
show_cv2_image(noisy_reconstruction, scale=2)
```

시나리오 2 – Kubernetes의 배치 추론

1. AI/ML 추론 워크로드를 위한 Kubernetes 네임스페이스를 생성합니다.

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

2. NetApp DataOps 툴킷을 사용하여 추론을 수행할 데이터를 저장할 영구 볼륨을 프로비저닝합니다.

```
$ 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. 추론을 수행할 데이터로 새 영구 볼륨을 채웁니다.

   PVC로 데이터를 로드하는 방법은 여러 가지가 있습니다. 현재 데이터가 NetApp StorageGRID 또는 Amazon S3와 같은 S3 호환 오브젝트 스토리지 플랫폼에 저장되어 있는 경우 를 사용할 수 있습니다 "NetApp DataOps 툴킷 S3 Data Mover 기능". 또 하나의 간단한 방법은 JupyterLab 작업 공간을 만든 다음, 섹션 “의 3-5단계에 설명된 대로 JupyterLab 웹 인터페이스를 통해 파일을 업로드하는 것입니다시나리오 1 – JupyterLab의 온디맨드 추론.”

4. 배치 추론 작업을 위해 Kubernetes 작업을 생성합니다. 다음 예는 이미지 감지 사용 사례에 대한 배치 추론 작업을 보여줍니다. 이 작업은 이미지 세트의 각 이미지에서 추론을 수행하고 추론 정확도 메트릭을 stdout에 씁니다.

```
$ 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. 추론 작업이 성공적으로 완료되었는지 확인합니다.

```
$ 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. 추론 작업에 Protopia 난독 처리를 추가합니다. 이 기술 보고서의 범위를 벗어나는 Protopia에서 직접 Protopia 난독 처리를 추가하기 위한 사용 사례별 지침을 찾을 수 있습니다. 다음 예제는 알파 값 0.8을 사용하여 Protopia 난독 처리가 추가된 얼굴 인식 사용 사례에 대한 일괄 추론 작업을 보여 줍니다. 이 작업은 이미지 세트의 각 이미지에 대한 추론을 수행하기 전에 Protopia 난독 처리를 적용한 다음 추론 정확도 메트릭을 stdout에 기록합니다.

   알파 값 0.05, 0.1, 0.2, 0.4, 0.6, 0.8, 0.9 및 0.95. 에서 결과를 볼 수 있습니다 ""추론 정확도 비교.""

```
$ 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. 추론 작업이 성공적으로 완료되었는지 확인합니다.

```
$ 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
```

시나리오 3 – NVIDIA Triton Inference Server

1. AI/ML 추론 워크로드를 위한 Kubernetes 네임스페이스를 생성합니다.

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

2. NetApp DataOps 툴킷을 사용하여 NVIDIA Triton Inference Server의 모델 저장소로 사용할 영구 볼륨을 프로비저닝합니다.

```
$ 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. 의 새 영구 볼륨에 모델을 저장합니다 "형식" 이 기능은 NVIDIA Triton Inference Server에서 인식됩니다.

   PVC로 데이터를 로드하는 방법은 여러 가지가 있습니다. 간단한 방법은 “의 3-5단계에 설명된 대로 JupyterLab 작업 공간을 만든 다음 JupyterLab 웹 인터페이스를 통해 파일을 업로드하는 것입니다시나리오 1 – JupyterLab의 온디맨드 추론. ”

4. NetApp DataOps 툴킷을 사용하여 새 NVIDIA Triton Inference Server 인스턴스를 구축합니다.

```
$ 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. Triton 클라이언트 SDK를 사용하여 추론 작업을 수행합니다. 인용된 다음 Python 코드는 Triton Python 클라이언트 SDK를 사용하여 얼굴 감지 사용 사례에 대한 추론 작업을 수행합니다. 이 예에서는 Triton API를 호출하고 추론을 위해 이미지를 전달합니다. 그런 다음 Triton Inference Server가 요청을 수신하고 모델을 호출하고 추론 출력을 API 결과의 일부로 반환합니다.

```
# 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. 추론 코드에 Protopia 난독 처리를 추가합니다. Protopia에서 직접 Protopia 난독 처리를 추가하기 위한 사용 사례별 지침을 찾을 수 있지만 이 프로세스는 이 기술 보고서의 범위를 벗어납니다. 다음 예제에서는 앞의 5단계에서 표시되지만 Protopia 난독 처리를 추가한 것과 동일한 Python 코드를 보여 줍니다.

   이 경우, Triton API로 전달되기 전에 Protopia 난독 처리 기능이 이미지에 적용됩니다. 따라서, 난독 처리된

이미지가 로컬 시스템에서 절대 빠져나가지는 않습니다. 난독 처리된 이미지만 네트워크를 통해 전달됩니다. 이 워크플로는 신뢰할 수 있는 영역 내에서 데이터를 수집한 다음 추론을 위해 신뢰할 수 있는 영역 외부로 전달해야 하는 사용 사례에 적용됩니다. Protopia 난독 처리를 사용하지 않으면 중요한 데이터가 신뢰할 수 있는 영역을 벗어나지 않으면 이러한 유형의 워크플로를 구현할 수 없습니다.

```
# 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)
```

추론 정확도 비교

이 검증에서는 일련의 원시 이미지를 사용하여 이미지 검색 사용 사례에 대한 추론을 수행했습니다. 그런 다음 추론 전에 Protopia 난독 처리를 추가하여 동일한 이미지 집합에서 동일한 추론 작업을 수행했습니다. 우리는 프로토피아 난독 처리 구성 요소에 대해 서로 다른 알파 값을 사용하여 작업을 반복했습니다. Protopia 난독 처리 컨텍스트에서 알파 값은 더 높은 난독 처리 수준을 나타내는 더 높은 알파 값으로 적용되는 난독 처리 양을 나타냅니다. 그런 다음 이 서로 다른 실행 간에 추론 정확도를 비교합니다.

다음 두 표에서는 사용 사례에 대한 자세한 내용과 결과에 대해 설명합니다.

Protopia는 고객과 직접 협력하여 특정 사용 사례에 적합한 알파 값을 결정합니다.

구성 요소	세부 정보
모델	FaceBoXes(PyTorch) -
데이터 세트	FDDB 데이터 세트

프로토피아 난독화	알파	정확성
아니요	해당 없음	0.9337148153739079
예	0.05	0.902876627325002
예	0.1	0.9024301009661478
예	0.2	0.9081836283186224
예	0.4	0.9073066107482036
예	0.6	0.8847816568680239
예	0.8	0.8841195749171925
예	0.9	0.8455427675252052
예	0.95	0.8455427675252052

난독화 속도

이 검증을 위해 Protopia 난독 처리를 1920 x 1080 픽셀 이미지에 5회 적용하고 난독 처리 단계가 매번 완료되는 데 걸리는 시간을 측정했습니다.

난독 처리를 적용하기 위해 단일 NVIDIA V100 GPU에서 실행되는 PyTorch를 사용했고 실행 간에 GPU 캐시를 지웠습니다. 난독화 단계는 5회 실행에서 각각 5.47ms, 5.27ms, 4.54ms, 5.24ms, 4.84ms를 완료하는 데 각각 걸렸습니다. 평균 속도는 5.072ms였습니다.

결론

데이터는 세 가지 상태, 즉 유휴 상태, 전송 중 상태 및 계산 중에 있습니다. AI 추론 서비스의 중요한 부분은 전체 프로세스 동안 위협으로부터 데이터를 보호하는 것이 되어야 합니다. 추론 도중에 데이터를 보호하는 것은 매우 중요합니다. 이 프로세스에서는 외부 고객과 추론 서비스를 제공하는 회사 모두에 대한 비공개 정보를 표시할 수 있기 때문입니다. Protopia AI는 오늘날의 시장에서 기밀 AI 추론을 위한 비간섭 소프트웨어 전용 솔루션입니다. Protopia를 통해 AI는 현재 AI/ML 작업을 수행하는 데 필수적인 데이터 레코드에 변환된 정보만 제공합니다. 이 확률적 변환은 마스킹의 형태가 아니며 큐레이션 노이즈를 사용하여 데이터의 표현을 수학적으로 변경하는 것을 기반으로 합니다.

ONTAP 기능을 갖춘 NetApp 스토리지 시스템은 로컬 SSD 스토리지와 동일하거나 더 우수한 성능을 제공하며 NetApp DataOps Toolkit과 함께 데이터 과학자, 데이터 엔지니어, AI/ML 개발자 및 비즈니스 또는 엔터프라이즈 IT 의사 결정자에게 다음과 같은 이점을 제공합니다.

- AI 시스템, 분석 및 기타 중요한 비즈니스 시스템 간에 데이터를 손쉽게 공유 이러한 데이터 공유는 인프라

오버헤드를 줄이고 성능을 향상하며 기업 전체에서 데이터 관리를 간소화합니다.

- 컴퓨팅과 스토리지를 독립적으로 확장하므로 비용을 최소화하고 리소스 사용량을 높일 수 있습니다.
- 즉각적이고 공간 효율적인 사용자 작업 공간, 통합 버전 제어 및 자동화된 구축을 위해 통합 Snapshot 복사본과 클론을 사용하여 개발 및 구축 워크플로우를 간소화했습니다.
- 재해 복구, 비즈니스 연속성 및 규정 요구사항을 충족하는 엔터프라이즈급 데이터 보호 및 데이터 거버넌스
- Jupyter 노트북에 있는 NetApp DataOps Toolkit에서 데이터 과학자 작업 공간의 Snapshot 복사본을 신속하게 만들어 백업 및 추적 기능을 제공합니다.

NetApp 및 Protopia 솔루션은 엔터프라이즈급 AI 추론 구축에 이상적인 유연한 스케일아웃 아키텍처를 제공합니다. 이 솔루션은 사내 및 하이브리드 클라우드 구축 모두에서 책임 있는 AI 사례를 충족할 수 있는 기밀 AI 추론 요구사항을 충족하는 중요한 정보에 대해 데이터 보호를 지원합니다.

추가 정보 및 승인 정보를 찾을 수 있는 위치

이 문서에 설명된 정보에 대한 자세한 내용은 다음 문서 및/또는 웹 사이트를 참조하십시오.

- NetApp ONTAP 데이터 관리 소프트웨어 - ONTAP 정보 라이브러리

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

- 컨테이너용 NetApp 영구 스토리지 - NetApp Trident

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

- NetApp DataOps 툴킷

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

- 컨테이너용 NetApp 영구 스토리지 - NetApp Astra Trident

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

- 토폴로지 AI - 기밀 추론

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

- NetApp BlueXP 복사 및 동기화

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

- NVIDIA Triton Inference Server를 참조하십시오

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

- NVIDIA Triton Inference Server 설명서

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

- PyTorch의 FaceBoxes

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

감사의 말

- Mark Cates, 수석 제품 관리자, NetApp
- NetApp 기술 마케팅 엔지니어 Sufian Ahmad
- Hadi Esmaeilzadeh, 최고 기술 책임자 및 Protopia AI 교수

저작권 정보

Copyright © 2024 NetApp, Inc. All Rights Reserved. 미국에서 인쇄된 본 문서의 어떠한 부분도 저작권 소유자의 사전 서면 승인 없이는 어떠한 형식이나 수단(복사, 녹음, 녹화 또는 전자 검색 시스템에 저장하는 것을 비롯한 그래픽, 전자적 또는 기계적 방법)으로도 복제될 수 없습니다.

NetApp이 저작권을 가진 자료에 있는 소프트웨어에는 아래의 라이센스와 고지사항이 적용됩니다.

본 소프트웨어는 NetApp에 의해 '있는 그대로' 제공되며 상품성 및 특정 목적에의 적합성에 대한 명시적 또는 묵시적 보증을 포함하여(이에 제한되지 않음) 어떠한 보증도 하지 않습니다. NetApp은 대체품 또는 대체 서비스의 조달, 사용 불능, 데이터 손실, 이익 손실, 영업 중단을 포함하여(이에 국한되지 않음), 이 소프트웨어의 사용으로 인해 발생하는 모든 직접 및 간접 손해, 우발적 손해, 특별 손해, 징벌적 손해, 결과적 손해의 발생에 대하여 그 발생 이유, 책임론, 계약 여부, 엄격한 책임, 불법 행위(과실 또는 그렇지 않은 경우)와 관계없이 어떠한 책임도 지지 않으며, 이와 같은 손실의 발생 가능성이 통지되었다 하더라도 마찬가지입니다.

NetApp은 본 문서에 설명된 제품을 언제든지 예고 없이 변경할 권리를 보유합니다. NetApp은 NetApp의 명시적인 서면 동의를 받은 경우를 제외하고 본 문서에 설명된 제품을 사용하여 발생하는 어떠한 문제에도 책임을 지지 않습니다. 본 제품의 사용 또는 구매의 경우 NetApp에서는 어떠한 특허권, 상표권 또는 기타 지적 재산권이 적용되는 라이센스도 제공하지 않습니다.

본 설명서에 설명된 제품은 하나 이상의 미국 특허, 해외 특허 또는 출원 중인 특허로 보호됩니다.

제한적 권리 표시: 정부에 의한 사용, 복제 또는 공개에는 DFARS 252.227-7013(2014년 2월) 및 FAR 52.227-19(2007년 12월)의 기술 데이터-비상업적 품목에 대한 권리(Rights in Technical Data -Noncommercial Items) 조항의 하위 조항 (b)(3)에 설명된 제한사항이 적용됩니다.

여기에 포함된 데이터는 상업용 제품 및/또는 상업용 서비스(FAR 2.101에 정의)에 해당하며 NetApp, Inc.의 독점 자산입니다. 본 계약에 따라 제공되는 모든 NetApp 기술 데이터 및 컴퓨터 소프트웨어는 본질적으로 상업용이며 개인 비용만으로 개발되었습니다. 미국 정부는 데이터가 제공된 미국 계약과 관련하여 해당 계약을 지원하는 데에만 데이터에 대한 전 세계적으로 비독점적이고 양도할 수 없으며 재사용이 불가능하며 취소 불가능한 라이센스를 제한적으로 가집니다. 여기에 제공된 경우를 제외하고 NetApp, Inc.의 사전 서면 승인 없이는 이 데이터를 사용, 공개, 재생산, 수정, 수행 또는 표시할 수 없습니다. 미국 국방부에 대한 정부 라이센스는 DFARS 조항 252.227-7015(b)(2014년 2월)에 명시된 권한으로 제한됩니다.

상표 정보

NETAPP, NETAPP 로고 및 http://www.netapp.com/TM에 나열된 마크는 NetApp, Inc.의 상표입니다. 기타 회사 및 제품 이름은 해당 소유자의 상표일 수 있습니다.