人工知能
NetApp Solutions

NetApp
April 10, 2024

目次

ネットアップの人工知能ソリューション 1
AI 統合インフラ 1
データパイプライン、データレイク、管理 16
ユースケース 162

ネットアップの人工知能ソリューション

AI 統合インフラ

AIおよびMLモデルトレーニング用のLenovo ThinkSystem SR670 V2搭載NetApp AFF A400

TR-4810：『NetApp AFF A400 with Lenovo ThinkSystem SR670 V2 for AI and ML Model Training』

Sathish Thyagarajan、David Arnette、NetApp Mircea Troaca、Lenovo

この解決策 は、ネットアップストレージと、人工知能（AI）ワークロード向けに最適化されたLenovoサーバを使用したミッドレンジのクラスタアーキテクチャを提供します。これは、ほとんどのコンピューティングジョブがシングルノード（シングルまたはマルチGPU）であるか、少数のコンピューティングノードに分散される、中小企業を対象としています。この解決策 は、多くの企業で日 々 行われているAIトレーニングの業務に対応しています。

本ドキュメントでは、8台のGPU Lenovo SR670V2サーバ、ミッドレンジのNetApp AFF A400ストレージシステム、100GbEインターコネクトスイッチで構成されるコンピューティングとストレージの構成のテストと検証について説明します。パフォーマンスを測定するために、ResNet50をImageNetデータセットで使用し、バッチサイズを408、ハーフ精度、CUDA、cuDNNにしました。このアーキテクチャは、NetApp ONTAP クラウド対応データストレージのエンタープライズクラスの機能を必要とするAIイニシアチブから始めた中堅企業のお客様に、効率性と対費用効果に優れた解決策 を提供します。

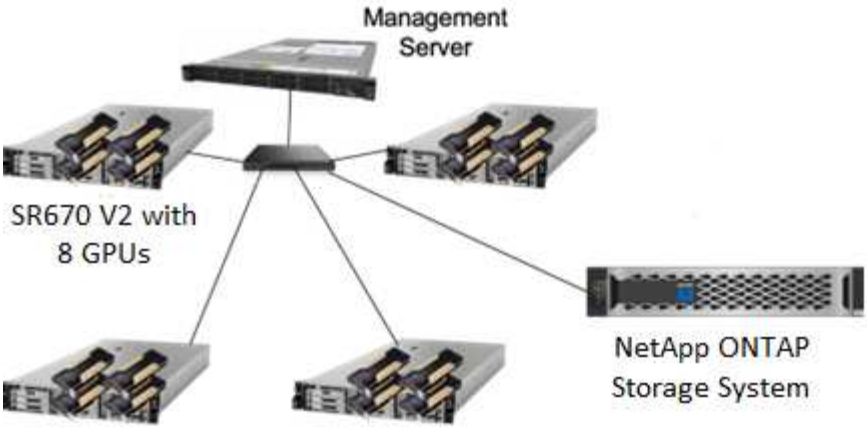
対象読者

本ドキュメントは、次のような方を対象としています。

- データサイエンティスト、データエンジニア、データ管理者、AIシステムの開発者
- AIモデルの開発のためのソリューションを設計するエンタープライズアーキテクト
- ディープラーニング（DL）と機械学習（ML）の開発目標を達成するための効率的な方法を探しているデータサイエンティストとデータエンジニア
- AI導入の市場投入までの時間を最短に短縮したいと考えているビジネスリーダーや、OT / ITの意思決定者

解決策アーキテクチャ

Lenovo ThinkSystemサーバを搭載したこの解決策 と、AFF ストレージを搭載したNetApp ONTAP は、GPUの処理能力と従来のCPUを組み合わせて、大規模なデータセットでAIトレーニングを処理するように設計されています。この検証では、1台のNetApp AFF A400ストレージシステムに加え、Lenovo SR670 V2サーバを1台、2台、または4台使用するスケールアウトアーキテクチャにより、パフォーマンスの向上と最適なデータ管理を実現します。次の図に、アーキテクチャの概要を示します。



このネットアップと Lenovo 解決策は、主に次のようなメリットをもたらします。

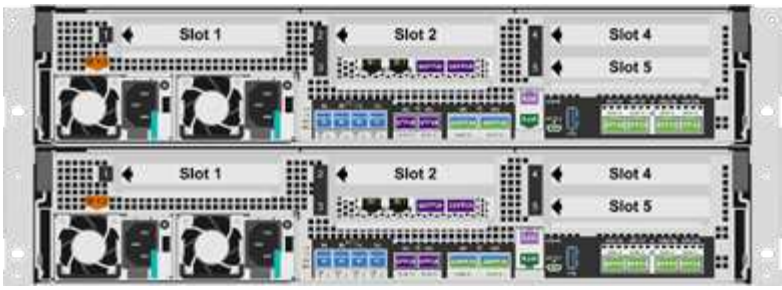
- 複数のトレーニングジョブを並行して実行する場合、効率性とコスト効率に優れたパフォーマンスを実現します
- Lenovoのサーバの数や、ネットアップストレージコントローラのさまざまなモデルに基づく拡張性に優れたパフォーマンス
- 堅牢なデータ保護により、データ損失ゼロで目標復旧時点（RPO）と目標復旧時間（RTO）を達成
- Snapshotとクローンを使用してデータ管理を最適化し、開発ワークフローを合理化

テクノロジの概要

このセクションでは、この解決策 の主なコンポーネントについて詳しく説明します。

NetApp AFF システム

NetApp AFF ストレージシステムは、業界をリードするパフォーマンス、卓越した柔軟性、クラウド統合、業界最高のデータ管理機能を備えており、エンタープライズストレージの要件を満たすことができます。フラッシュに特化して設計されたAFF システムは、ビジネスクリティカルなデータの高速化、管理、保護に役立ちます。



NetApp AFF A400は、次の機能を搭載したミッドレンジのNVMeフラッシュストレージシステムです。

- 最大実効容量：約20PB
- 最大スケールアウト：2~24ノード（HAペア×12）
- 25GbEおよび16Gb FCホストがサポートされます
- NVMe拡張ストレージシェルフへの100GbE RDMA over Converged Ethernet（RoCE）接続
- NVMeシェルフが接続されていない場合は、100GbE RoCEポートをホストネットワークの接続に使用できます
- 拡張ストレージシェルフに12Gbps SAS接続を提供します
- 次の2つの構成があります。
  - イーサネット：25Gbイーサネット（SFP28）ポート×4
  - ファイバチャネル：16Gb FC（SFP+）ポート×4
- 100% 8KBのランダムリード、レイテンシは0.4ミリ秒、400k IOPS

エントリレベルのAI / ML環境向けのNetApp AFF A250には、次のような機能があります。

- 最大実効容量：35PB
- 最大スケールアウト：2~24ノード（HAペア×12）
- 440 万 IOPS のランダムリード： 1 ミリ秒
- 最新のNetApp ONTAP リリースONTAP 9.8以降を基盤としています
- HAおよびクラスタインターコネクト用に25GBのイーサネットポートを2つ

ネットアップは、AFF A800やAFF A700など、大規模なAI / ML環境に優れたパフォーマンスと拡張性を提供するストレージシステムも提供しています。

NetApp ONTAP

ネットアップが提供する最新世代のストレージ管理ソフトウェアONTAP 9を使用すれば、インフラを最新化し、クラウド対応のデータセンターに移行できます。ONTAP は、業界をリードするデータ管理機能を活用して、データの格納場所に関係なく、単一のツールセットでデータの管理と保護を実現します。データは、エッジ、コア、クラウドなど、必要な場所に自由に移動することもできます。ONTAP 9には、データ管理を簡易化し、重要なデータの高速化と保護を実現するさまざまな機能が搭載されており、ハイブリッドクラウドアーキテクチャ全体で将来のニーズに対応できるインフラを実現できます。

データ管理を簡易化

データ管理は、アプリケーションやデータセットに適切なリソースを使用できるようにするために、エンター

プライズ IT 運用にとって非常に重要です。ONTAP には、運用を合理化および簡易化し、総運用コストを削減するための次の機能が含まれています。

- * インラインデータコンパクションと重複排除の強化。 * データコンパクションはストレージブロック内の無駄なスペースを削減し、重複排除は実効容量を大幅に増やします。この環境データはローカルに格納され、データはクラウドに階層化されます。
- * 最小、最大、アダプティブの Quality of Service （ QoS ；サービス品質）。 * きめ細かい QoS 管理機能により、高度に共有された環境で重要なアプリケーションのパフォーマンスレベルを維持できます。
- * ONTAP FabricPool 。*この機能は、Amazon Web Services（AWS）、Azure、NetApp StorageGRID オブジェクトストレージなどのパブリックおよびプライベートクラウドストレージオプションに、コールドデータを自動的に階層化します。

データの高速化と保護

ONTAP は、卓越したパフォーマンスとデータ保護を実現し、以下の方法でこれらの機能を拡張します。

- * パフォーマンスと低レイテンシ。 * ONTAP は、可能な限り低いレイテンシで最高のスループットを提供します。
- * データ保護。 * ONTAP は、組み込みのデータ保護機能を提供し、すべてのプラットフォームで共通の管理を実現します。
- * NetApp Volume Encryption* ONTAP は、オンボードと外部の両方のキー管理をサポートし、ボリュームレベルのネイティブ暗号化を実現します。

将来のニーズにも対応できるインフラ

ONTAP 9は、要求の厳しい絶えず変化するビジネスニーズに対応するのに役立ちます。

- シームレスな拡張とノンストップオペレーション。 ONTAP は、既存のコントローラやスケールアウトクラスタに無停止で容量を追加できます。NVMe や 32Gb FC などの最新テクノロジへのアップグレードも、コストのかかるデータ移行やシステム停止を行わずに実行できます。
- * クラウドへの接続。 * ONTAP は、すべてのパブリッククラウドで Software-Defined Storage （ ONTAP Select ）とクラウドネイティブインスタンス（ NetApp Cloud Volumes Service ）を選択できる、最もクラウドに接続されたストレージ管理ソフトウェアです。
- 新しいアプリケーションとの統合。 ONTAP は、既存のエンタープライズアプリケーションをサポートする同じインフラを使用して、OpenStack、Hadoop、MongoDBなどの次世代プラットフォームやアプリケーションにエンタープライズクラスのデータサービスを提供します。

NetApp FlexGroup ボリューム

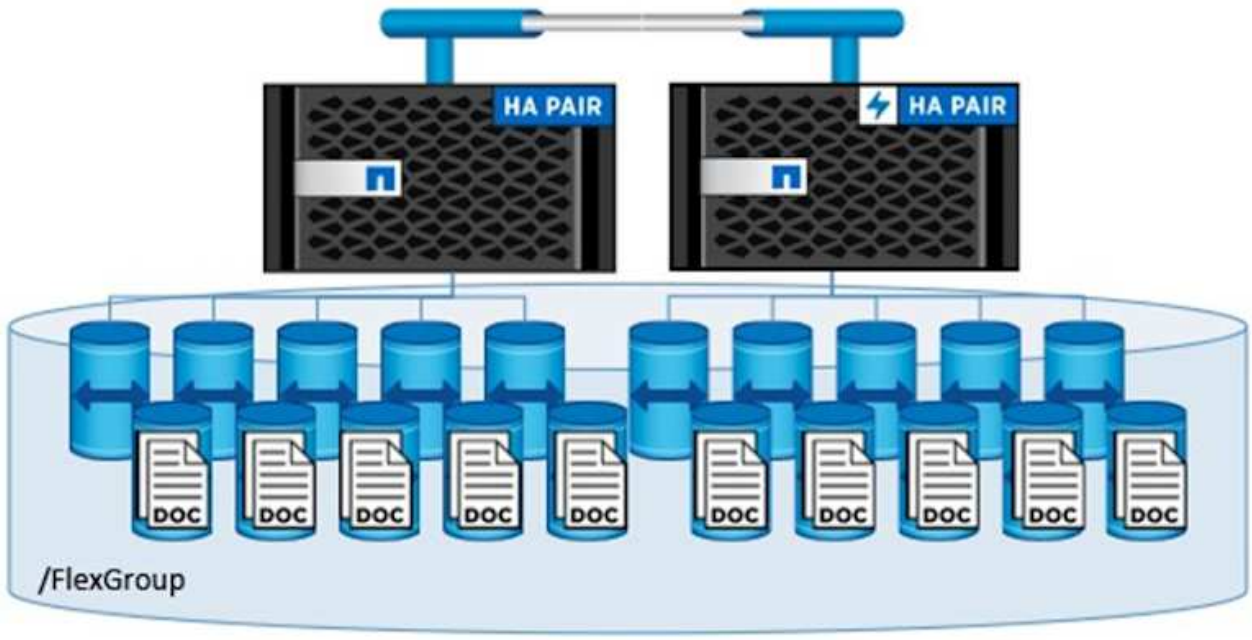
一般に、トレーニングデータセットは数十億個のファイルからなる可能性があります。ファイルには、テキスト、オーディオ、ビデオなどの形式の非構造化データを含めることができます。これらのデータは、並行して読み込まれるように保存して処理する必要があります。ストレージシステムは、多数の小さなファイルを格納し、シーケンシャルI/OとランダムI/Oの両方で、それらのファイルを並列に読み取る必要があります

FlexGroup ボリューム（次の図）は、複数のコンスティチュエントメンバーボリュームで構成された単一のネームスペースで、ストレージ管理者はNetApp FlexVol と同様に使用および管理することができます。FlexGroup ボリューム内のファイルは、個々のメンバーボリュームに割り当てられ、複数のボリュームやノードにまたがってストライプされることはありません。次の機能が有効になります。

- メタデータ比率の高いワークロード向けに、最大20ペタバイトの容量と予測可能な低レイテンシを実現し

ます

- 同じネームスペースに最大4、000億個のファイルを格納できます
- CPU、ノード、アグリゲート、コンスティチュエントFlexVol ボリューム全体でNASワークロードの並行処理をサポート



"

Lenovo ThinkSystemポートフォリオ

Lenovo ThinkSystem サーバは、革新的なハードウェア、ソフトウェア、サービスを搭載しており、お客様の現在の課題を解決し、将来の課題に対処するための、進化した、用途に合わせたモジュラー設計アプローチを提供します。これらのサーバは、クラス最高の業界標準テクノロジーと、差別化された Lenovo の革新技術を組み合わせて、 x86 サーバで可能な限り高い柔軟性を提供します。

Lenovo ThinkSystemサーバを導入する主なメリットは次のとおりです。

- ビジネスとともに成長する拡張性に優れたモジュラ設計
- 業界をリードする耐障害性により、計画外停止にかかるコストを時間単位で削減します
- 高速フラッシュテクノロジにより、レイテンシを低減し、応答時間を短縮し、リアルタイムでのデータ管理をスマートに実現します

Lenovo は、 AI 分野において、企業がワークロードに ML と AI のメリットを理解し、採用できるようにするための実践的なアプローチをとっています。Lenovo のお客様は、 Lenovo AI Innovation Center で Lenovo AI 製品を調査および評価し、特定のユースケースの価値を十分に理解することができます。価値実現までの時間を短縮するために、このお客様中心のアプローチでは、AI向けに最適化され、すぐに使用できる解決策 開発プラットフォームのコンセプトの実証が可能です。

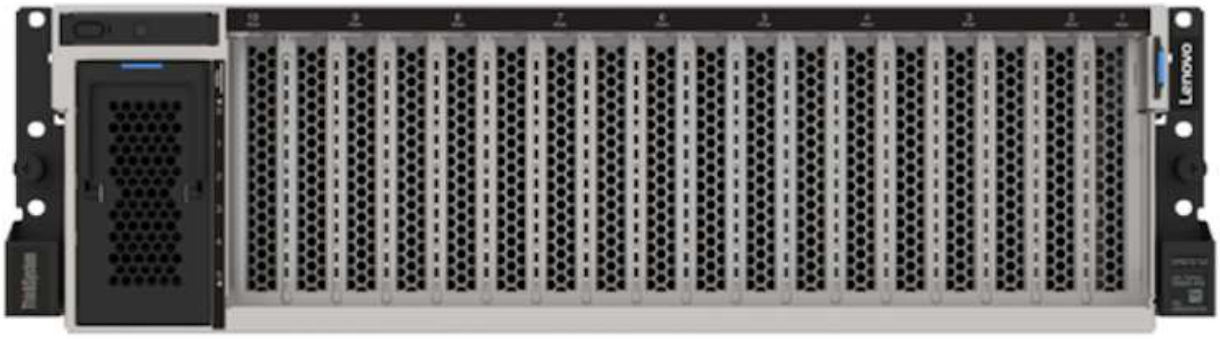
Lenovo SR670 V2

Lenovo ThinkSystem SR670 V2ラックサーバは、AIの高速化とハイパフォーマンスコンピューティング（HPC）に最適なパフォーマンスを提供します。SR670 V2は、最大8基のGPUをサポートし、機械学習、

DL、推論の負荷が高いワークロード要件に対応します。

4x SXM GPUs with 8x 2.5-inch HS drives and 2x PCIe I/O slots

4x double-wide or 8x single-wide GPU slots and 2x PCIe I/O slots
with 8x 2.5-inch or 4x 3.5-inch HS drives



8x double-wide GPU slots with 6x EDSFF HS drives and 2x PCIe I/O slots

ハイエンドGPU（NVIDIA A100 80GB PCIe 8x GPUを含む）をサポートする最新の拡張性に優れたIntel Xeon CPUを搭載したThinkSystem SR670 V2は、AIおよびHPCワークロード向けに最適化された高速なパフォーマンスを提供します。

アクセラレータのパフォーマンスを使用するワークロードが増えるにつれて、GPU密度の需要も増加しています。小売、金融サービス、エネルギー、医療などの業界では、GPUを使用して分析情報を引き出し、ML、DL、推論の手法でイノベーションを推進しています。

ThinkSystem SR670 V2は、高速化されたHPCおよびAIワークロードを本番環境に導入するためのエンタープライズクラスの最適化された解決策 で、次世代プラットフォームを使用したスーパーコンピューティングクラスタのデータセンター密度を維持しながら、システムパフォーマンスを最大限に高めます。

その他の機能は次のとおりです。

- GPUに高速ネットワークアダプタを直接接続し、I/Oパフォーマンスを最大化するGPU直接RDMA I/Oのサポート。
- NVMeドライブをGPUに直接接続してストレージのパフォーマンスを最大限に高める、GPU直接ストレージのサポート。

MLPerf

MLPerf は、 AI のパフォーマンスを評価するための業界をリードするベンチマークスイートです。この検証では、最も人気の高いAIフレームワークの1つであるMXNetで画像分類ベンチマークを使用しました。MXNet_benchmarksトレーニングスクリプトは、AIトレーニングの促進に使用されました。このスクリプトには、複数の一般的な従来型モデルの実装が含まれており、できる限り高速になるように設計されています。単一のマシンで実行することも、複数のホスト間で分散モードで実行することもできます。

テスト計画

この検証では、MLPerf v2.0の規定に従って画像認識トレーニングを実施しました。具体的には、ImageNetデータセットを使用してResNet v2.0モデルのトレーニングを行い、精度が76.1%になるようにしました。主な指標は、目的の精度に到達するまでの時間です。また、1秒あたりの画像数でトレーニングの帯域幅をレポートし、スケールアウトの効率性をより適切に判断します。

プライマリテストケースで、複数の独立したトレーニングプロセス（ノードごとに1つ）を同時に実行して評価しました。これは、複数のデータサイエンティストが使用する共有システムである主なユースケースをシミュレートします。2つ目のテストケースでは、スケールアウトの効率性を評価しました。

テスト結果

次の表は、この解決策 に対して実行されたすべてのテストの結果をまとめたものです。

概要 をテストします	結果の概要
画像認識トレーニング：複数のジョブを同時に実行できます	効率性に優れたパフォーマンス。クラスタが完全に使用されていても、すべてのジョブがフルスピードで実行されました。ネットアップのストレージシステムは、ローカルのSSDストレージと同等のトレーニングパフォーマンスを提供すると同時に、サーバ間でデータを簡単に共有できるようにしました。
画像認識トレーニング：スケールアウト	最大4ノードまで効率性に優れています。この時点では、スケールアウトの効率は低下していましたが、まだ実現可能です。高速のコンピューティングネットワークを使用すると、拡張性が向上します。ネットアップのストレージシステムは、ローカルのSSDストレージと同等のトレーニングパフォーマンスを提供すると同時に、サーバ間でデータを簡単に共有できるようにしました。

設定をテストします

このセクションでは、テストした構成、ネットワークインフラ、SR670 V2サーバ、およ

びネットアップストレージプロビジョニングの詳細について説明します。

解決策アーキテクチャ

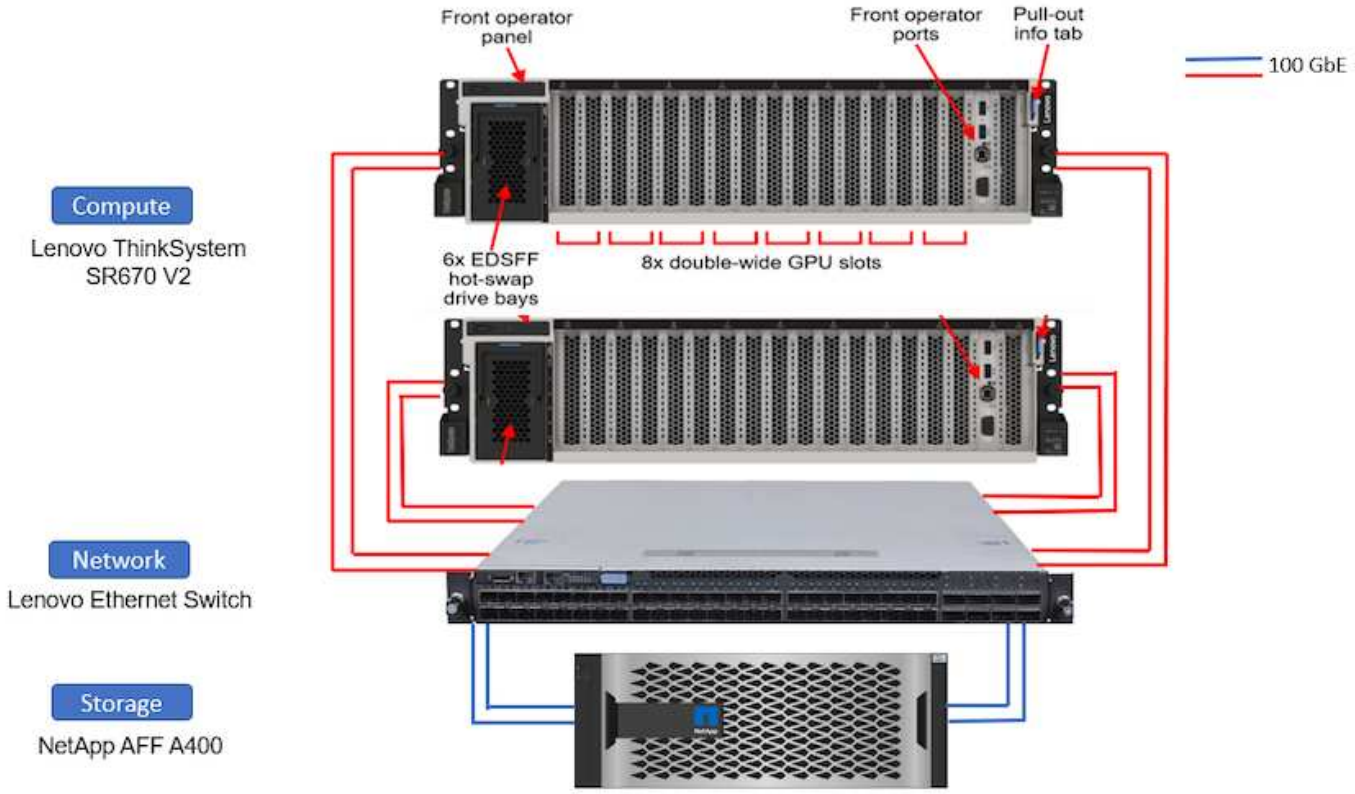
次の表に示す解決策 コンポーネントを使用して、この検証を実施しました。

解決策コンポーネント	詳細
Lenovo ThinkSystem サーバ	• それぞれ8台のNVIDIA A100 80GB GPUカードを搭載したSR670 V2サーバ2台 • 各サーバには、Intel Xeon Platinum 8360Y CPU（28個の物理コア）2個と1TBのRAMが搭載されています
Linux（Ubuntu–20.04とCUDA 11.8）	
NetApp AFF ストレージシステム（HAペア）	• NetApp ONTAP 9.10.1ソフトウェア • 960GB SSD × 24 • NFS プロトコル • コントローラごとに1つのインターフェイスグループ（ifgrp）。マウントポイント用に4つの論理IPアドレスが必要です

この検証では、MLPerf v2.0で指定されたImageNetベースでResNet v2.0を使用しました。データセットは、NFSプロトコルを使用してNetApp AFF ストレージシステムに保存されます。SR670sは、100GbEスイッチ経由でNetApp AFF A400ストレージシステムに接続されています。

ImageNetは、頻繁に使用されるイメージデータセットです。これには、合計で144 GBの約130万枚の画像が含まれています。平均画像サイズは108KBです。

次の図は、テストした構成のネットワークトポロジを示しています。



ストレージコントローラ

次の表に、ストレージ構成を示します。

コントローラ	アグリゲート	**FlexGroup** ボリューム	アグリゲートのサイズ	ボリュームサイズ	オペレーティングシステムのマウントポイント
コントローラ 1	aggr1	/ 400-100g	9.9TB	19TB	/ 400-100g
コントローラ 2	aggr2	/ 400-100g	9.9TB		/ 400-100g

/400-100gフォルダには、ResNet検証に使用されるデータセットが含まれています。

手順 のテストと詳細な結果

このセクションでは、手順 の詳細なテスト結果について説明します。

ONTAP の**ResNet**を使用した画像認識トレーニング

1台および2台のSR670 V2サーバでResNet50ベンチマークを実行しました。このテストでは、MXNet 22.04-py3 NGCコンテナを使用してトレーニングを実行しました。

この検証では、次のテスト手順 を使用しました。

1. スクリプトを実行する前にホストキャッシュをクリアして、データがキャッシュされていないことを確認しました。

```
sync ; sudo /sbin/sysctl vm.drop_caches=3
```

2. ベンチマークスクリプトを実行し、サーバストレージ（ローカルSSDストレージ）とNetApp AFF ストレージシステムのImageNetデータセットを使用しました。
3. を使用してネットワークとローカルストレージのパフォーマンスを検証しました `dd` コマンドを実行します
4. シングルノードの実行では、次のコマンドを使用しました。

```
python train_imagenet.py --gpus 0,1,2,3,4,5,6,7 --batch-size 408 --kv
-store horovod --lr 10.5 --mom 0.9 --lr-step-epochs pow2 --lars-eta
0.001 --label-smoothing 0.1 --wd 5.0e-05 --warmup-epochs 2 --eval-period
4 --eval-offset 2 --optimizer sgdwfastlars --network resnet-v1b-stats-fl
--num-layers 50 --num-epochs 37 --accuracy-threshold 0.759 --seed 27081
--dtype float16 --disp-batches 20 --image-shape 4,224,224 --fuse-bn-relu
1 --fuse-bn-add-relu 1 --bn-group 1 --min-random-area 0.05 --max-random
-area 1.0 --conv-algo 1 --force-tensor-core 1 --input-layout NHWC --conv
-layout NHWC --batchnorm-layout NHWC --pooling-layout NHWC --batchnorm
-mom 0.9 --batchnorm-eps 1e-5 --data-train /data/train.rec --data-train
-idx /data/train.idx --data-val /data/val.rec --data-val-idx
/data/val.idx --dali-dont-use-mmap 0 --dali-hw-decoder-load 0 --dali
-prefetch-queue 5 --dali-nvjpeg-memory-padding 256 --input-batch
-multiplier 1 --dali- threads 6 --dali-cache-size 0 --dali-roi-decode 1
--dali-preallocate-width 5980 --dali-preallocate-height 6430 --dali-tmp
-buffer-hint 355568328 --dali-decoder-buffer-hint 1315942 --dali-crop
-buffer-hint 165581 --dali-normalize-buffer-hint 441549 --profile 0
--e2e-cuda-graphs 0 --use-dali
```

5. 分散ランでは、パラメータサーバの並列化モデルを使用しました。ノードごとに2つのパラメータサーバを使用し、エポックの数をシングルノード実行と同じに設定しました。これは、分散トレーニングではプロセス間の完全な同期が行われないため、多くの場合、エポックの数が多くなるためです。エポックの数が異なると、シングルノードケースと分散ケースの比較が歪む場合があります。

データの読み取り速度：ローカルストレージとネットワークストレージの比較

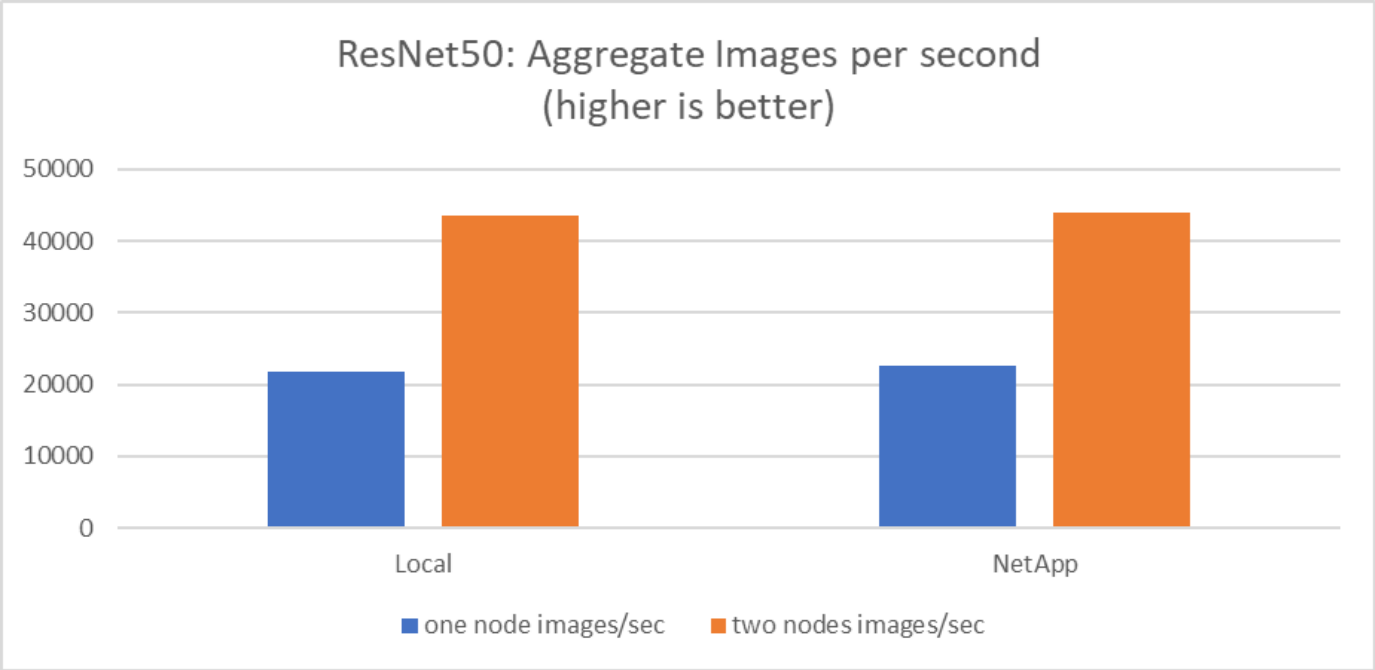
読み取り速度は、を使用してテストしました `dd` ImageNetデータセットのいずれかのファイルに対してコマンドを実行します。具体的には、ローカルデータとネットワークデータに対して次のコマンドを実行しました。

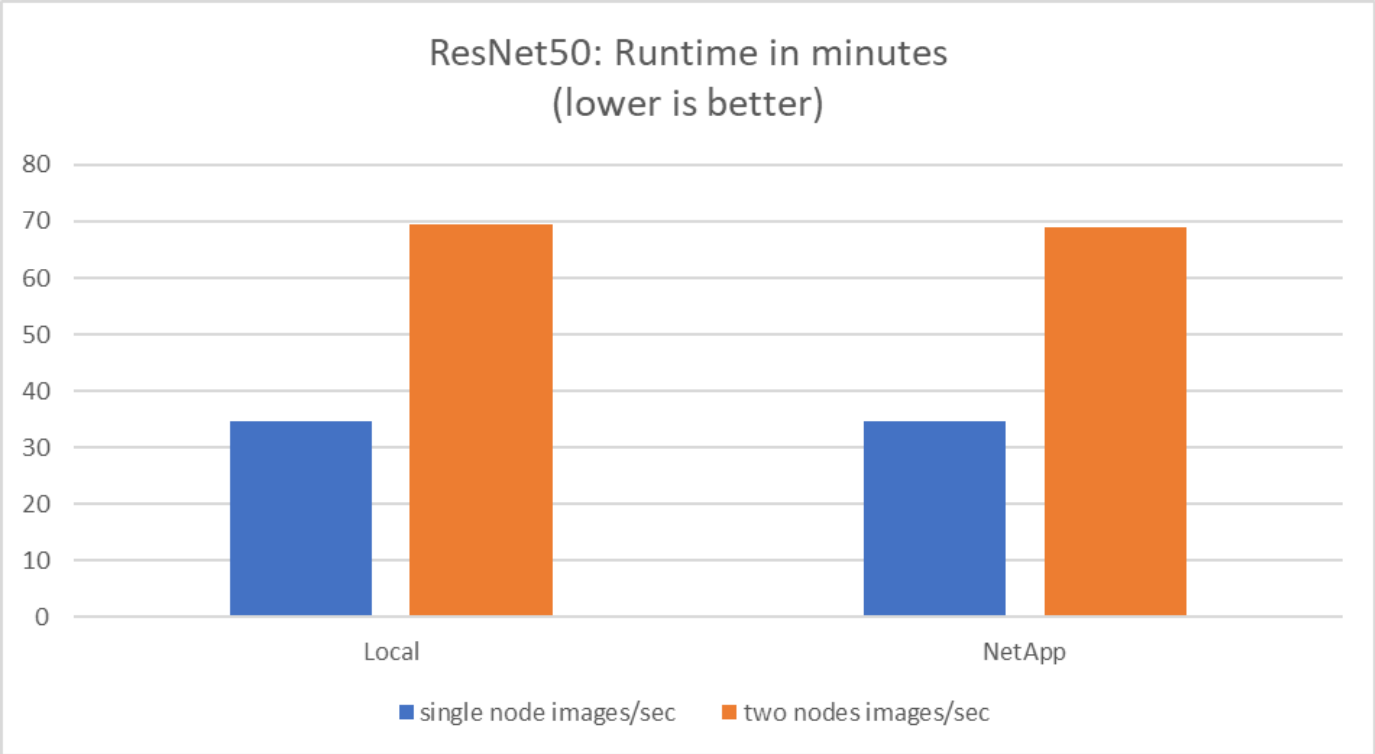
```
sync ; sudo /sbin/sysctl vm.drop_caches=3dd if=/a400-100g/netapp-
ra/resnet/data/preprocessed_data/train.rec of=/dev/null bs=512k
count=2048Results (average of 5 runs):
Local storage: 1.7 GB/s Network storage: 1.5 GB/s.
```

どちらの値もほぼ同じで、ネットワークストレージがローカルストレージとほぼ同じ速度でデータを提供できることを示しています。

共有の使用事例：複数の独立した同時ジョブ

このテストでは、この解決策 で想定されるユースケースをシミュレートしました。マルチジョブ、マルチユーザAIトレーニングです。共有ネットワークストレージを使用する際には、各ノードで独自のトレーニングを実行しました。次の図に、解決策 ケースでは、すべてのジョブが個 々 のジョブと基本的に同じ速度で実行されている場合に優れたパフォーマンスが得られたことを示します。合計スループットはノード数に比例して増加しました。





このグラフは、同時トレーニングモデルと単一のトレーニングモデルを組み合わせた、100GbEクライアントネットワーク上の各サーバから8基のGPUを使用したコンピューティングノードのランタイムを分単位で示し、1秒あたりの総イメージを示しています。トレーニングモデルの平均実行時間は35分9秒でした。各ラン

タイムは34分32秒、36分21秒、34分37秒、35分25秒、34分31秒でした。トレーニングモデルの1秒あたりの平均画像数は22、573で、1秒あたりの個別画像数は21、764、23、438、22、556、22、564、22、547でした。

今回の検証に基づき、ネットアップのデータランタイムを使用した独立型トレーニングモデルは34分54秒、画像数は22、231秒でしたローカルデータ(DAS)ランタイムを持つ独立したトレーニングモデルは、34分21秒、22,102枚/秒でしたNVIDIAとSMIで見られた平均GPU利用率は96%でした。この平均値には、GPUが使用されていないテストフェーズが含まれ、CPU利用率はmpstatで測定された40%であることに注意してください。これは、それぞれのケースでデータ配信率が十分であることを示しています。

アーキテクチャの調整

この検証に使用するセットアップは、他のユースケースに合わせて調整できます。

CPU調整

この検証には、Lenovoが推奨するSkylake Intel Xeon Platinum 8360Yプロセッサを使用しました。Intel Xeon Gold 6330プロセッサと同等のCascade Lake CPUが同等のパフォーマンスを発揮することが期待されています。これは、このワークロードがCPUに制限されていないためです。

ストレージ容量が拡張されました

追加のディスクシェルフとコントローラモデルがある場合は、必要なストレージ容量に基づいて、共有ストレージ（NFSボリューム）をオンデマンドで拡張できます。管理ユーザは、CLIまたはストレージコントローラのネットアップWebインターフェイスから実行できます。

まとめ

ネットアップとLenovo解決策 は、柔軟性に優れたスケールアウトアーキテクチャを採用しており、ミッドレベルのエンタープライズAI導入に最適です。

ネットアップのストレージは、ローカルのSSDストレージと同等以上のパフォーマンスを提供し、データサイエンティスト、データエンジニア、IT意思決定者に次のようなメリットをもたらします。

- AI システム、分析などの重要なビジネスシステム間でデータを容易に共有できます。このようなデータ共有により、インフラのオーバーヘッドを削減し、パフォーマンスを向上させ、企業全体のデータ管理を合理化できます。
- 個別に拡張可能なコンピューティングとストレージにより、コストを最小限に抑え、リソースの使用率を向上させます。
- 統合されたSnapshotとクローンを使用して開発と導入のワークフローを合理化し、ユーザのワークスペースを瞬時にスペース効率よく利用できるようにし、バージョン管理を統合し、導入を自動化します。
- ディザスタリカバリとビジネス継続性を実現するエンタープライズクラスのデータ保護

謝辞

- ネットアップテクニカルマーケティングエンジニアKarthikeyan Nagalingam氏
- Lenovo、AIラボシステム、管理者Jarrett Upton氏

追加情報の参照先

このドキュメントに記載されている情報の詳細については、以下のドキュメントや Web サイトを参照してください。

- NetApp All Flash Arraysの製品ページです

  "https://www.netapp.com/us/products/storage-systems/all-flash-array/aff-a-series.aspx"

- NetApp AFF A400のページです

  "https://docs.netapp.com/us-en/ontap-systems/a400/index.html"

- NetApp ONTAP データ管理ソフトウェアの製品紹介ページ

  "http://www.netapp.com/us/products/data-management-software/ontap.aspx"

- MLPerf

  "https://mlperf.org"

- TensorFlow ベンチマーク

  "https://github.com/tensorflow/benchmarks"

- NVIDIA SMI（NVIDIA - SMI）

  "https://developer.nvidia.com/nvidia-system-management-interface"

NVIDIA搭載NetApp AI

ネットアップと NVIDIA が提供する ONTAP AI 統合インフラソリューションの概要

NVIDIA DGX A100 システムを搭載した **NetApp ONTAP AI**

- "設計ガイド"
- "導入ガイド"

NVIDIA DGX A100 システムと **Mellanox** を搭載した **NetApp ONTAP AI Spectrum Ethernet** スイッチ

- "設計ガイド"
- "導入ガイド"

NVA-1151-design：NVIDIA DGX A100を搭載した**NetApp ONTAP AI**のシステム設計ガイド

ネットアップDavid ArnetteとSung-Han Lin

NVA-1151設計には、NetApp AFF A800ストレージシステム、NVIDIA DGX A100システム、NVIDIA Mellanoxネットワークスイッチを使用した、マシンラーニングと人工知能のワークロードに対応するネットアップの検証済みアーキテクチャが記載されています。また、実装されているアーキテクチャのベンチマークテスト結果も含まれます。

"NVA-1151-design：NVIDIA DGX A100を搭載したNetApp ONTAP AIのシステム設計ガイド"

NVA-1151-deploy：NVIDIA DGX A100システムを搭載した**NetApp ONTAP AI**

ネットアップ、David Arnette

NVA-1151 - NetApp AFF A800ストレージシステム、NVIDIA DGX A100システム、NVIDIA Mellanoxネットワークスイッチを使用した、機械学習（ML）と人工知能（AI）のワークロード向けのNetApp Verified Architecture（NVA）でのストレージシステムの導入手順を記載します。また、導入完了後に検証ベンチマークテストを実行する手順についても説明します。

"NVA-1151-deploy：NVIDIA DGX A100システムを搭載したNetApp ONTAP AI"

NVA-1153設計：**NVIDIA DGX A100**システムと**Mellanox Spectrum**イーサネットスイッチを搭載した**NetApp ONTAP AI**

ネットアップDavid ArnetteとSung-Han Lin

NVA-1153 -設計には、NetApp AFF A800ストレージシステム、NVIDIA DGX A100システム、NVIDIA Mellanox NVIDIA Spectrum SN3700V 200GBのイーサネットスイッチを使用した、機械学習（ML）と人工知能（AI）のワークロードに対応するネットアップの検証済みアーキテクチャが記述されています。この設計では、コンピューティングクラスタインターコネクトファブリック用のRDMA over Converged Ethernet（RoCE）を特徴としており、ハイパフォーマンスワークロード向けに完全なイーサネットベースのアーキテクチャを提供します。また、実装されているアーキテクチャのベンチマークテスト結果も記載します。

"NVA-1153設計：NVIDIA DGX A100システムとMellanox Spectrumイーサネットスイッチを搭載したNetApp ONTAP AI"

NVA-1153 -導入：**NVIDIA DGX A100**システムと**Mellanox Spectrum**イーサネットスイッチを搭載した**NetApp ONTAP AI**

ネットアップ、David Arnette

NVA-1153 - NetApp AFF A800ストレージシステム、NVIDIA DGX A100システム、NVIDIA Mellanox Spectrum SN3700V 200GBイーサネットスイッチを使用した、機械学習（ML）と人工知能（AI）のワークロード向けのNetApp Verified Architectureでのストレージシステムの導入手順を記載します。また、導入完了後に検証ベンチマークテストを実行する手順についても説明します。

"NVA-1153 -導入：NVIDIA DGX A100システムとMellanox Spectrumイーサネットスイッチを搭載したNetApp ONTAP AI"

NVIDIA を使用した **NetApp EF** シリーズ **AI**

ネットアップと NVIDIA が提供する EF シリーズ AI 統合インフラソリューションの概要

NVIDIA DGX A100 システムと **BeeGFS** を搭載した **EF** シリーズ **AI**

- "設計ガイド"
- "導入ガイド"
- "BeeGFS 導入ガイド"

NVA-1156設計：**NVIDIA DGX A100**システムと**BeeGFS**を搭載した**NetApp EF**シリーズ**AI**

Abdel Sadek、Tim Chau、Joe McCormick、David Arnette、ネットアップ

NVA-1156 -設計には、NetApp EF600 NVMeストレージシステム、BeeGFS並列ファイルシステム、NVIDIA DGX A100システム、NVIDIA Mellanox Quantum QM8700 200Gbps IBスイッチを使用した、機械学習（ML）と人工知能（AI）のワークロードに対応したネットアップの検証済みアーキテクチャが記述されています。この設計では、ストレージとコンピューティングクラスタのインターコネクトファブリックに12Gbps InfiniBand（IB）を採用して、ハイパフォーマンスワークロード向けの完全なIBベースのアーキテクチャをお客様に提供しています。また、実装されているアーキテクチャのベンチマークテスト結果も記載します。

"NVA-1156設計：NVIDIA DGX A100システムとBeeGFSを搭載したNetApp EFシリーズAI"

NVA-1156 -導入：**NVIDIA DGX A100**システムと**BeeGFS**を搭載した**NetApp EF**シリーズ**AI**

Abdel Sadek、Tim Chau、Joe McCormick、ネットアップDavid Arnette

このドキュメントでは、NetApp EF600 NVMeストレージシステム、ThinkParQ BeeGFS並列ファイルシステム、NVIDIA DGX A100システム、NVIDIA Mellanox Quantum QM8700 200Gbps InfiniBand（IB）スイッチを使用した、機械学習（ML）および人工知能（AI）ワークロード向けのNetApp Verified Architectureについて説明します。このドキュメントには、導入完了後に検証ベンチマークテストを実行する手順も記載されています。

"NVA-1156 -導入：NVIDIA DGX A100システムとBeeGFSを搭載したNetApp EFシリーズAI"

TR-4859：『Deploying IBM spectrum scale with NetApp E-Series storage-Installation and validation』

ネットアップ、Chris Seirer

TR-4859では、IBMのSpectrum Scaleソフトウェアスタックに基づいて、フル並列ファイルシステムの解決策 を導入するプロセスについて説明しています。TR-4859は、Spectrum Scaleのインストール、インフラの検証、構成の管理の方法に関する詳細を提供するように設計されています。

"TR-4859：『Deploying IBM spectrum scale with NetApp E-Series storage-Installation and validation』"

TR-48815：『NetApp AFF A800 and Fujitsu Server PRIMERGY GX2570 M5 for AI and ML model training workloads』

David Arnette、NetApp、大石隆、富士通

この解決策 は、ネットアップストレージシステムと富士通のサーバを使用して人工知能システムを導入するためのスケールアウトアーキテクチャに焦点を当てています。解決策 は、富士通GX2570サーバとNetApp AFF A800ストレージシステムを使用したMLperf v0.6モデルトレーニングベンチマークで検証されました。

"TR-48815：『NetApp AFF A800 and Fujitsu Server PRIMERGY GX2570 M5 for AI and ML model training workloads』"

データパイプライン、データレイク、管理

MLOps向けAWS FSx for NetApp ONTAP（FSxN）

作成者：
NetAppシニアデータ&アプリケーションサイエンティスト、Jian Jian（Ken）

このセクションでは、AIインフラ開発の実用的なアプリケーションについて詳しく説明し、FSxNを使用してMLOpsパイプラインを構築するためのエンドツーエンドのウォークスルーを提供します。3つの包括的な例で構成され、この強力なデータ管理プラットフォームを通じてMLOpsのニーズを満たすことができます。

以下の記事では、

1. "パート1 - AWS FSx for NetApp ONTAP（FSxN）をプライベートS3バケットとしてAWS SageMakerに統合する"
2. "パート2 - SageMakerのモデルトレーニングのデータソースとしてAWS FSx for NetApp ONTAP（FSxN）を活用"
3. "パート3 -簡易化されたMLOpsパイプラインの構築（CI/CT/CD）"

このセクションを終了すると、FSxNを使用してMLOpsプロセスを合理化する方法について十分に理解できるようになります。

パート**1 - AWS FSx for NetApp ONTAP**（**FSxN**）をプライベート**S3**バケットとして**AWS SageMaker**に統合する

作成者：
NetAppシニアデータ&アプリケーションサイエンティスト、Jian Jian（Ken）

はじめに

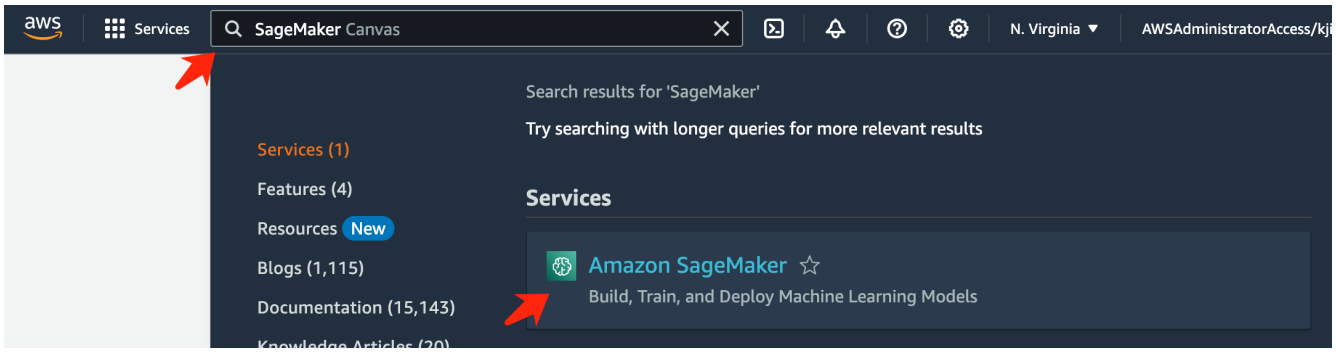
このページでは、SageMakerを例として使用して、FSxNをプライベートS3バケットとして設定する方法について説明します。

FSxNの詳細については、このプレゼンテーション（"ビデオリンク")

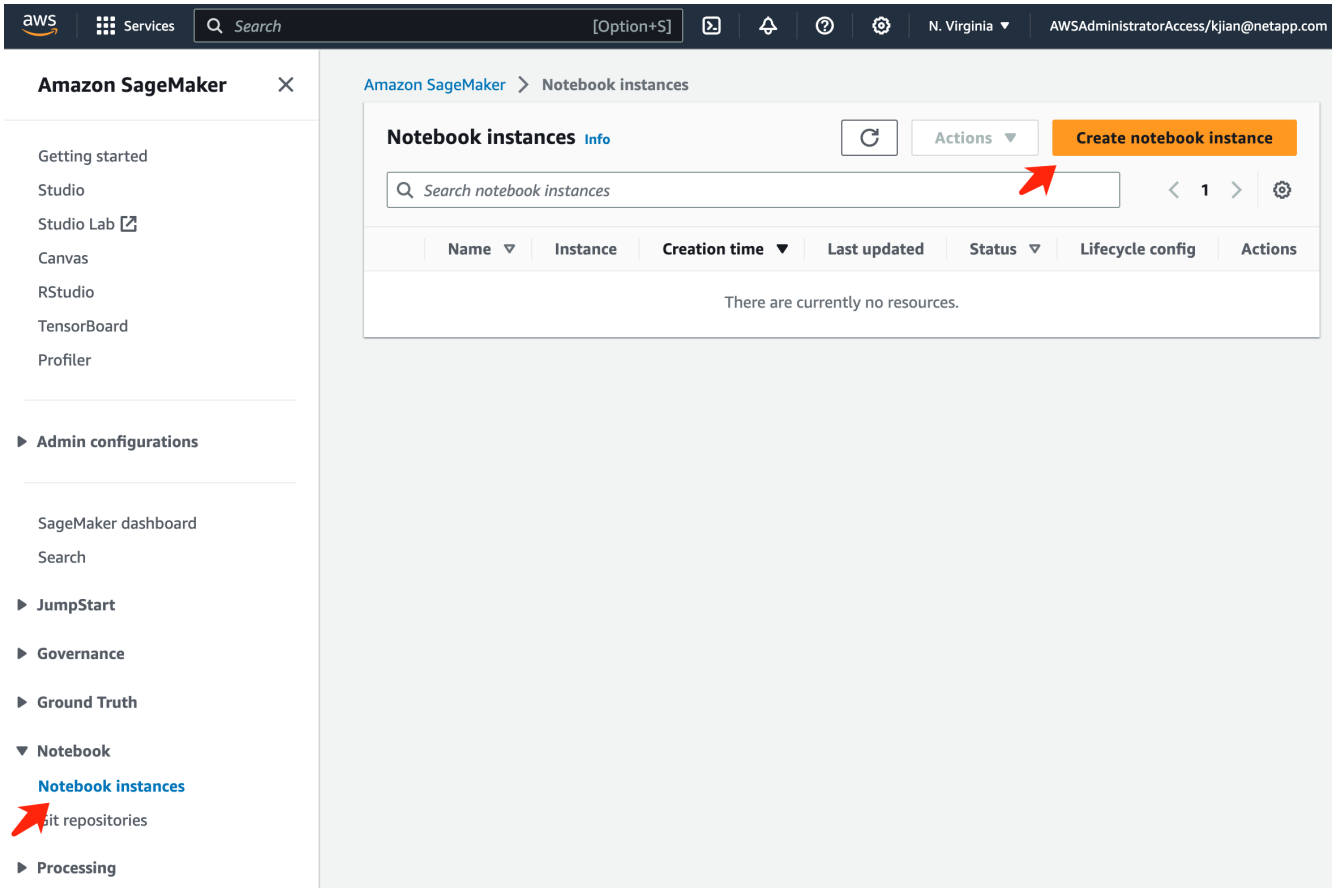
サーバの作成

SageMakerノートブックインスタンスの作成

1. AWSコンソールを開きます。検索パネルで、SageMakerを検索し、サービス* Amazon SageMaker *をクリックします。



2. Notebookタブの* Notebook Instances を開き、オレンジ色の Create notebook instance *ボタンをクリックします。



3. 作成ページで、
ノートブックインスタンス名*を入力します。
[ネットワーク]パネルを展開します。
[その他のエントリ]はデフォルトのままにして、**[VPC]**、**[サブネット]**、および*を選択します。（この

VPC と Subnet *は、あとでFSxNファイルシステムを作成するために使用されます）
右下にあるオレンジ色のボタン*ノートブックインスタンスの作成*をクリックします。

Amazon SageMaker > Notebook instances > Create notebook instance

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. Learn more

Notebook instance settings

Notebook instance name

fsxn-demo

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type

ml.t3.medium

Elastic Inference Learn more

none

Platform identifier Learn more

Amazon Linux 2, Jupyter Lab 3

▶ Additional configuration

Permissions and encryption

IAM role

Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the **AmazonSageMakerFullAccess** IAM policy attached.

AmazonSageMakerServiceCatalogProductsUseRole

Create role using the role creation wizard

Root access - *optional*

- (●) Enable - Give users root access to the notebook
- ( ) Disable - Don't give users root access to the notebook
  Lifecycle configurations always have root access

Encryption key - *optional*

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

▼ Network - *optional*

VPC - *optional*

Default vpc-0df3956ab1fca2ec9 (172.31.0.0/16)

Subnet

Choose a subnet in an availability zone supported by Amazon SageMaker.

subnet-00060df0d0f562672 (172.31.16.0/20) | us-east-1a

Security group(s)

sg-0a39b3985770e9256 (default) ✕

Direct internet access

- (●) Enable — Access the internet directly through Amazon SageMaker
- ( ) Disable — Access the internet through a VPC
  To train or host models from a notebook, you need internet access. To enable internet access, make sure that your VPC has a NAT gateway and your security group allows outbound connections. Learn more

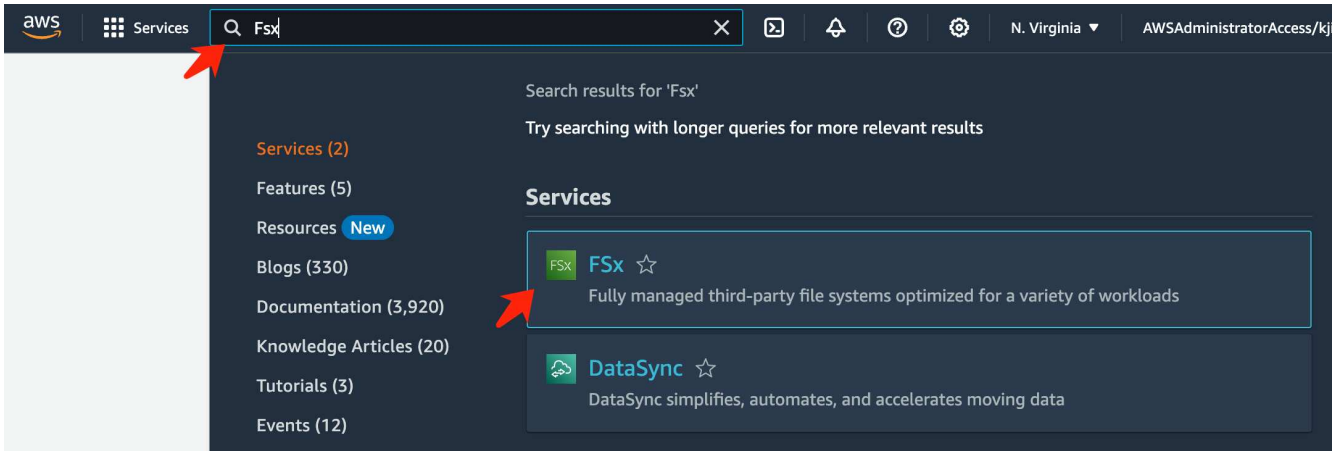
▶ Git repositories- *optional*

▶ Tags - *optional*

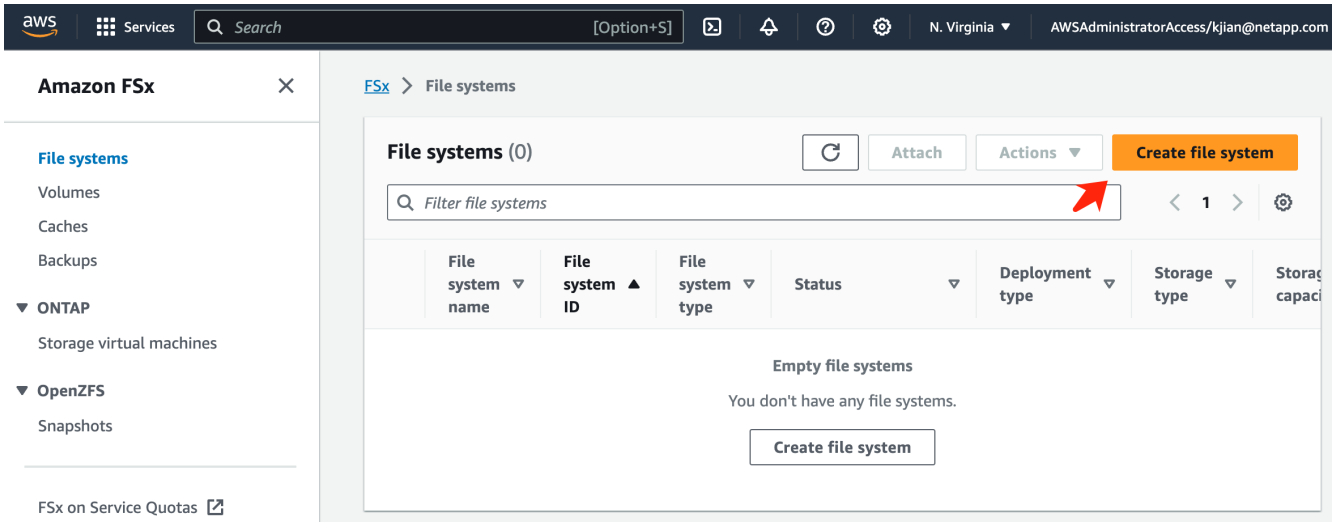
Cancel | Create notebook instance

FSxNファイルシステムの作成

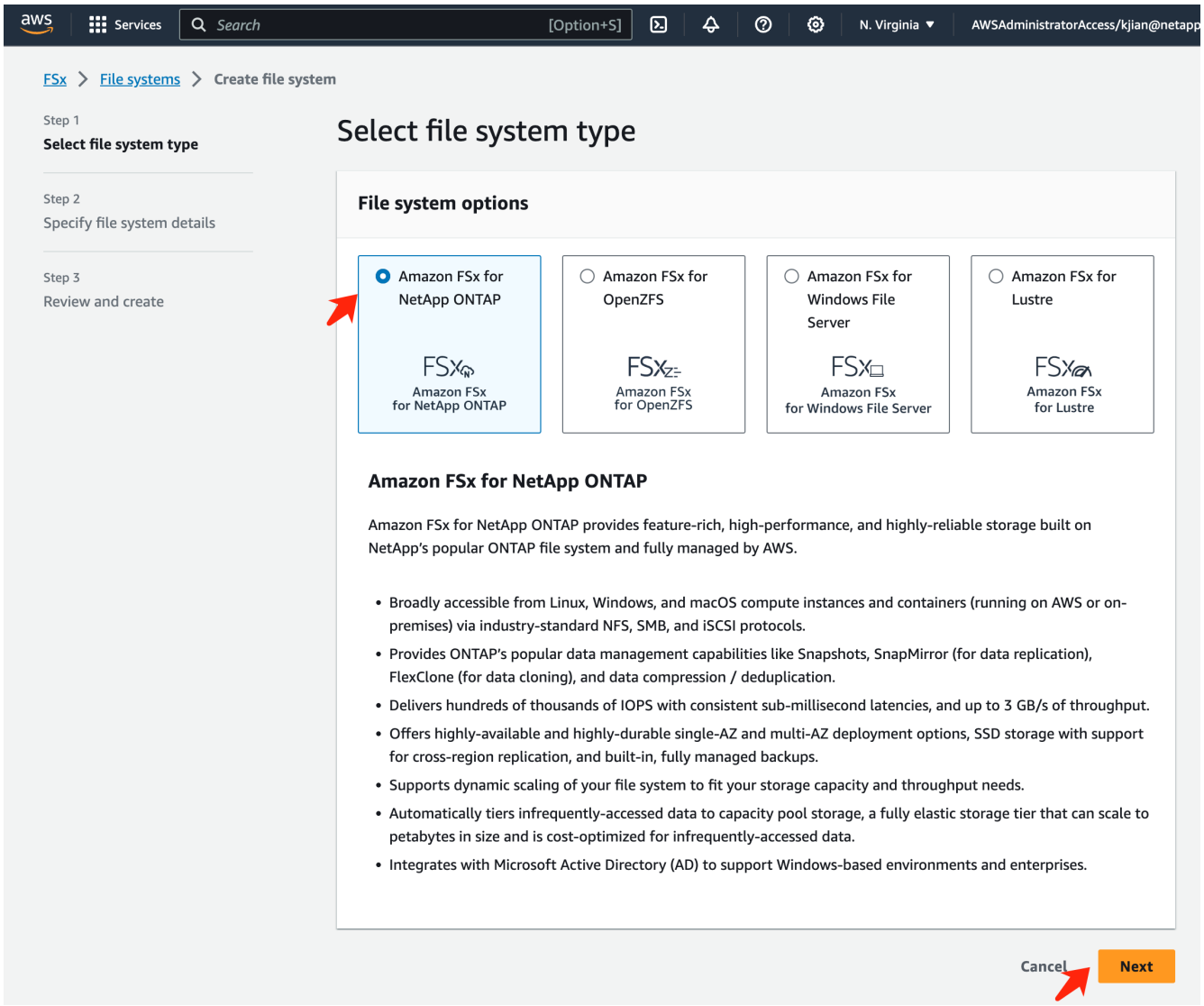
1. AWSコンソールを開きます。検索パネルでFSxを検索し、サービス* FSX *をクリックします。



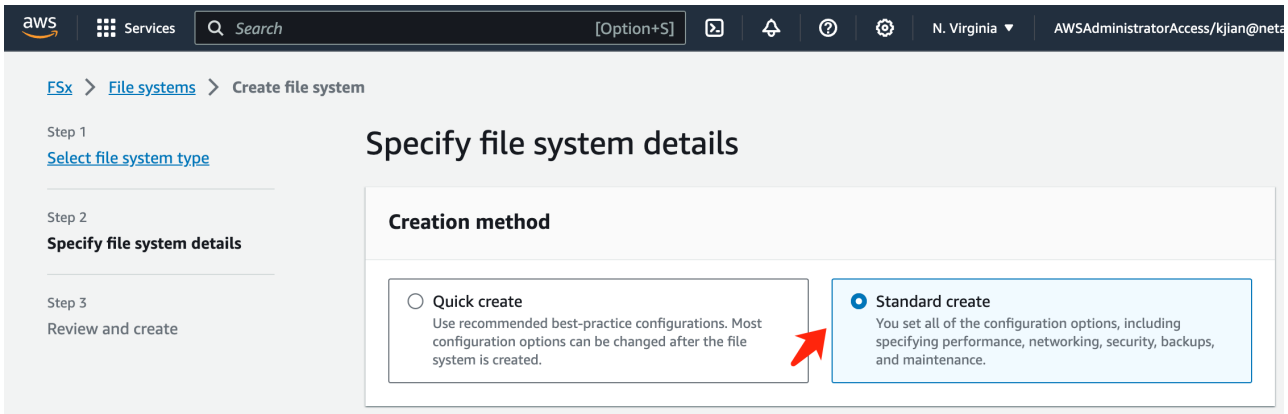
2. [ファイルシステムの作成]*をクリックします。



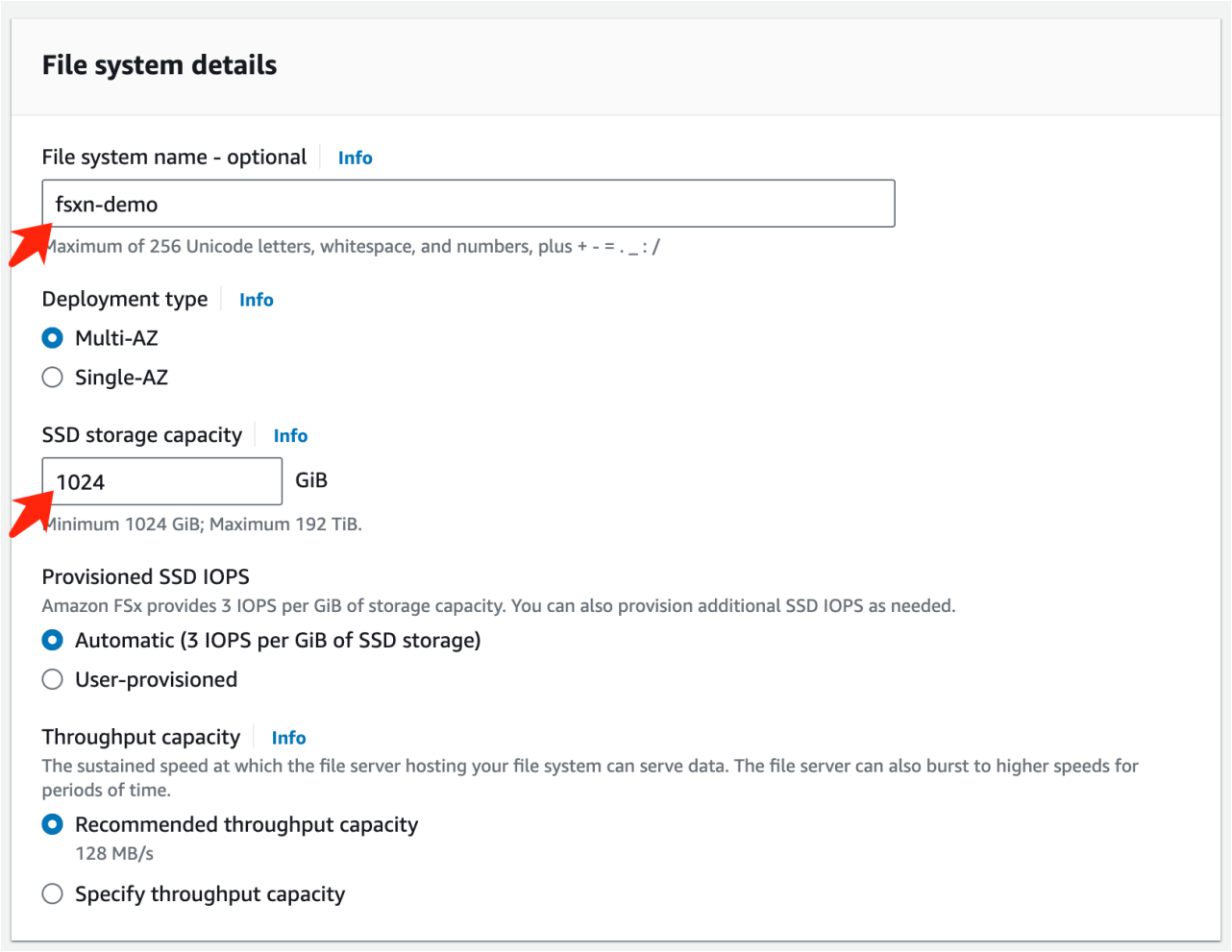
3. 最初のカード* FSx for NetApp ONTAP を選択し、 Next *をクリックします。



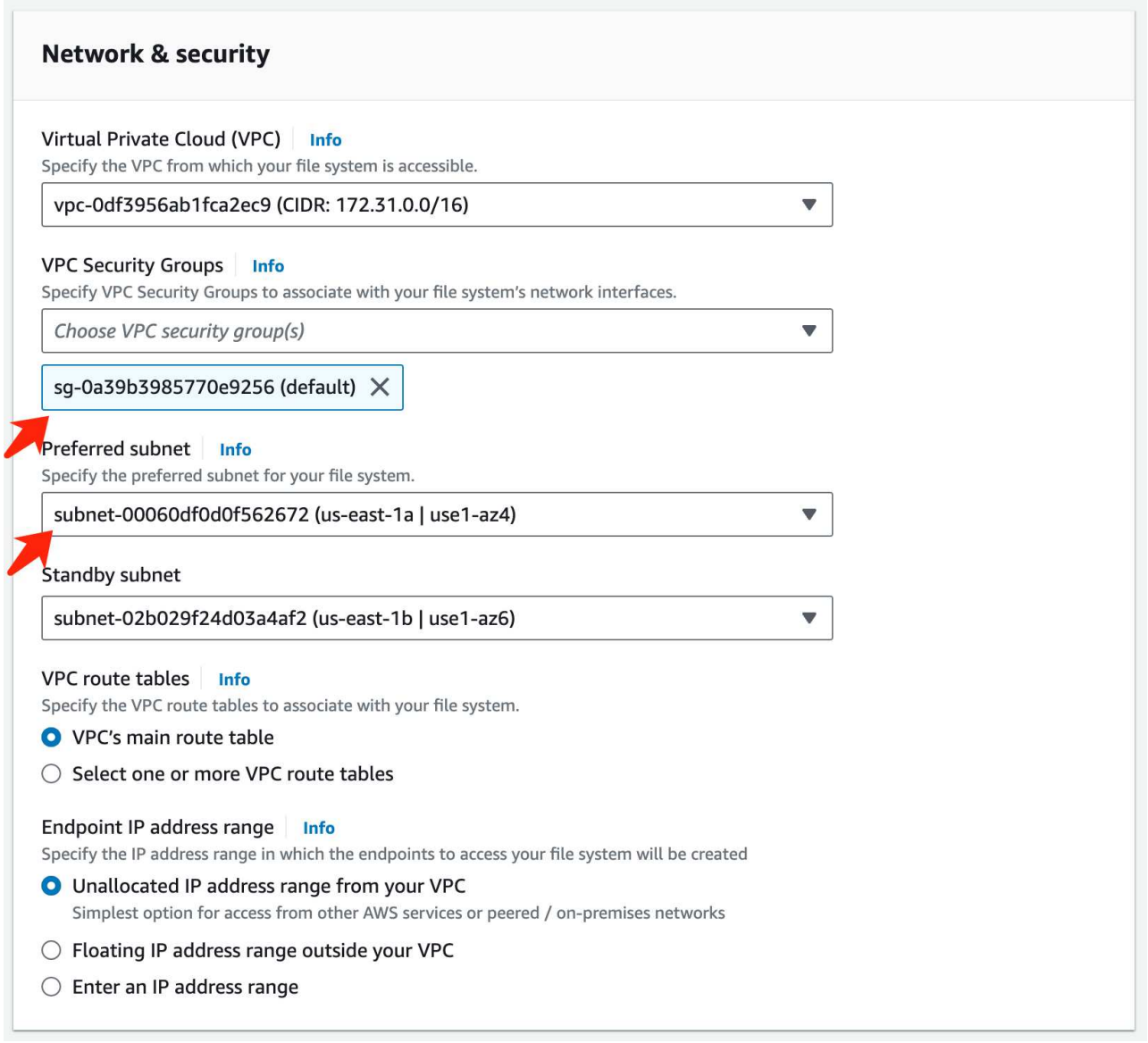
4. をクリックします。
   a. [標準作成（Standard create）]*オプションを選択します。



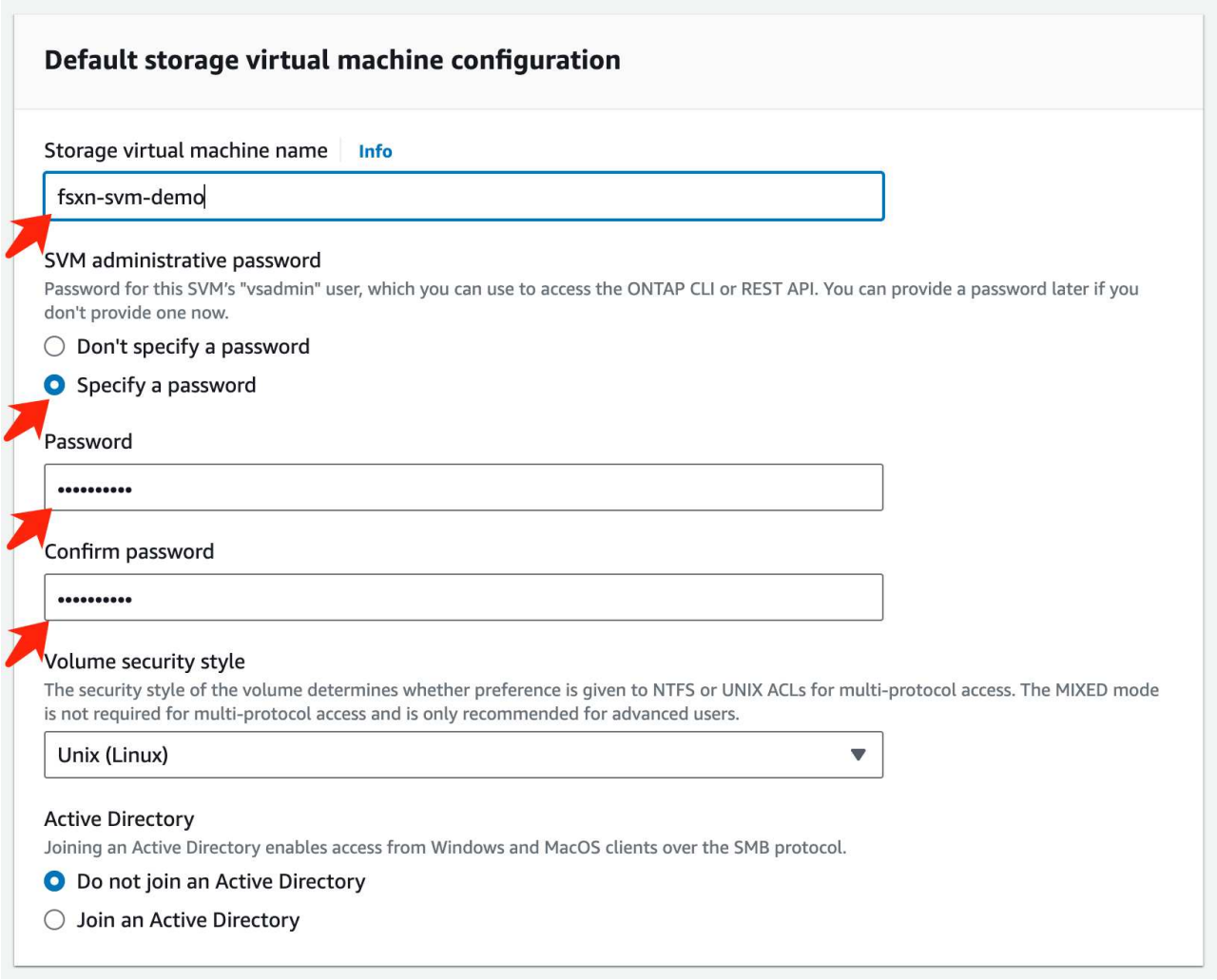
   b. [File system name]*と[SSD storage capacity]*を入力します。



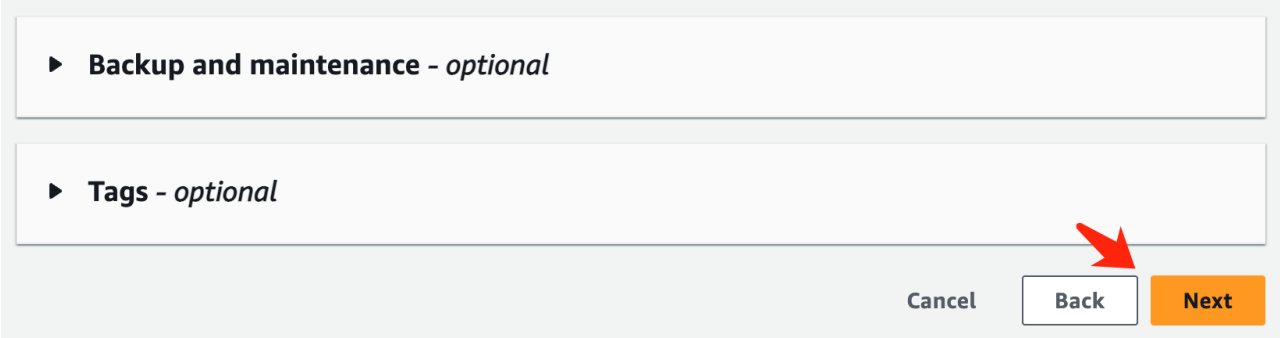
c. 必ず* VPC と*サブネット*を SageMaker Notebook *インスタンスと同じにしてください。



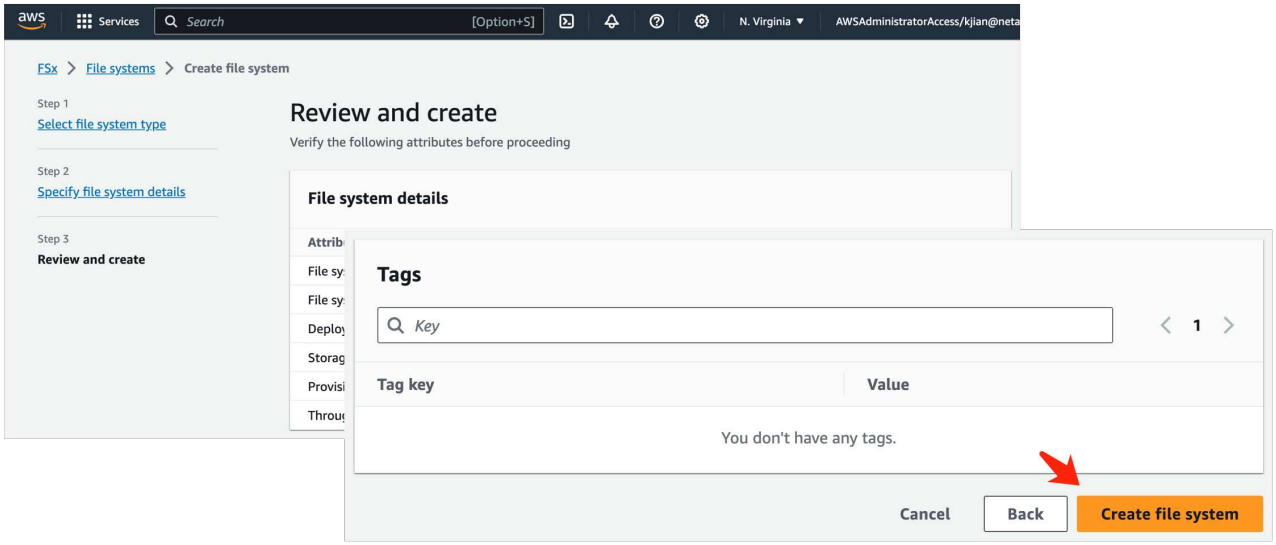
d. Storage Virtual Machine *の名前を入力し、SVM（Storage Virtual Machine）の*パスワードを*指定してください。



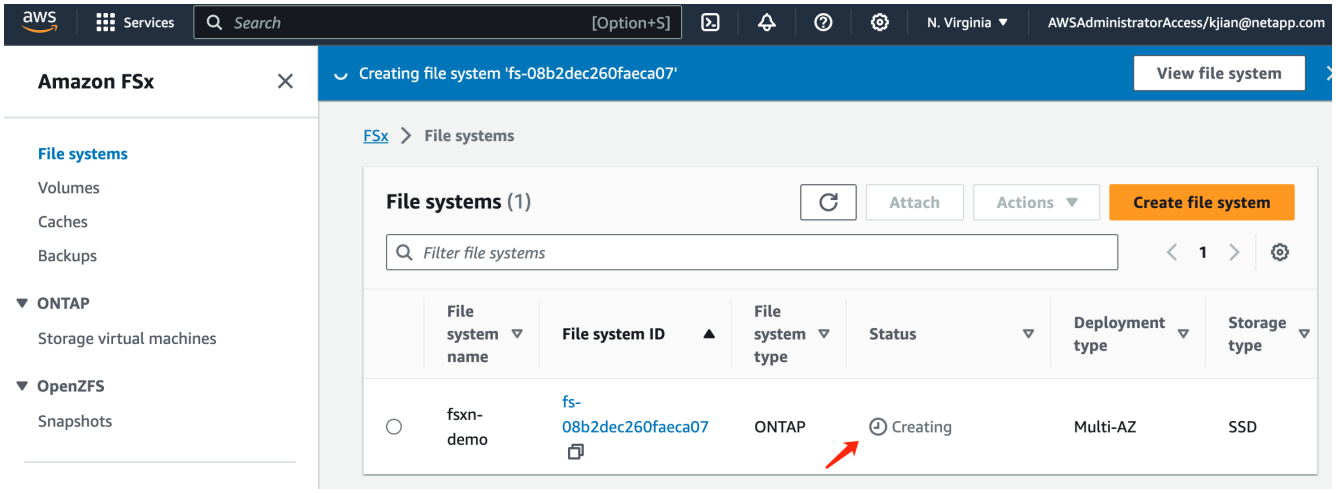
e. [その他のエントリ]はデフォルトのままにして、右下のオレンジ色のボタン*[次へ]*をクリックします。



f. レビューページの右下にあるオレンジ色の*ファイルシステムの作成*ボタンをクリックします。



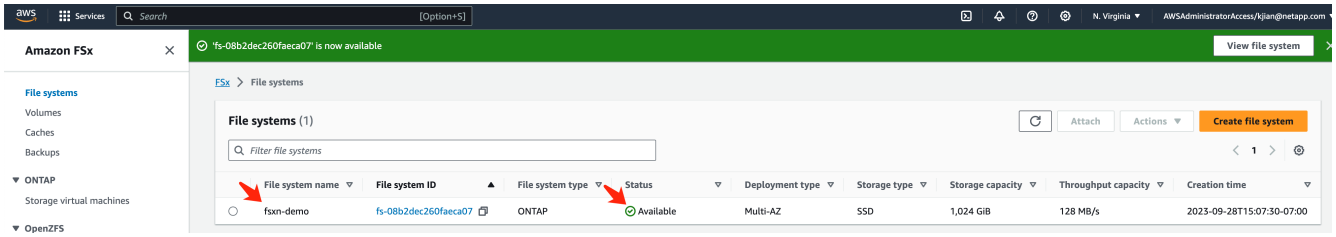
5. FSxファイルシステムのスピンアップには約* 20~40分*かかる場合があります。



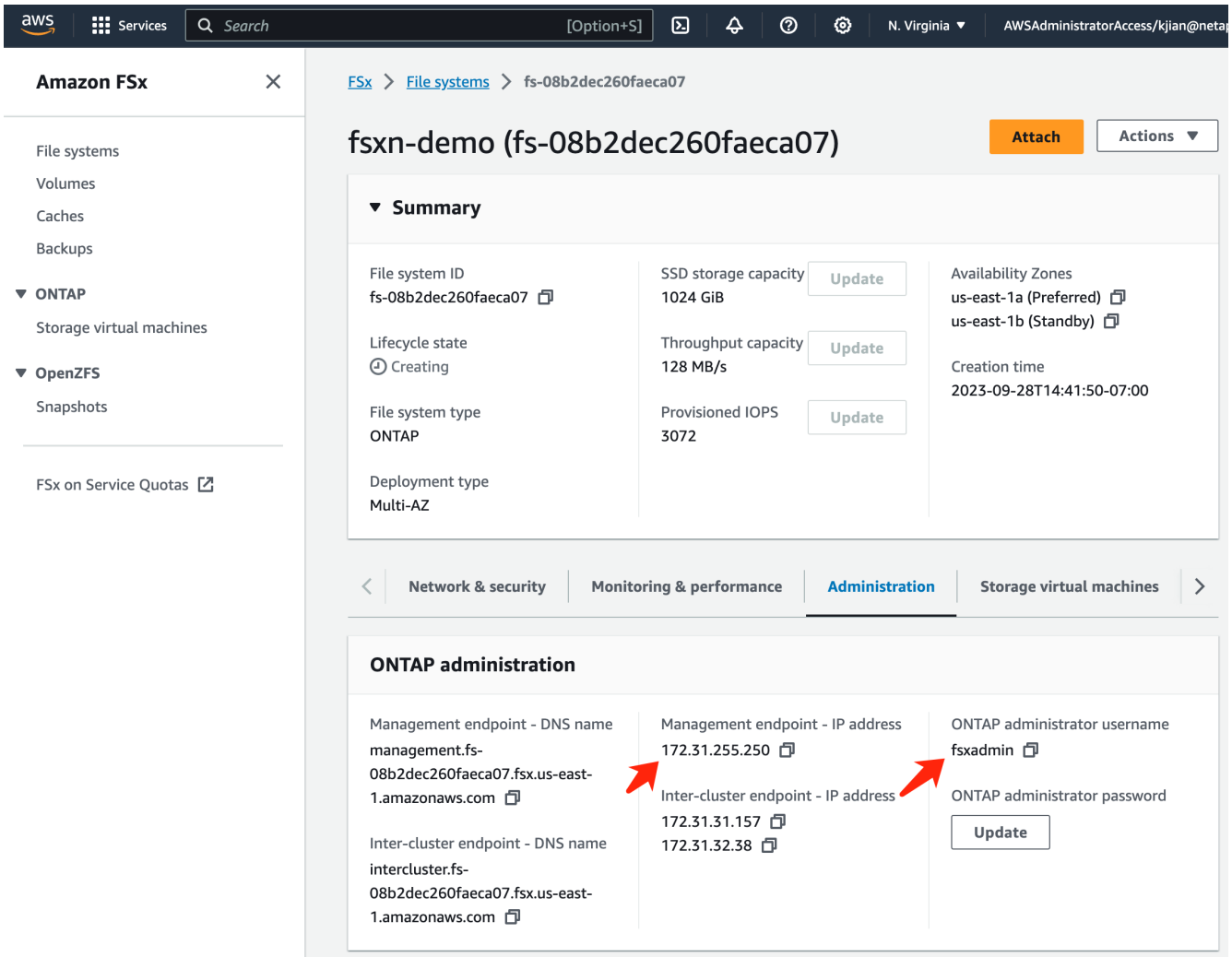
サーバの設定

ONTAPの設定

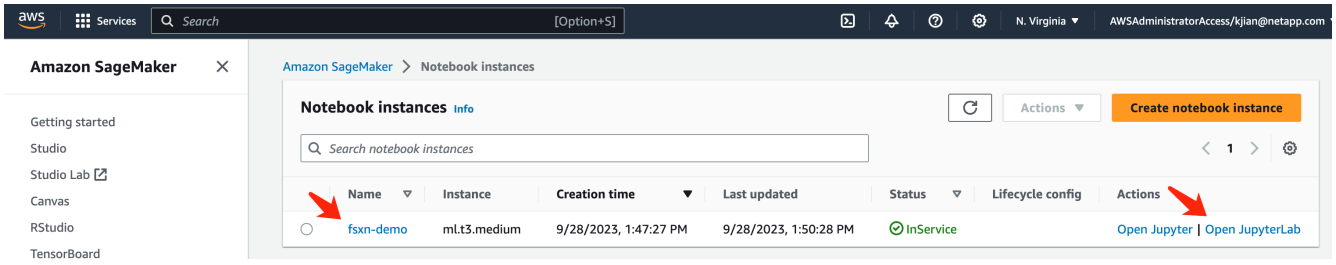
1. 作成したFSxファイルシステムを開きます。ステータスが*利用可能*であることを確認してください。



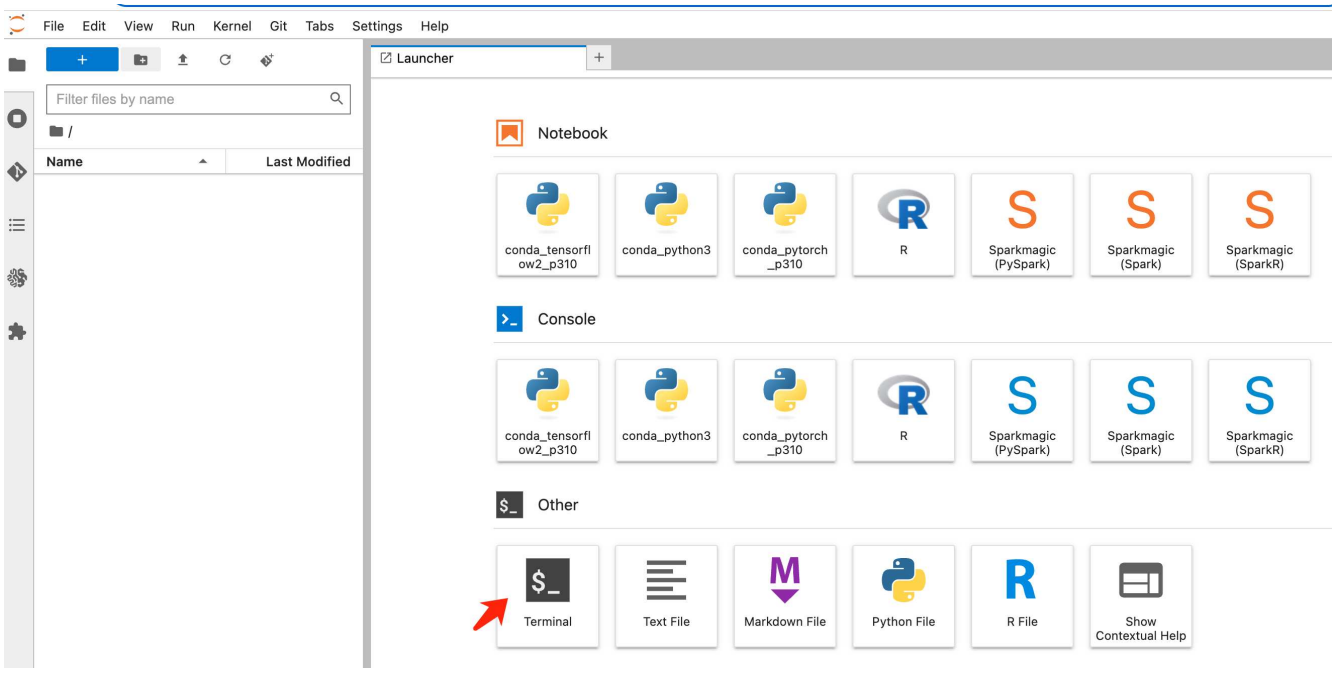
2. [管理]タブを選択し、[管理エンドポイント- IPアドレス]*と[ONTAP管理者のユーザー名]*のままにします。



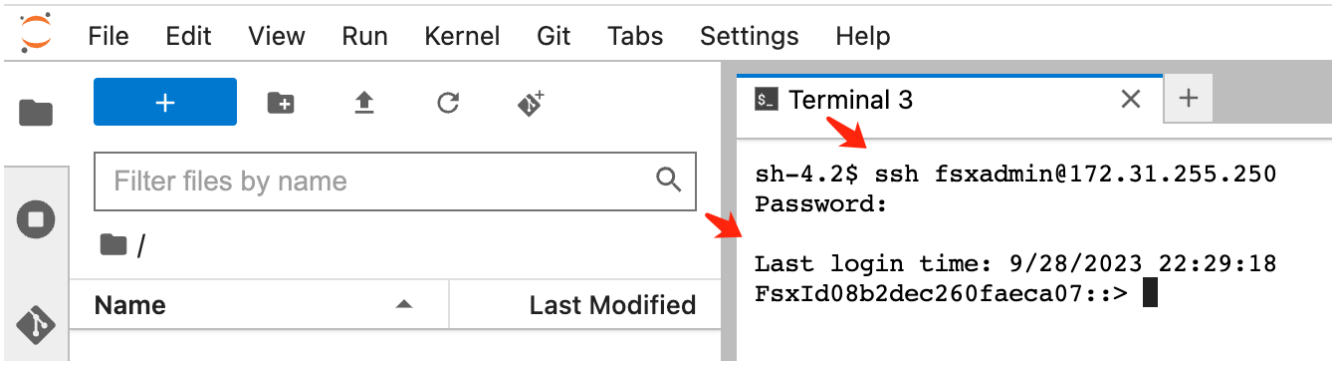
3. 作成した* SageMaker Notebookインスタンス*を開き、*[JupyterLab]*をクリックします。



4. Jupyter Labページで、新しい*ターミナル*を開きます。

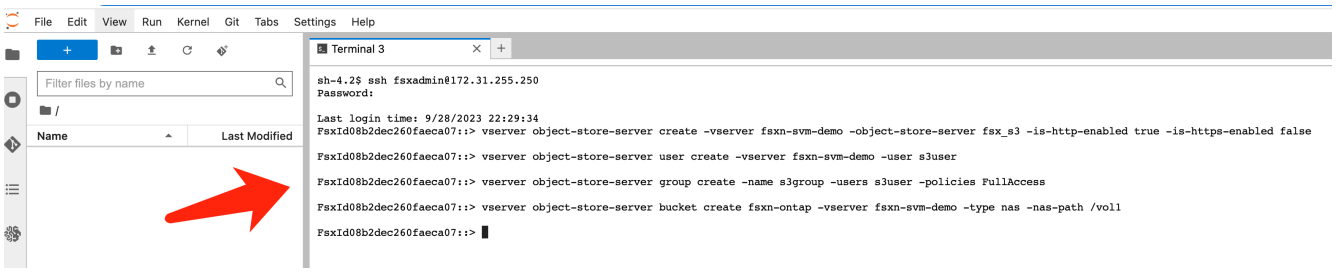


5. sshコマンドssh < admin user name >@< ONTAP server IP >を入力して、FSxN ONTAPファイルシステムにログインします。（ユーザ名とIPアドレスは手順2で取得されます）。
Storage Virtual Machine *の作成時に使用したパスワードを使用してください。



6. 次の順序でコマンドを実行します。
FSxNプライベートS3バケット名*の名前には* fsxn-ontap を使用します。
SVM *引数には Storage Virtual Machine名*を使用してください。

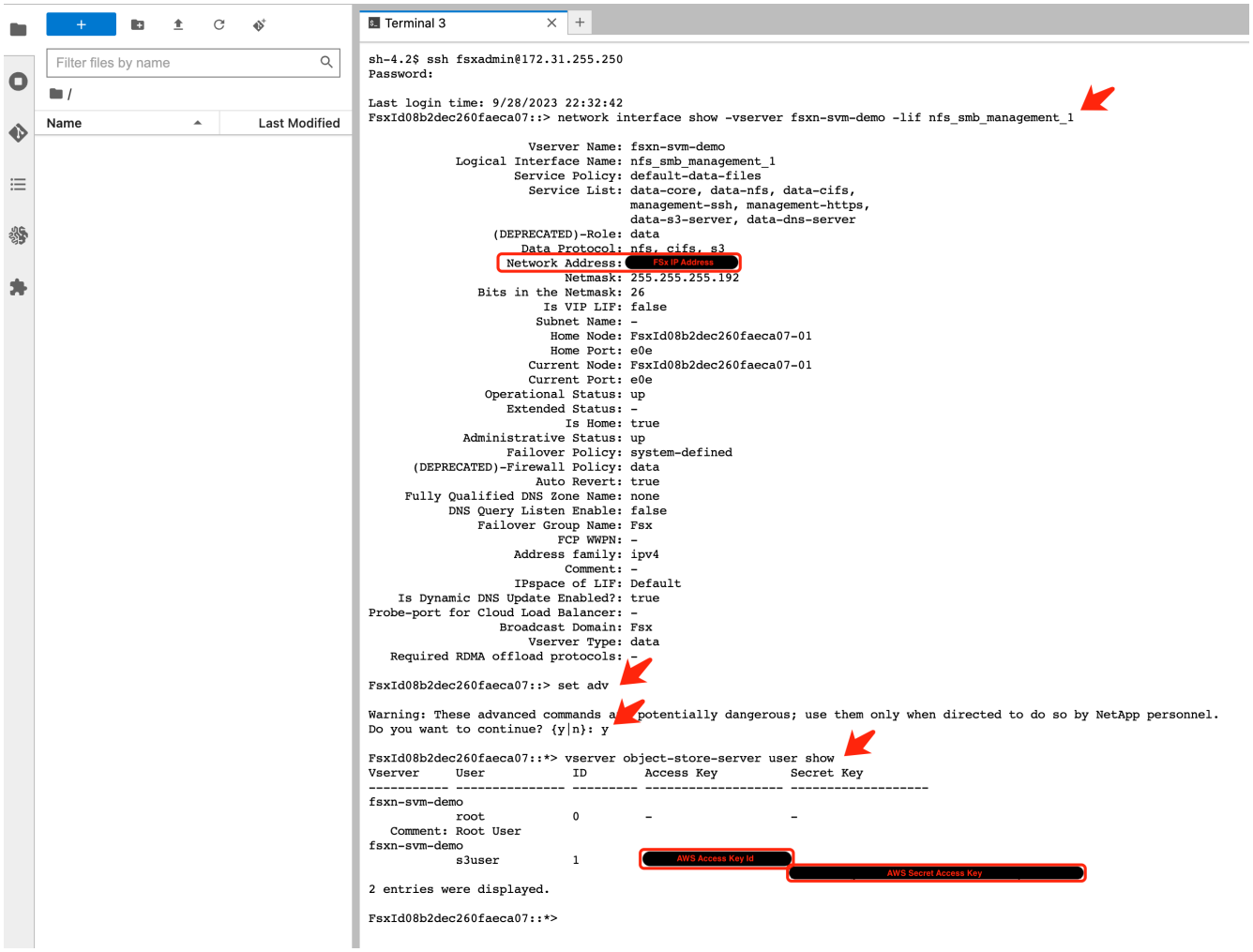
```
vserver object-store-server create -vserver fsxn-svm-demo -object-store
-server fsx_s3 -is-http-enabled true -is-https-enabled false

vserver object-store-server user create -vserver fsxn-svm-demo -user
s3user

vserver object-store-server group create -name s3group -users s3user
-policies FullAccess

vserver object-store-server bucket create fsxn-ontap -vserver fsxn-svm-
demo -type nas -nas-path /vol1
```



7. 次のコマンドを実行して、FSxNプライベートS3のエンドポイントIPとクレデンシャルを取得します。

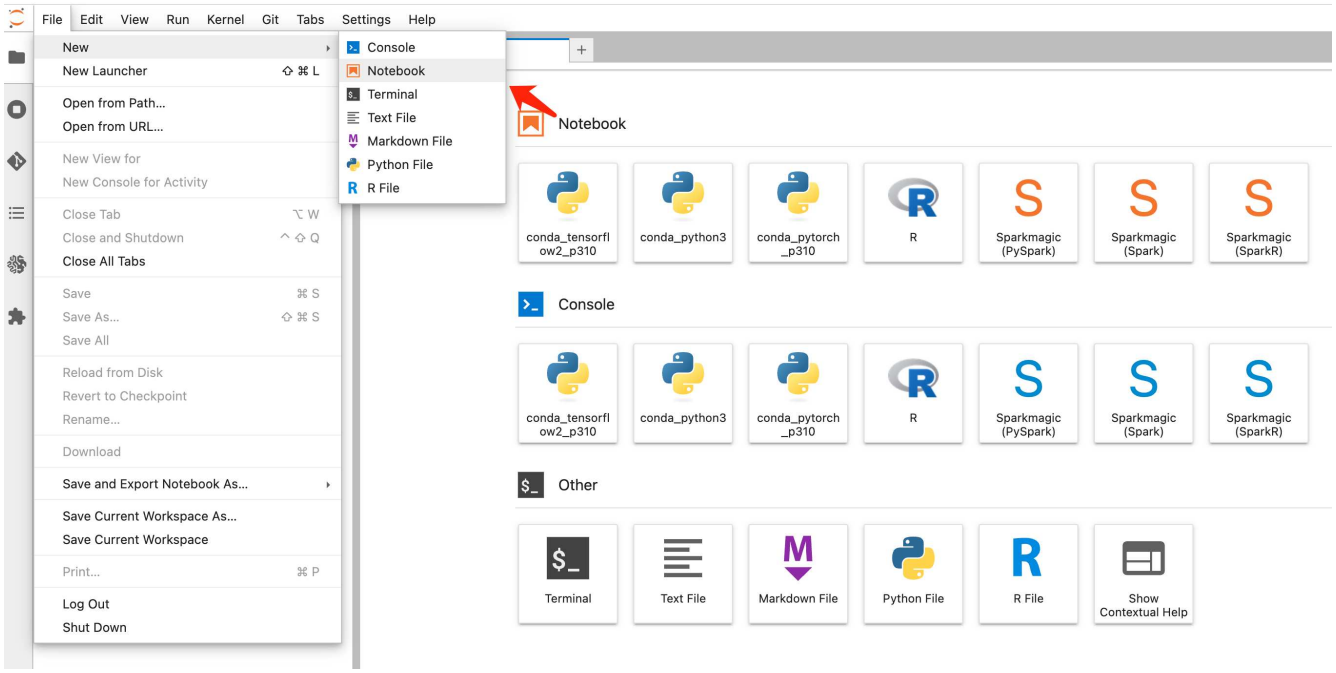
```
network interface show -vserver fsxn-svm-demo -lif nfs_smb_management_1

set adv

vserver object-store-server user show
```

8. あとで使用できるように、エンドポイントのIPとクレデンシャルを保持します。



クライアント設定

1. SageMaker Notebookインスタンスで、新しいJupyterノートブックを作成します。



2. 解決策を回避してFSxNプライベートS3バケットにファイルをアップロードするには、次のコードを使用します。
包括的なコード例については、このノートブックを参照してください。
"fsxn_demo.ipynb"

```
# Setup configurations
# -------- Manual configurations --------
seed: int = 77                                                  # Random
seed
bucket_name: str = 'fsxn-ontap'                                 # The bucket
name in ONTAP
aws_access_key_id = '<Your ONTAP bucket key id>'                # Please get
this credential from ONTAP
aws_secret_access_key = '<Your ONTAP bucket access key>'        # Please get
this credential from ONTAP
fsx_endpoint_ip: str = '<Your FSxN IP address>'                 # Please get
this IP address from FSXN
# -------- Manual configurations --------

# Workaround
## Permission patch
!mkdir -p vol1
!sudo mount -t nfs $fsx_endpoint_ip:/vol1 /home/ec2-user/SageMaker/vol1
!sudo chmod 777 /home/ec2-user/SageMaker/vol1
```

```
## Authentication for FSxN as a Private S3 Bucket
!aws configure set aws_access_key_id $aws_access_key_id
!aws configure set aws_secret_access_key $aws_secret_access_key

## Upload file to the FSxN Private S3 Bucket
%%capture
local_file_path: str = <Your local file path>

!aws s3 cp --endpoint-url http://$fsx_endpoint_ip /home/ec2-user
/SageMaker/$local_file_path  s3://$bucket_name/$local_file_path

# Read data from FSxN Private S3 bucket
## Initialize a s3 resource client
import boto3

# Get session info
region_name = boto3.session.Session().region_name

# Initialize Fsxn S3 bucket object
# --- Start integrating SageMaker with FSXN ---
# This is the only code change we need to incorporate SageMaker with
FSXN
s3_client: boto3.client = boto3.resource(
    's3',
    region_name=region_name,
    aws_access_key_id=aws_access_key_id,
    aws_secret_access_key=aws_secret_access_key,
    use_ssl=False,
    endpoint_url=f'http://{fsx_endpoint_ip}',
    config=boto3.session.Config(
        signature_version='s3v4',
        s3={'addressing_style': 'path'}
    )
)
# --- End integrating SageMaker with FSXN ---

## Read file byte content
bucket = s3_client.Bucket(bucket_name)

binary_data = bucket.Object(data.filename).get()['Body']
```

これで、FSxNとSageMakerインスタンスの統合は終了です。

便利なデバッグチェックリスト

- SageMaker NotebookインスタンスとFSxNファイルシステムが同じVPC内にあることを確認します。

- ONTAPで* set dev コマンドを実行して、特権レベルを dev *に設定することを忘れないでください。

FAQ（**2023**年**9**月**27**日現在）

Q: FSxNにファイルをアップロードするときに、CreateMultipartUpload操作を呼び出したときに「エラーが発生しました**(NotImplemented):**要求した**s3**コマンドが実装されていません」というエラーが表示されるのはなぜですか?

A：プライベートS3バケットとして、FSxNは最大100MBのファイルのアップロードをサポートしています。S3プロトコルを使用する場合、100MBを超えるファイルは100MBのチャンクに分割され、「CreateMultipartUpload」関数が呼び出されます。ただし、FSxNプライベートS3の現在の実装では、この機能はサポートされていません。

Q: FSxNにファイルをアップロードするときに、「* PutObject操作を呼び出したときにエラーが発生しました(AccessDenied):アクセスが拒否されました*」というエラーが表示されるのはなぜですか?

A：SageMaker NotebookインスタンスからFSxNプライベートS3バケットにアクセスするには、AWSクレデンシャルをFSxNクレデンシャルに切り替えます。ただし、インスタンスに書き込み権限を付与するには、バケットをマウントし、「chmod」シェルコマンドを実行して権限を変更する 回避策 解決策 が必要です。

Q: FSxNプライベートS3バケットを他のSageMaker MLサービスと統合するにはどうすればよいですか?

A:残念ながら、SageMakerサービスSDKは、プライベートS3バケットのエンドポイントを指定する方法を提供していません。そのため、FSxN S3はSagemaker Data Wrangler、Sagemaker Clarify、Sagemaker Glue、Sagemaker Athena、Sagemaker AutoMLなどのSageMakerサービスと互換性がありません。 その他。

パート**2 - SageMaker**のモデルトレーニングのデータソースとして**AWS FSx for NetApp ONTAP**（**FSxN**）を活用

作成者：
NetAppシニアデータ&アプリケーションサイエンティスト、Jian Jian（Ken）

はじめに

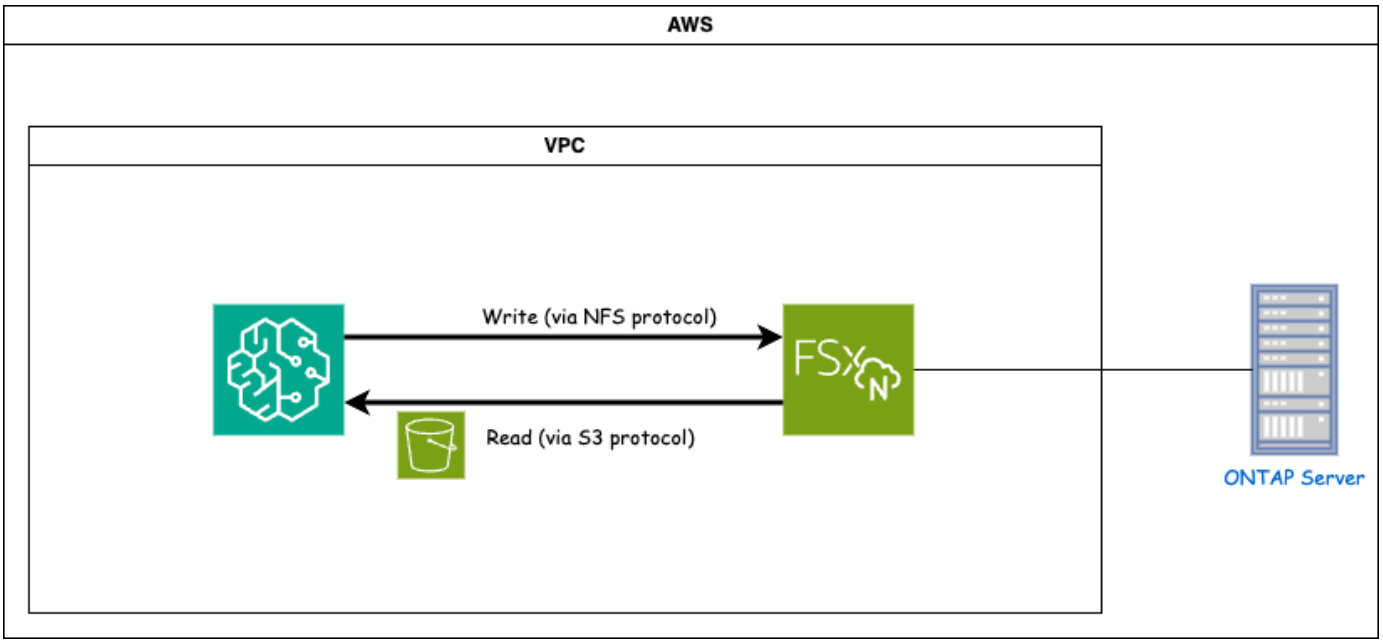
このチュートリアルでは、コンピュータビジョン分類プロジェクトの実践的な例を示し、SageMaker環境内でFSxNをデータソースとして使用するMLモデルを構築するための実践的な経験を提供します。このプロジェクトでは、ディープラーニングフレームワークであるPyTorchを使用して、タイヤの画像に基づいてタイヤの品質を分類することに焦点を当てています。Amazon SageMakerのデータソースとしてFSxNを使用した機械学習モデルの開発に重点を置いています。

FSxNとは

Amazon FSx for NetApp ONTAPは、AWSが提供するフルマネージドストレージ解決策です。ネットアップのONTAPファイルシステムを活用して、信頼性の高いハイパフォーマンスストレージを提供します。NFS、SMB、iSCSIなどのプロトコルをサポートしているため、さまざまなコンピューティングインスタンスやコンテナからシームレスにアクセスできます。このサービスは、卓越したパフォーマンスを提供し、高速かつ効率的なデータ運用を実現するように設計されています。また、高可用性とデータ保持性を実現し、データへのアクセスと保護を維持します。さらに、Amazon FSx for NetApp ONTAPのストレージ容量は拡張性に優れているため、ニーズに合わせて簡単に調整できます。

前提条件

ネットワーク環境



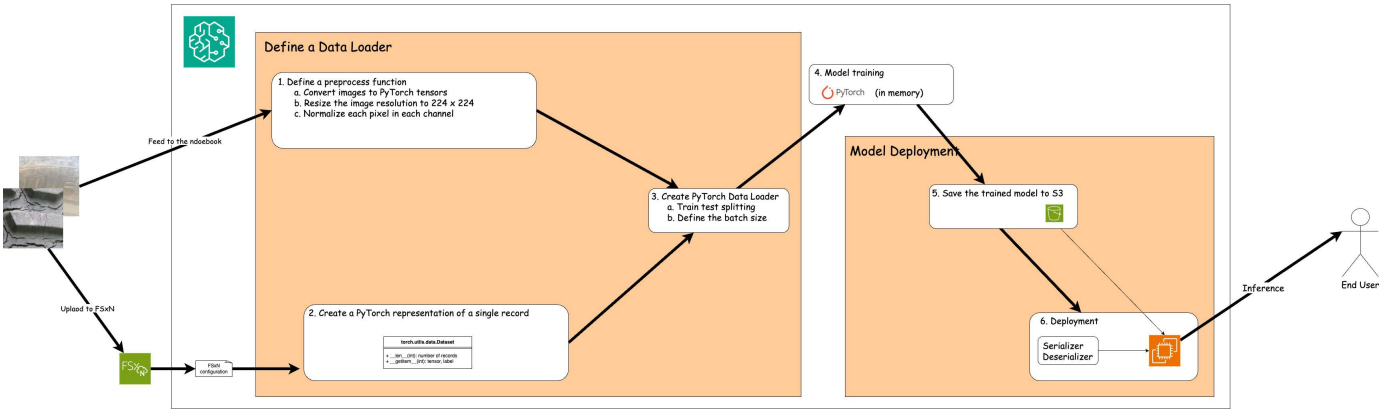
FSxN（Amazon FSx for NetApp ONTAP）は、AWSのストレージサービスです。これには、NetApp ONTAPシステムで実行されているファイルシステムと、接続するAWSで管理されるSystem Virtual Machine（SVM）が含まれます。次の図では、AWSで管理されるNetApp ONTAPサーバがVPCの外部に配置されています。SVMはSageMakerとNetApp ONTAPシステムの仲介役として機能し、SageMakerからの処理要求を受け取り、基盤となるストレージに転送します。FSxNにアクセスするには、SageMakerをFSxN展開と同じVPC内に配置する必要があります。この構成により、SageMakerとFSxN間の通信とデータアクセスが保証されます。

データアクセス

実際のシナリオでは、データサイエンティストは通常、FSxNに保存されている既存のデータを利用して機械学習モデルを構築します。ただし、デモ目的では、FSxNファイルシステムは作成後に最初は空であるため、トレーニングデータを手動でアップロードする必要があります。これは、FSxNをボリュームとしてSageMakerにマウントすることで実現できます。ファイルシステムが正常にマウントされたら、データセットをマウントされた場所にアップロードして、SageMaker環境内でモデルをトレーニングするためにアクセスできるようにすることができます。このアプローチでは、SageMakerと連携してモデルの開発とトレーニングを行いながら、FSxNのストレージ容量と機能を活用できます。

データ読み取りプロセスでは、FSxNをプライベートS3バケットとして設定します。詳細な設定手順については、を参照してください。 "パート1 - AWS FSx for NetApp ONTAP（FSxN）をプライベートS3バケットとしてAWS SageMakerに統合する"

統合の概要



FSxNのトレーニングデータを使用してSageMakerでディープラーニングモデルを構築するワークフローは、データローダーの定義、モデルのトレーニング、デプロイの3つの主なステップに要約できます。大まかに言えば、これらのステップはMLOpsパイプラインの基盤を形成します。ただし、各ステップには、包括的な実装のためのいくつかの詳細なサブステップが含まれています。これらのサブステップには、データの前処理、データセットの分割、モデルの構成、ハイパーパラメータの調整、モデルの評価など、さまざまなタスクが含まれます。 モデルの導入を支援します。これらの手順により、SageMaker環境内でFSxNからのトレーニングデータを使用してディープラーニングモデルを構築し、展開するための徹底的で効果的なプロセスが保証されます。

ステップバイステップの統合

データローダ

PyTorchディープラーニングネットワークをデータでトレーニングするために、データのフィードを容易にするためのデータローダーが作成されます。データローダーは、バッチサイズを定義するだけでなく、バッチ内の各レコードを読み取って前処理するための手順も決定します。データローダーを構成することで、データの処理をバッチで処理し、ディープラーニングネットワークのトレーニングを可能にします。

データローダーは3つの部分で構成されています。

前処理機能

```
from torchvision import transforms

preprocess = transforms.Compose([
    transforms.ToTensor(),
    transforms.Resize((224,224)),
    transforms.Normalize(
        mean=[0.485, 0.456, 0.406],
        std=[0.229, 0.224, 0.225]
    )
])
```

上記のコードスニペットは、**torchvision.transforms***モジュールを使用した画像前処理変換の定義を示しています。この**turtorial**では、一連の変換を適用するためにプリプロセスオブジェクトが作成されます。まず、***ToTensor()***変換は画像をテンソル表現に変換する。その後、***Resize(224,224***変換により、画像のサイズが**224x224**ピクセルの固定サイズに変更されます。最後に、 Normalize()*変換は、平均を減算し、各チャ

ンネルに沿った標準偏差で割ることによってテンソル値を正規化します。正規化に使用される平均値と標準偏差値は、事前にトレーニングされたニューラルネットワークモデルで一般的に使用されます。全体的に、このコードは、画像データをテンソルに変換し、サイズを変更し、ピクセル値を正規化することで、事前にトレーニングされたモデルにさらに処理または入力できるように準備します。

PyTorchデータセットクラス

```
import torch
from io import BytesIO
from PIL import Image


class FSxNImageDataset(torch.utils.data.Dataset):
    def __init__(self, bucket, prefix='', preprocess=None):
        self.image_keys = [
            s3_obj.key
            for s3_obj in list(bucket.objects.filter(Prefix=prefix).all())
        ]
        self.preprocess = preprocess

    def __len__(self):
        return len(self.image_keys)

    def __getitem__(self, index):
        key = self.image_keys[index]
        response = bucket.Object(key)

        label = 1 if key[13:].startswith('defective') else 0

        image_bytes = response.get()['Body'].read()
        image = Image.open(BytesIO(image_bytes))
        if image.mode == 'L':
            image = image.convert('RGB')

        if self.preprocess is not None:
            image = self.preprocess(image)
        return image, label
```

このクラスは、データセット内のレコードの総数を取得する機能を提供し、各レコードのデータを読み取る方法を定義します。***getItem****関数内で、コードは**boto3 S3**バケットオブジェクトを使用して**FSxN**からバイナリデータを取得します。**FSxN**からデータにアクセスするためのコードスタイルは、**Amazon S3**からデータを読み取るのと似ています。以降の説明では、プライベート**S3**オブジェクト Bucket *の作成プロセスについて詳しく説明します。

プライベート**S3**リポジトリとしての**FSxN**

```
seed = 77                                                    # Random seed
bucket_name = '<Your ONTAP bucket name>'                     # The bucket
name in ONTAP
aws_access_key_id = '<Your ONTAP bucket key id>'             # Please get
this credential from ONTAP
aws_secret_access_key = '<Your ONTAP bucket access key>'     # Please get
this credential from ONTAP
fsx_endpoint_ip = '<Your FSxN IP address>'                   # Please get
this IP address from FSXN
```

```
import boto3

# Get session info
region_name = boto3.session.Session().region_name

# Initialize Fsxn S3 bucket object
# --- Start integrating SageMaker with FSXN ---
# This is the only code change we need to incorporate SageMaker with FSXN
s3_client: boto3.client = boto3.resource(
    's3',
    region_name=region_name,
    aws_access_key_id=aws_access_key_id,
    aws_secret_access_key=aws_secret_access_key,
    use_ssl=False,
    endpoint_url=f'http://{fsx_endpoint_ip}',
    config=boto3.session.Config(
        signature_version='s3v4',
        s3={'addressing_style': 'path'}
    )
)
# s3_client = boto3.resource('s3')
bucket = s3_client.Bucket(bucket_name)
# --- End integrating SageMaker with FSXN ---
```

SageMakerでFSxNからデータを読み取るために、S3プロトコルを使用してFSxNストレージを指すハンドラが作成されます。これにより、FSxNをプライベートS3バケットとして扱うことができます。ハンドラの設定では、FSxN SVMのIPアドレス、バケット名、および必要なクレデンシャルを指定します。これらの設定項目の入手方法については、次のWebサイトにあるドキュメントを参照してください。 "パート1 - AWS FSx for NetApp ONTAP（FSxN）をプライベートS3バケットとしてAWS SageMakerに統合する"。

前述の例では、Bucketオブジェクトを使用してPyTorchデータセットオブジェクトをインスタンス化しています。データセットオブジェクトについては、次のセクションで詳しく説明します。

PyTorchデータローダ

```
from torch.utils.data import DataLoader
torch.manual_seed(seed)

# 1. Hyperparameters
batch_size = 64

# 2. Preparing for the dataset
dataset = FSxNImageDataset(bucket, 'dataset/tyre', preprocess=preprocess)

train, test = torch.utils.data.random_split(dataset, [1500, 356])

data_loader = DataLoader(dataset, batch_size=batch_size, shuffle=True)
```

この例では、64のバッチサイズが指定されています。これは、各バッチに64レコードが含まれることを示しています。PyTorch * Dataset *クラス、前処理関数、およびトレーニングバッチサイズを組み合わせることで、トレーニング用のデータローダーを取得します。このデータローダーは、トレーニングフェーズ中にデータセットをバッチで反復処理するプロセスを容易にします。

モデルトレーニング

```
from torch import nn


class TyreQualityClassifier(nn.Module):
    def __init__(self):
        super().__init__()
        self.model = nn.Sequential(
            nn.Conv2d(3,32,(3,3)),
            nn.ReLU(),
            nn.Conv2d(32,32,(3,3)),
            nn.ReLU(),
            nn.Conv2d(32,64,(3,3)),
            nn.ReLU(),
            nn.Flatten(),
            nn.Linear(64*(224-6)*(224-6),2)
        )
    def forward(self, x):
        return self.model(x)
```

```
import datetime

num_epochs = 2
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

model = TyreQualityClassifier()
fn_loss = torch.nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=1e-3)


model.to(device)
for epoch in range(num_epochs):
    for idx, (X, y) in enumerate(data_loader):
        X = X.to(device)
        y = y.to(device)

        y_hat = model(X)

        loss = fn_loss(y_hat, y)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        current_time = datetime.datetime.now().strftime("%Y-%m-%d
%H:%M:%S")
        print(f"Current Time: {current_time} - Epoch [{epoch+1}/
{num_epochs}]- Batch [{idx + 1}] - Loss: {loss}", end='\r')
```

このコードは標準のPyTorchトレーニングプロセスを実装しています。これは、畳み込み層と線形層を使用してタイヤの品質を分類する*TireQualityClassifier*と呼ばれるニューラルネットワークモデルを定義します。トレーニングループはデータバッチを繰り返し、損失を計算し、バックプロパゲーションと最適化を使用してモデルのパラメータを更新します。さらに、現在の時刻、エポック、バッチ、および損失を監視するために印刷します。

モデルの導入

導入

```
import io
import os
import tarfile
import sagemaker

# 1. Save the PyTorch model to memory
buffer_model = io.BytesIO()
traced_model = torch.jit.script(model)
torch.jit.save(traced_model, buffer_model)

# 2. Upload to AWS S3
sagemaker_session = sagemaker.Session()
bucket_name_default = sagemaker_session.default_bucket()
model_name = f'tyre_quality_classifier.pth'

# 2.1. Zip PyTorch model into tar.gz file
buffer_zip = io.BytesIO()
with tarfile.open(fileobj=buffer_zip, mode="w:gz") as tar:
    # Add PyTorch pt file
    file_name = os.path.basename(model_name)
    file_name_with_extension = os.path.split(file_name)[-1]
    tarinfo = tarfile.TarInfo(file_name_with_extension)
    tarinfo.size = len(buffer_model.getbuffer())
    buffer_model.seek(0)
    tar.addfile(tarinfo, buffer_model)

# 2.2. Upload the tar.gz file to S3 bucket
buffer_zip.seek(0)
boto3.resource('s3') \
    .Bucket(bucket_name_default) \
    .Object(f'pytorch/{model_name}.tar.gz') \
    .put(Body=buffer_zip.getvalue())
```

このコードはPyTorchモデルを* Amazon S3 に保存します。これは、**SageMaker**が展開するためにモデルを**S3**に格納する必要があるためです。モデルを Amazon S3 *にアップロードすることで、SageMakerからアクセスできるようになり、デプロイされたモデルでのデプロイと推論が可能になります。

```
import time
from sagemaker.pytorch import PyTorchModel
from sagemaker.predictor import Predictor
from sagemaker.serializers import IdentitySerializer
from sagemaker.deserializers import JSONDeserializer


class TyreQualitySerializer(IdentitySerializer):
```

```
    CONTENT_TYPE = 'application/x-torch'

    def serialize(self, data):
        transformed_image = preprocess(data)
        tensor_image = torch.Tensor(transformed_image)

        serialized_data = io.BytesIO()
        torch.save(tensor_image, serialized_data)
        serialized_data.seek(0)
        serialized_data = serialized_data.read()

        return serialized_data


class TyreQualityPredictor(Predictor):
    def __init__(self, endpoint_name, sagemaker_session):
        super().__init__(
            endpoint_name,
            sagemaker_session=sagemaker_session,
            serializer=TyreQualitySerializer(),
            deserializer=JSONDeserializer(),
        )

sagemaker_model = PyTorchModel(
    model_data=f's3://{bucket_name_default}/pytorch/{model_name}.tar.gz',
    role=sagemaker.get_execution_role(),
    framework_version='2.0.1',
    py_version='py310',
    predictor_cls=TyreQualityPredictor,
    entry_point='inference.py',
    source_dir='code',
)

timestamp = int(time.time())
pytorch_endpoint_name = '{}-{}-{}'.format('tyre-quality-classifier', 'pt',
timestamp)
sagemaker_predictor = sagemaker_model.deploy(
    initial_instance_count=1,
    instance_type='ml.p3.2xlarge',
    endpoint_name=pytorch_endpoint_name
)
```

このコードは、SageMakerへのPyTorchモデルのデプロイを容易にします。これは、入力データをPyTorchテンソルとして前処理してシリアライズするカスタムシリアライザ*TireQualitySerializer*を定義します。**TireQualityPredictor***クラスは、定義されたシリアライザと***JSONDeserializer***を利用するカスタムプレディクタです。コードはまた、モデルの**S3**の場所、**IAM**の役割、フレームワークのバージョン、推論のエントリポイントを指定する PyTorchModel *オブジェクトを作成します。コードはタイムスタンプを生成し、モデ

ルとタイムスタンプに基づいてエンドポイント名を構築します。最後に、インスタンス数、インスタンスタイプ、生成されたエンドポイント名を指定して、deployメソッドを使用してモデルをデプロイします。これにより、PyTorchモデルをデプロイし、SageMakerで推論できるようになります。

推論

```
image_object = list(bucket.objects.filter('dataset/tyre'))[0].get()
image_bytes = image_object['Body'].read()

with Image.open(with Image.open(BytesIO(image_bytes)) as image:
    predicted_classes = sagemaker_predictor.predict(image)

    print(predicted_classes)
```

次の例では、導入したエンドポイントを使用して推論を実行しています。

パート**3 -**簡易化された**MLOps**パイプラインの構築（**CI/CT/CD**）

作成者：NetAppシニアデータ&アプリケーションサイエンティスト、Jian Jian（Ken）

はじめに

このチュートリアルでは、さまざまなAWSサービスを活用して、継続的統合（CI）、継続的トレーニング（CT）、継続的導入（CD）を含むシンプルなMLOpsパイプラインを構築する方法を学習します。従来のDevOpsパイプラインとは異なり、MLOpsでは運用サイクルを完了するために追加の考慮事項が必要です。このチュートリアルに従うことで、CTをMLOpsループに組み込む方法についての洞察を得ることができ、モデルの継続的なトレーニングと推論のためのシームレスな導入が可能になります。このチュートリアルでは、AWSサービスを利用してこのエンドツーエンドのMLOpsパイプラインを確立するプロセスをガイドします。

マニフェスト

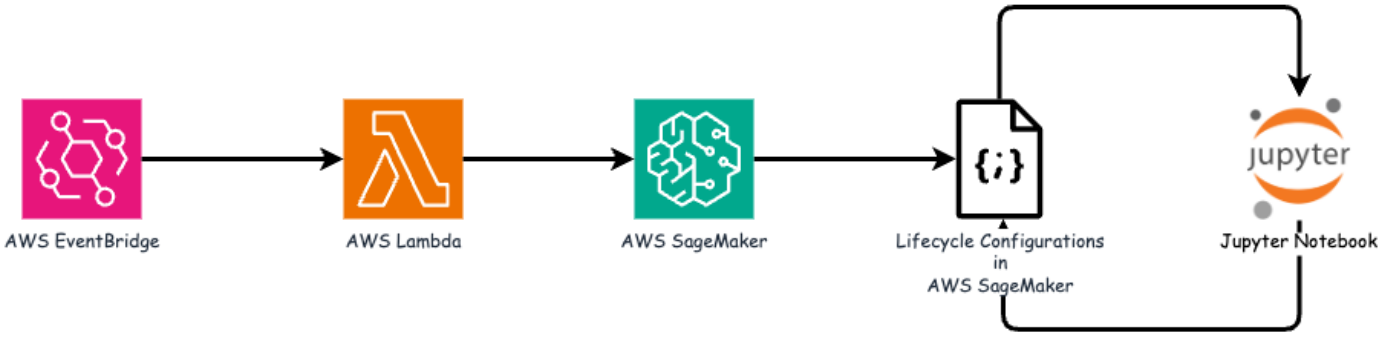
機能性	名前	コメント（ **Comment** ）
データストレージ	AWS FSxN	を参照してください "パート1 - AWS FSx for NetApp ONTAP（FSxN）をプライベートS3バケットとしてAWS SageMakerに統合する"。
データサイエンスIDE	AWS SageMaker	このチュートリアルは、で紹介されているJupyterノートブックをベースにしています。 "パート2 - SageMakerのモデルトレーニングのデータソースとしてAWS FSx for NetApp ONTAP（FSxN）を活用"。
MLOpsパイプラインをトリガーする機能	AWS Lambda関数	-
cronジョブトリガー	AWSイベントブリッジ	-

機能性	名前	コメント（ **Comment** ）
ディープラーニングフレームワーク	PyTorch	-
AWS Python SDK	ボット3	-
プログラミング言語	Python	v3.10

前提条件

- 事前設定されたFSxNファイルシステム。このチュートリアルでは、FSxNに保存されているデータをトレーニングプロセスに利用します。
- 前述のFSxNファイルシステムと同じVPCを共有するように構成された* SageMaker Notebookインスタンス*。
- * AWS Lambda関数*をトリガーする前に、* SageMaker Notebookインスタンス*が*停止*ステータスになっていることを確認してください。
- ディープニューラルネットワークの計算に必要なGPUアクセラレーションを利用するには、* ml.g4dn.xlarge *インスタンスタイプが必要です。

アーキテクチャ



このMLOpsパイプラインは、cronジョブを利用してサーバレス関数をトリガーし、ライフサイクルコールバック関数に登録されたAWSサービスを実行する実用的な実装です。AWS EventBridge は**cron**ジョブとして機能します。モデルの再トレーニングと再デプロイを担当する AWS Lambda関数*を定期的に呼び出します。このプロセスでは、* AWS SageMaker Notebook *インスタンスをスピンアップして必要なタスクを実行します。

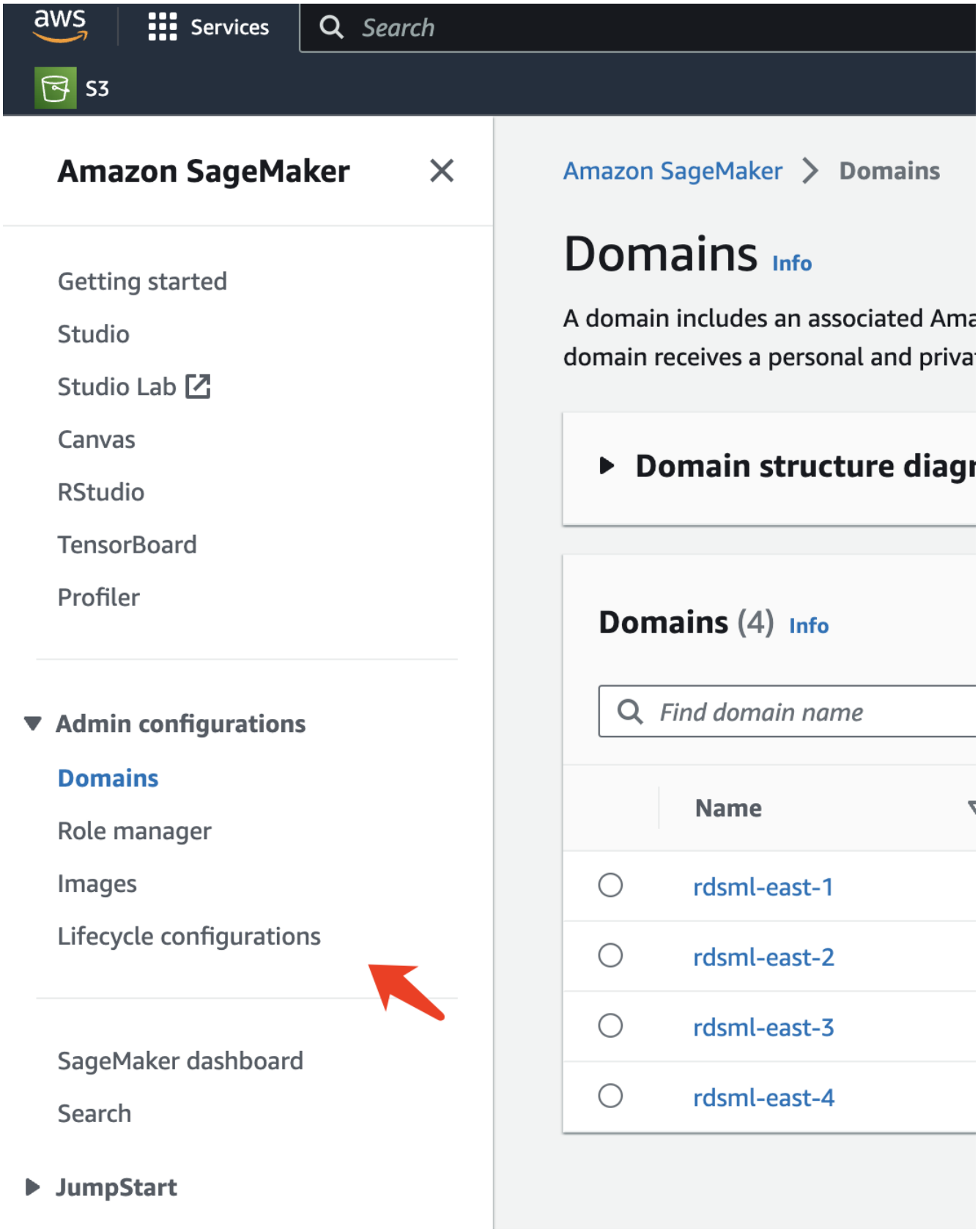
ステップバイステップ構成

ライフサイクル設定

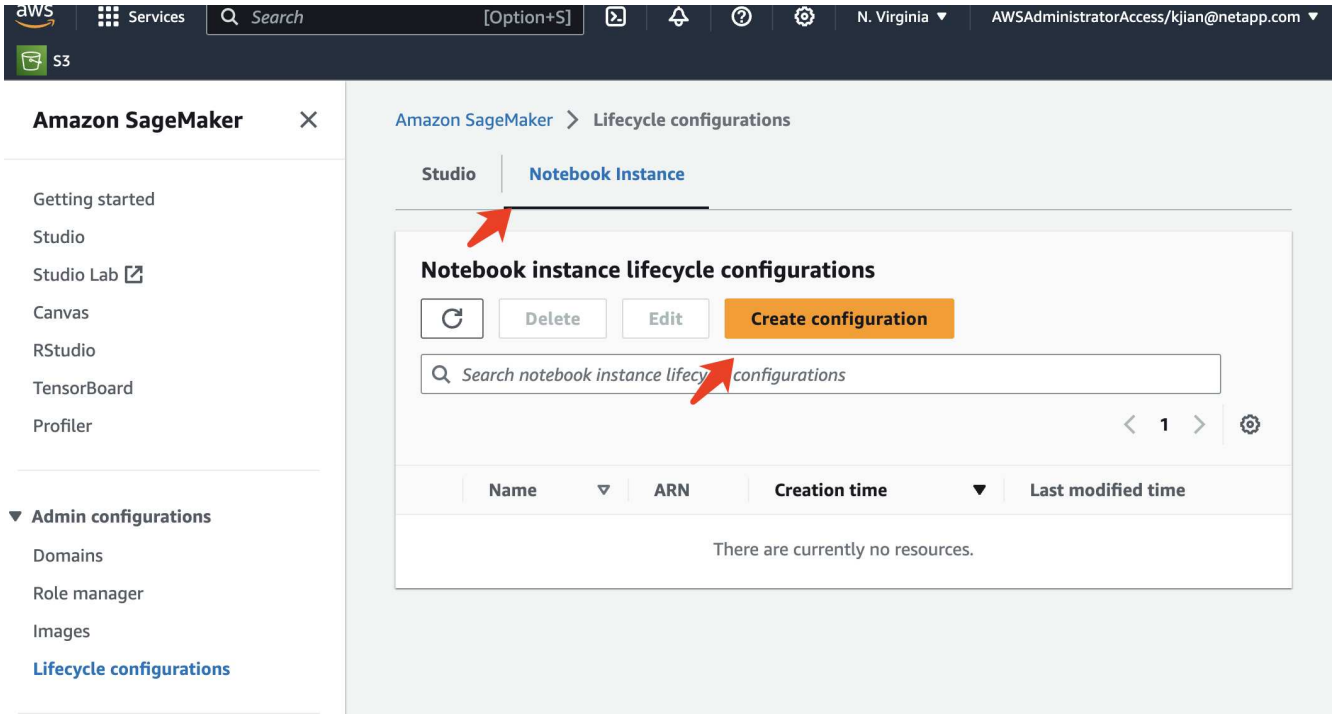
AWS SageMaker Notebookインスタンスのライフサイクルコールバック関数を設定するには、ライフサイクル設定*を使用します。このサービスでは、ノートブックインスタンスをスピンアップするときに実行する必要のあるアクションを定義できます。具体的には、 Lifecycle configurations *内にシェルスクリプトを実装して、トレーニングおよび展開プロセスが完了するとノートブックインスタンスを自動的にシャットダウンすることができます。MLOpsではコストが重要な考慮事項の1つであるため、これは必須の設定です。

重要なのは、*ライフサイクル構成*の構成は事前に設定する必要があることです。したがって、他のMLOpsパイプラインのセットアップに進む前に、この側面の設定を優先することをお勧めします。

1. ライフサイクル構成を設定するには、* Sagemaker パネルを開き、 Admin configurations セクションの Lifecycle configurations *に移動します。



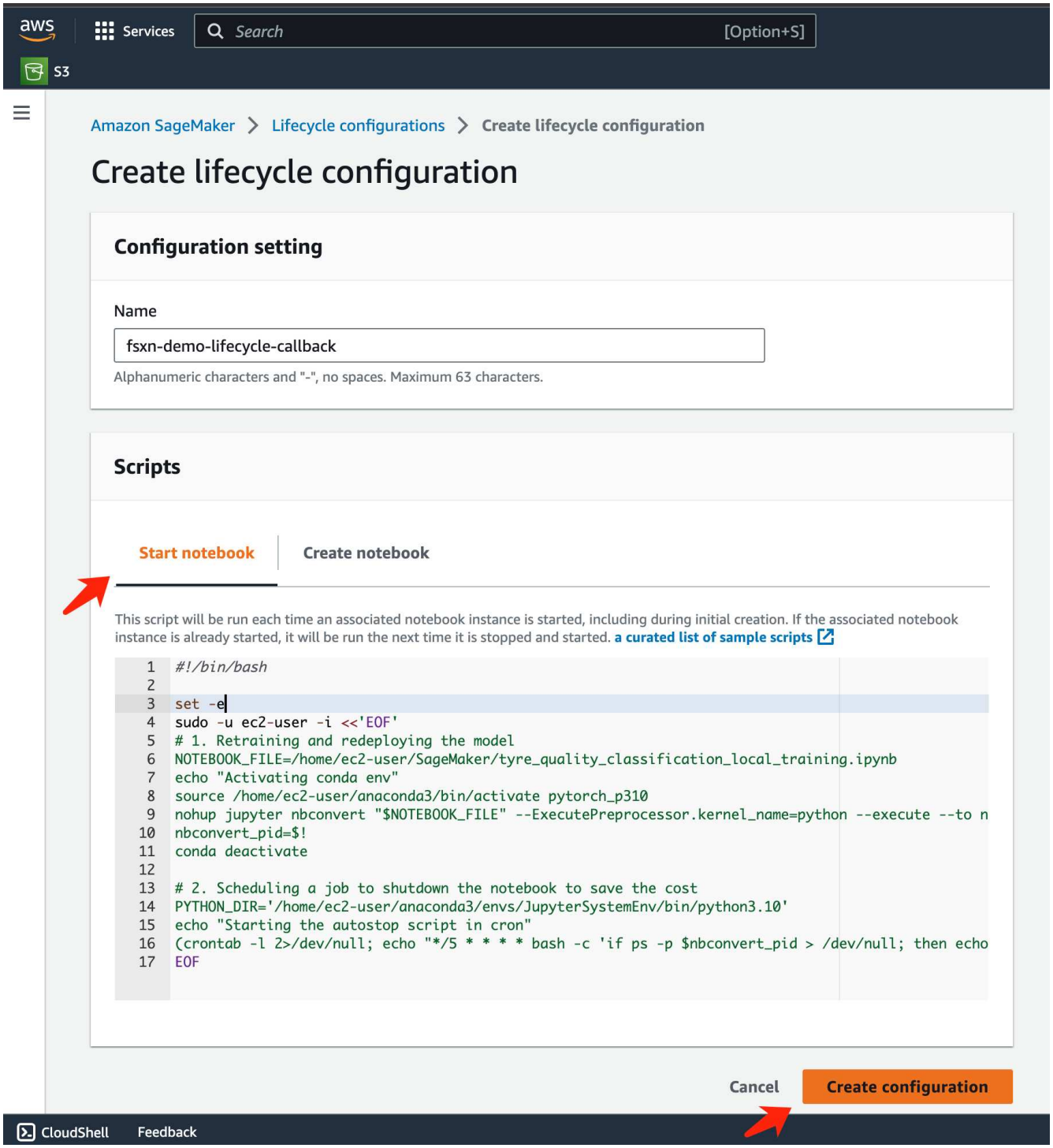
2. [Notebook Instance]タブを選択し、[Create configuration]ボタンをクリックします。



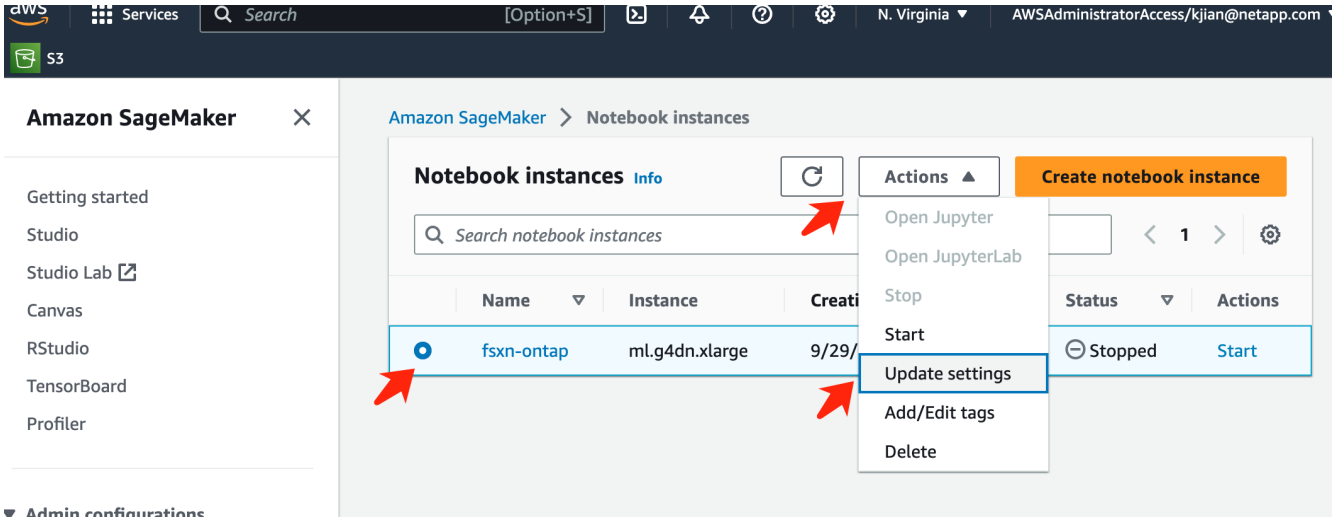
3. 次のコードを入力領域に貼り付けます。

```
#!/bin/bash

set -e
sudo -u ec2-user -i <<'EOF'
# 1. Retraining and redeploying the model
NOTEBOOK_FILE=/home/ec2-
user/SageMaker/tyre_quality_classification_local_training.ipynb
echo "Activating conda env"
source /home/ec2-user/anaconda3/bin/activate pytorch_p310
nohup jupyter nbconvert "$NOTEBOOK_FILE"
--ExecutePreprocessor.kernel_name=python --execute --to notebook &
nbconvert_pid=$!
conda deactivate

# 2. Scheduling a job to shutdown the notebook to save the cost
PYTHON_DIR='/home/ec2-
user/anaconda3/envs/JupyterSystemEnv/bin/python3.10'
echo "Starting the autostop script in cron"
(crontab -l 2>/dev/null; echo "*/5 * * * * bash -c 'if ps -p
$nbconvert_pid > /dev/null; then echo \"Notebook is still running.\" >>
/var/log/jupyter.log; else echo \"Notebook execution completed.\" >>
/var/log/jupyter.log; $PYTHON_DIR -c \"import boto3;boto3.client(
\'sagemaker\').stop_notebook_instance(NotebookInstanceName=get_notebook_
name())\" >> /var/log/jupyter.log; fi'") | crontab -
EOF
```

4. このスクリプトは、推論のためのモデルの再トレーニングと再配置を処理するJupyter Notebookを実行します。実行が完了すると、ノートブックは5分以内に自動的にシャットダウンされます。問題点とコード実装の詳細については、を参照してください。 "パート2 - SageMakerのモデルトレーニングのデータソースとしてAWS FSx for NetApp ONTAP（FSxN）を活用"。

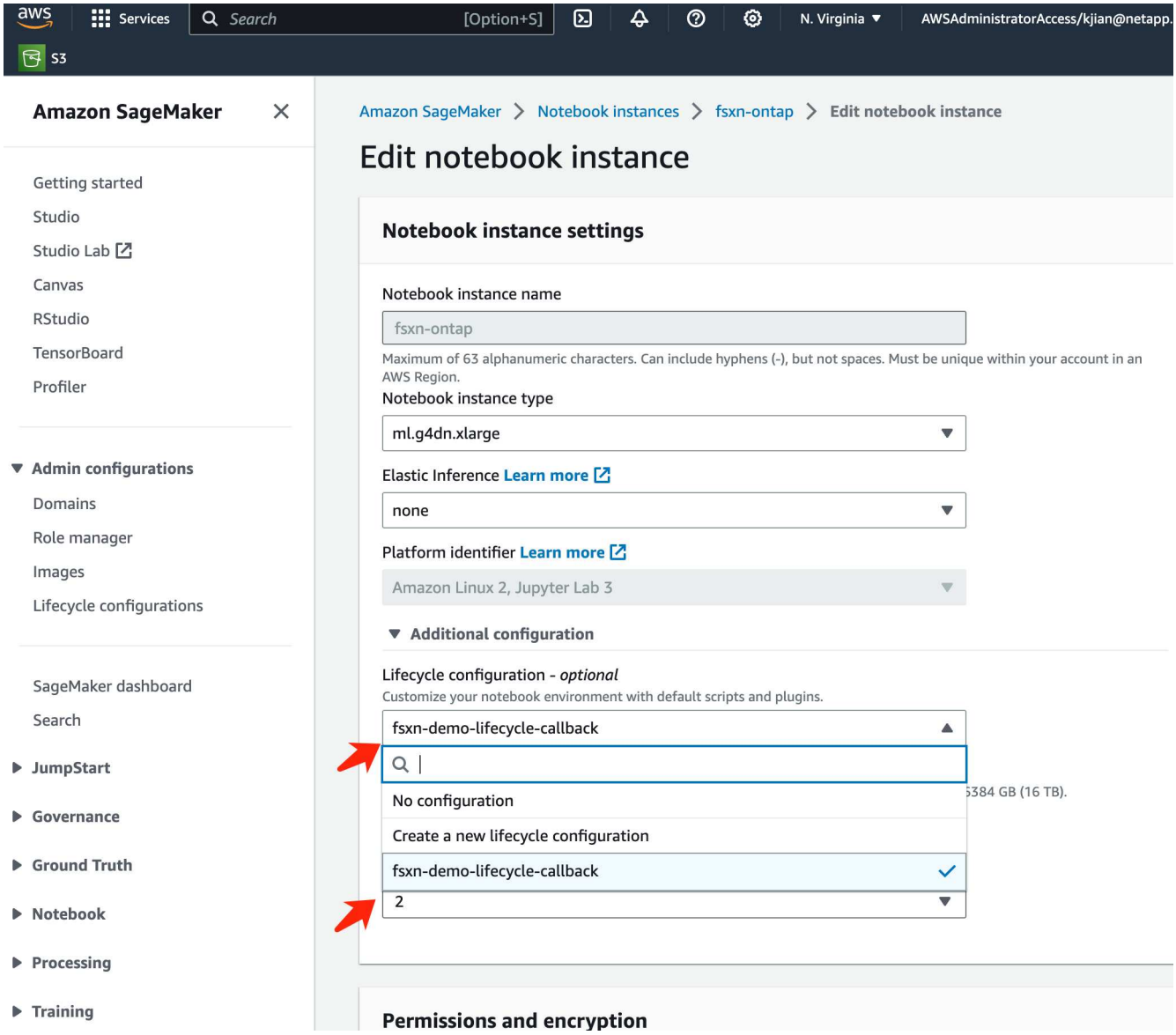


5. 作成後、Notebook Instances（ノートブックインスタンス）に移動してターゲットインスタンスを選択し、Actions（アクション）ドロップダウンの* Update settings（設定の更新）*をクリックします。



ドロップダウン"]

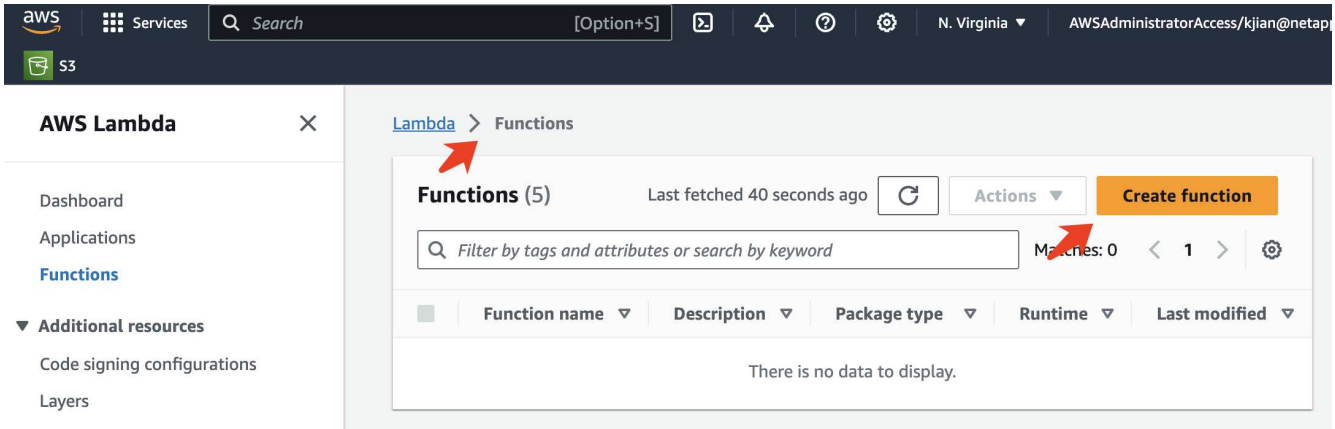
6. 作成した* Lifecycle configuration を選択し、 Update notebook instance *をクリックします。



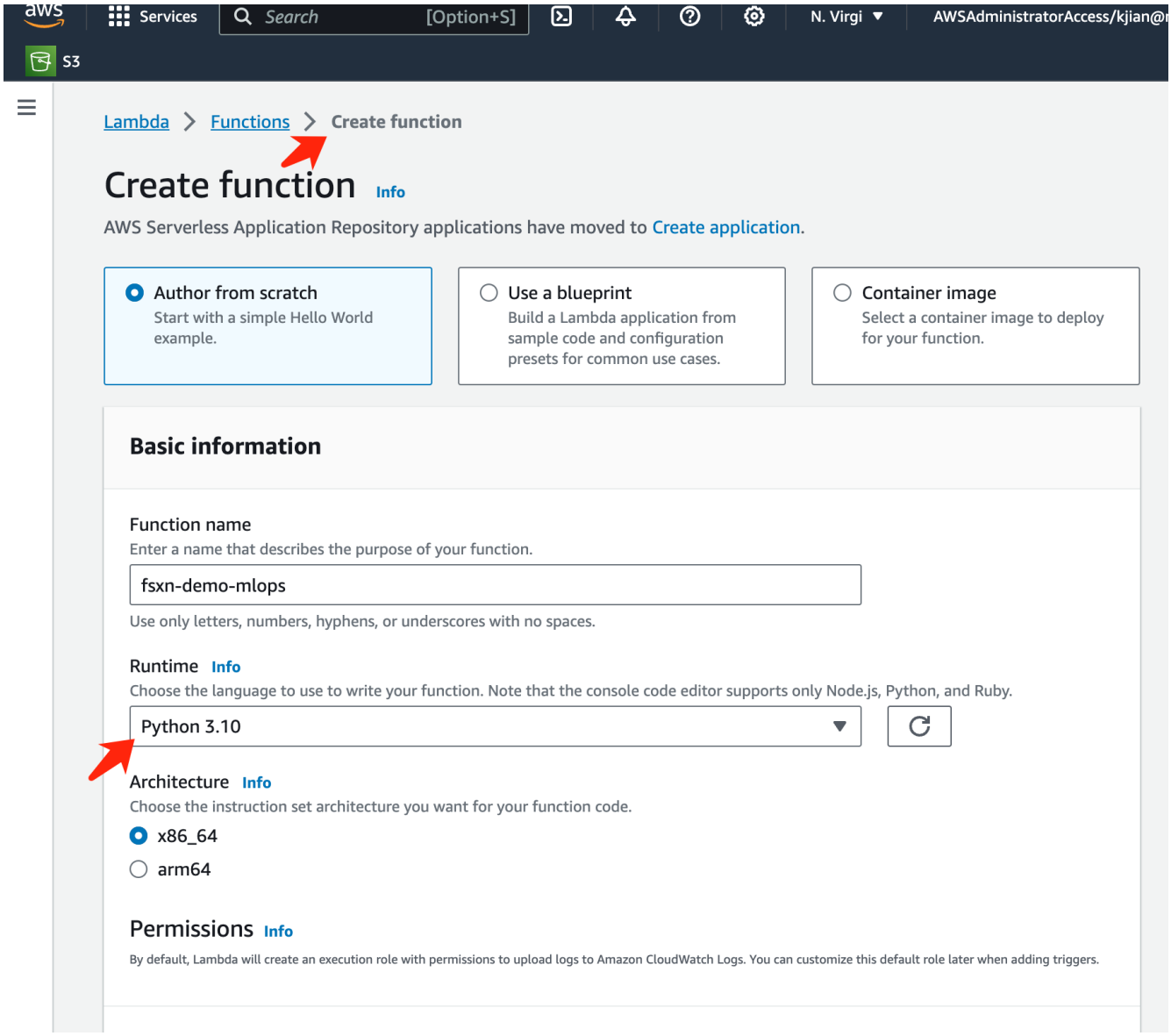
AWS Lambdaサーバレス関数

前述したように、* AWS Lambda関数*は* AWS SageMaker Notebookインスタンス*のスピンアップを担当します。

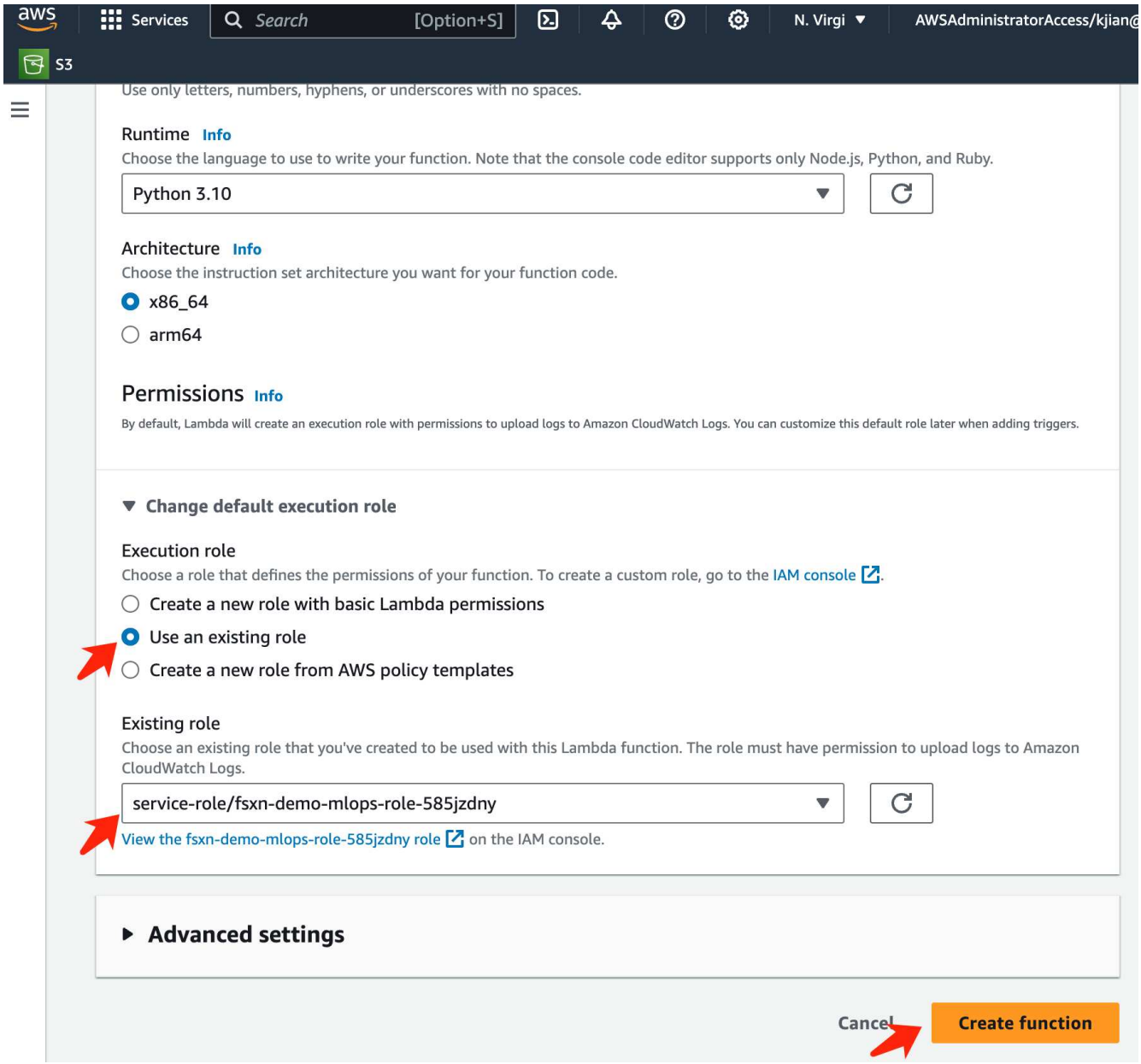
1. AWS Lambda Function を作成するには、該当するパネルに移動し、 Functions タブに切り替えて Create Function *をクリックします。



2. ページに必要なすべてのエントリをファイルし、ランタイムを*Python 3.10*に切り替えることを忘れないでください。



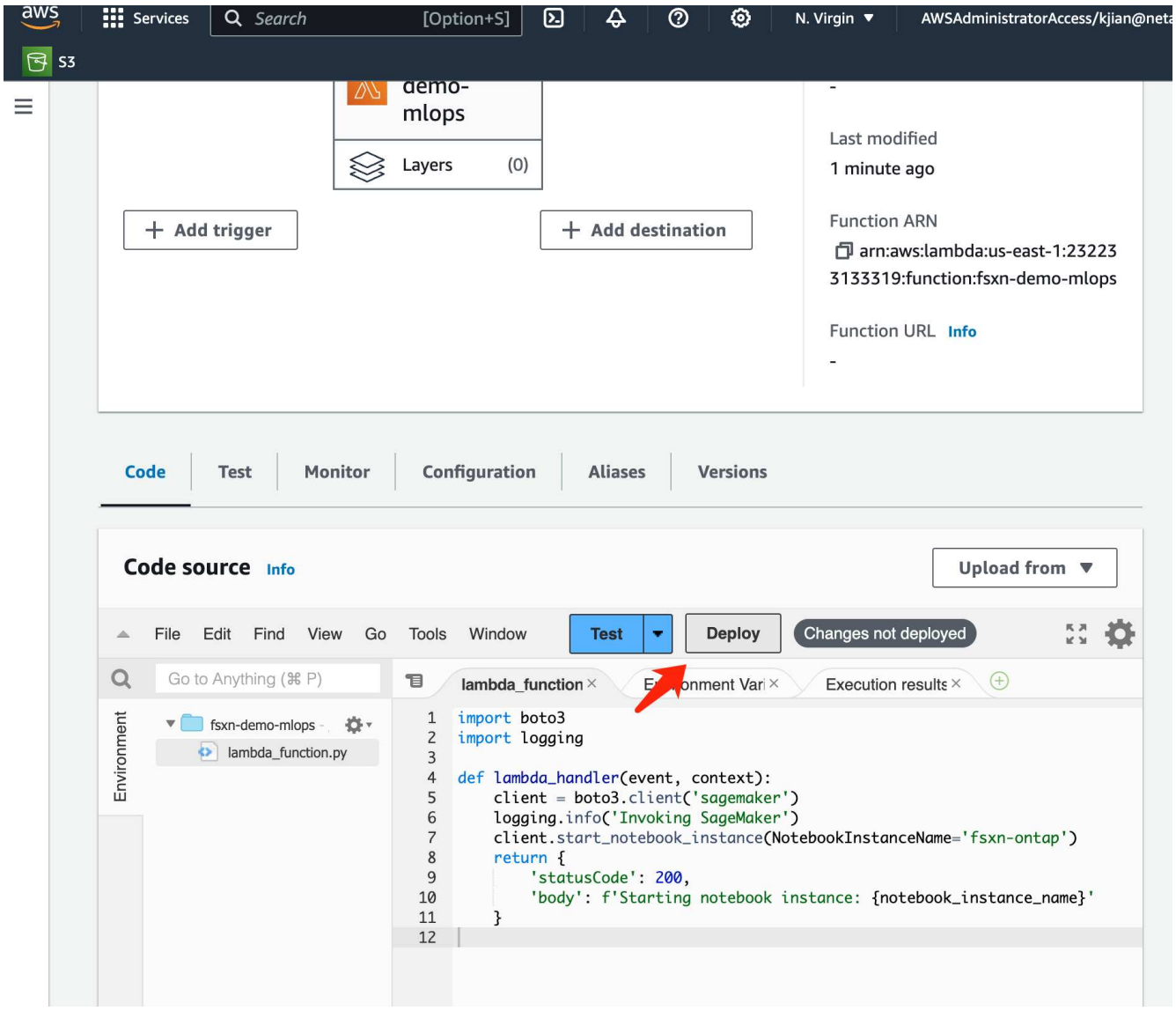
3. 指定されたロールに必要な権限* AmazonSageMakerFullAccess*があることを確認し、* Create Function * ボタンをクリックしてください。



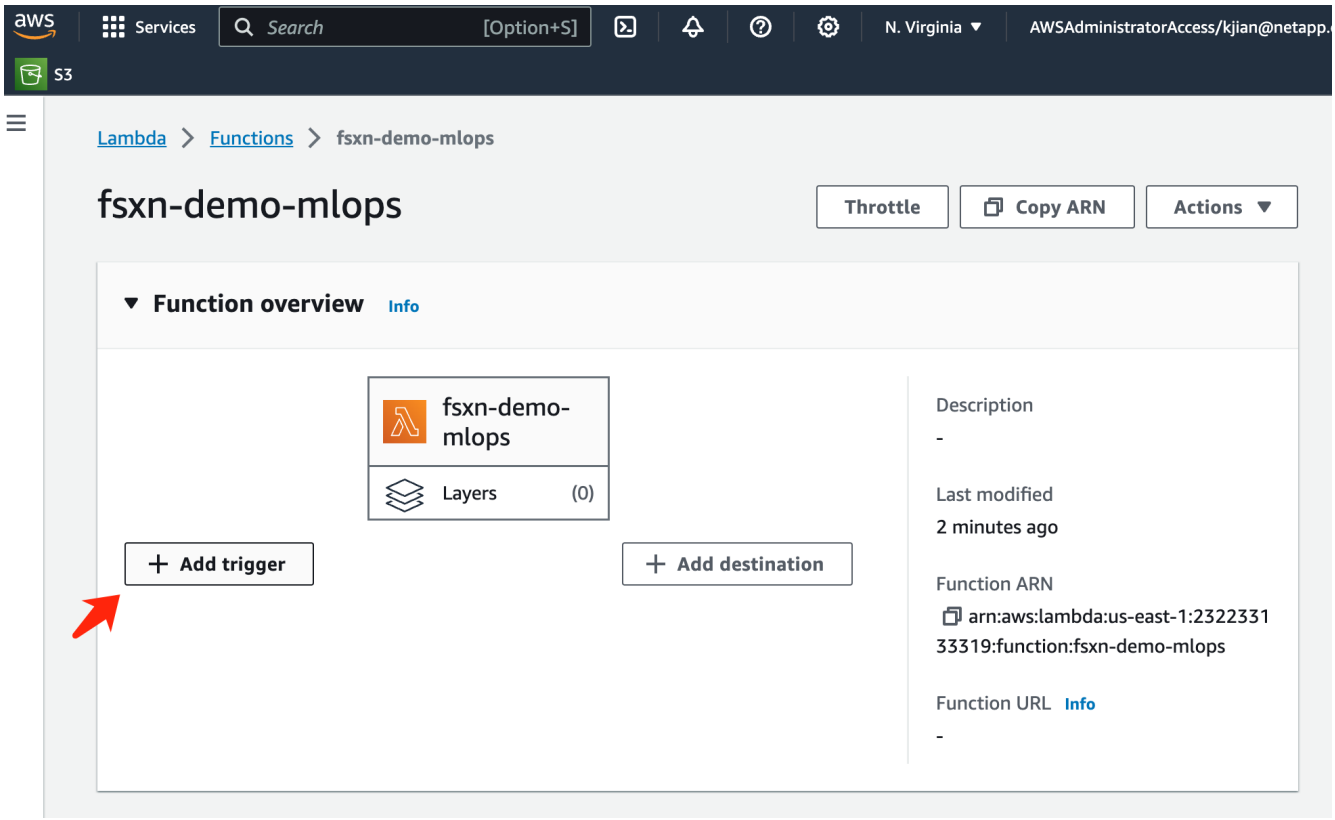
4. 作成したLambda関数を選択します。[コード]タブで、次のコードをコピーしてテキスト領域に貼り付けます。このコードは、* fsxn-ontap *という名前のノートブックインスタンスを起動します。

```
import boto3
import logging

def lambda_handler(event, context):
    client = boto3.client('sagemaker')
    logging.info('Invoking SageMaker')
    client.start_notebook_instance(NotebookInstanceName='fsxn-ontap')
    return {
        'statusCode': 200,
        'body': f'Starting notebook instance: {notebook_instance_name}'
    }
```

5. このコード変更を適用するには、*配布*ボタンをクリックします。

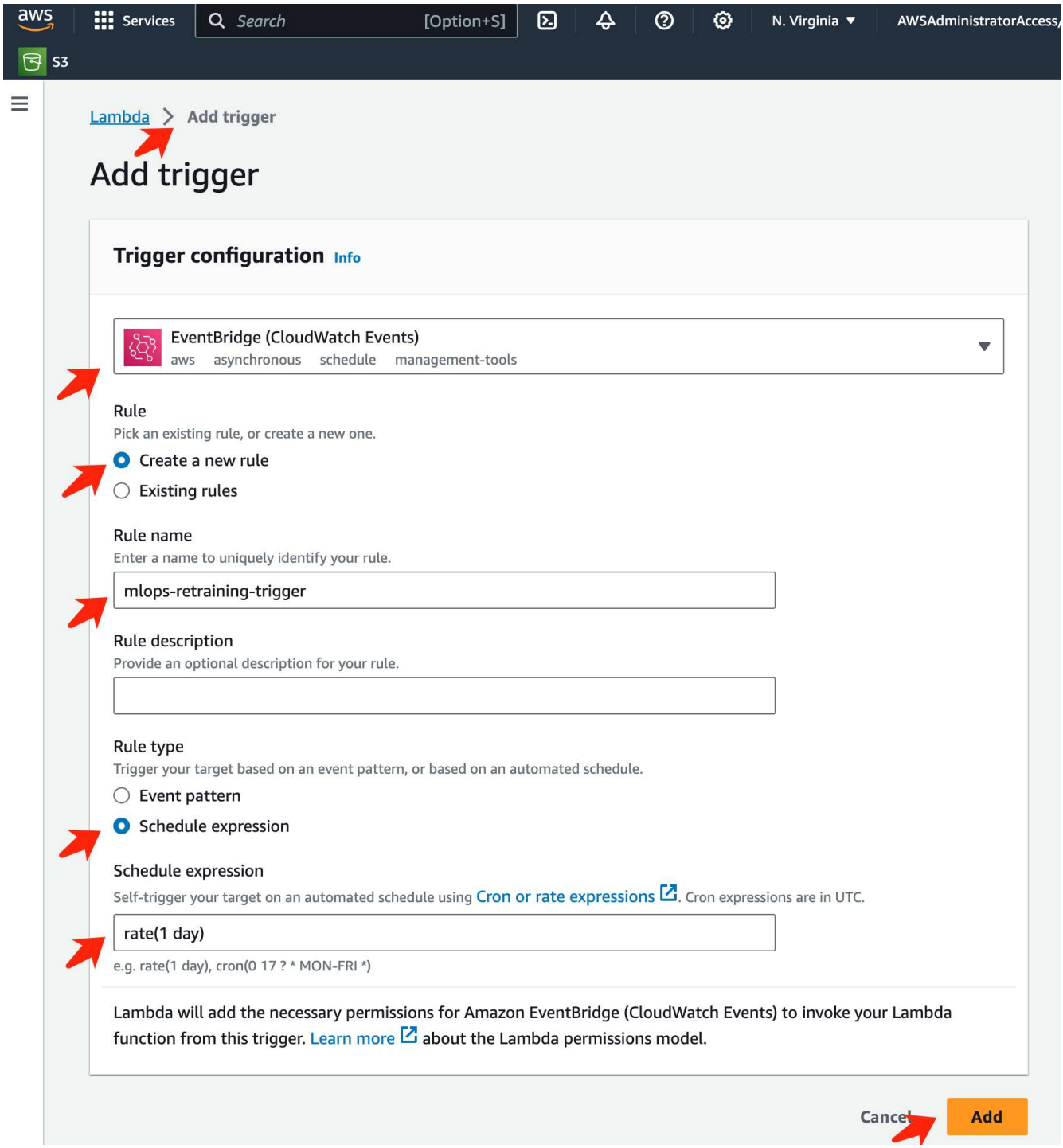


6. このAWS Lambda関数をトリガーする方法を指定するには、Add Triggerボタンをクリックします。



"]

7. ドロップダウンメニューから[EventBridge]を選択し、[Create a new rule]というラベルの付いたラジオボタンをクリックします。[スケジュール式]フィールドに、次のように入力します。 `rate(1 day)`をクリックし、[追加]ボタンをクリックして、この新しいcronジョブルールを作成し、AWS Lambda関数に適用します。



2段階の設定が完了すると、* AWS Lambda関数*が毎日* SageMaker Notebook を開始し、 FSxN リポジトリのデータを使用してモデルの再トレーニングを実行し、更新されたモデルを本番環境に再導入し、SageMaker Notebook *インスタンスを自動的にシャットダウンしてコストを最適化します。これにより、モデルが常に最新の状態に保たれます。

これで、MLOpsパイプラインを開発するためのチュートリアルは終了です。

Domino Data LabとNetAppによるハイブリッドマルチクラウドMLOps

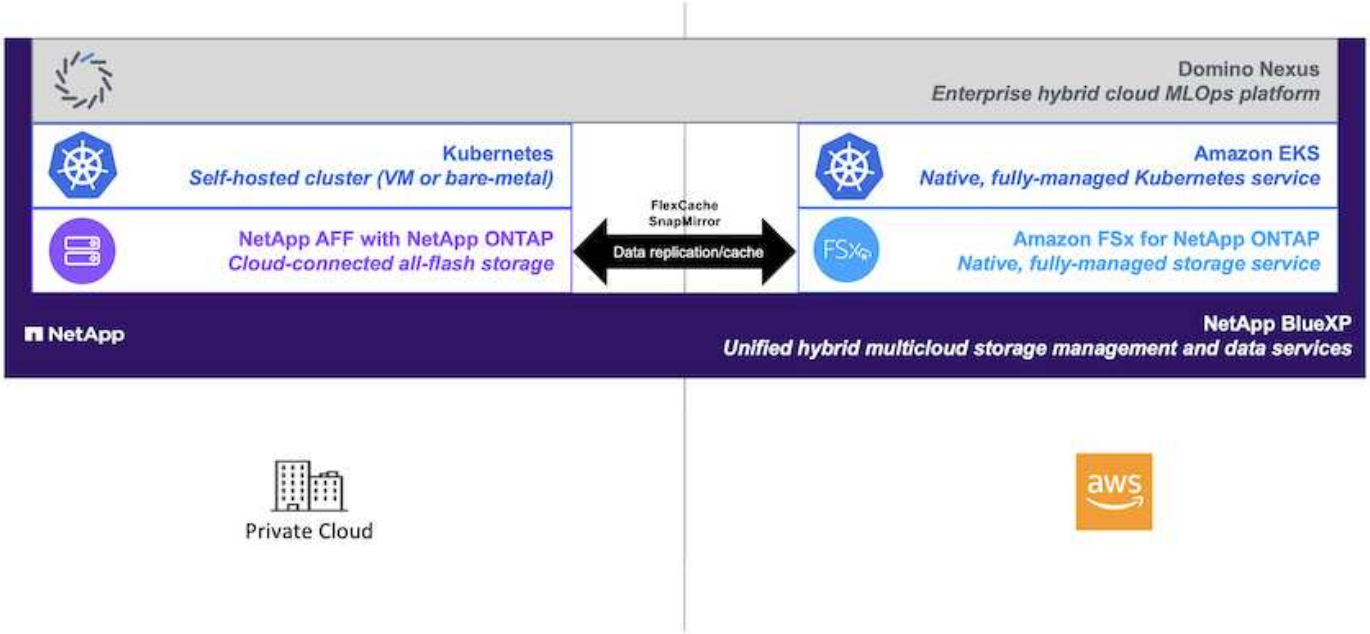
Domino Data LabとNetAppによるハイブリッドマルチクラウドMLOps

ネットアップ、 Mike Oglesby

現在、世界中の組織が、ビジネスとプロセスを変革するためにAIを導入しています。そのため、多くの場合、AI対応のコンピューティングインフラが不足しています。企業は、さまざまな地域、データセンター、クラウドで利用可能なコンピューティング環境を活用して、コスト、可用性、パフォーマンスのバランスを取るために、ハイブリッドマルチクラウドMLOpsアーキテクチャを採用しています。

Domino Nexusは、Domino Data Labが提供する統合MLOpsコントロールプレーンです。クラウド、リージョン、オンプレミスなど、あらゆるコンピューティングクラスタにわたってデータサイエンスと機械学習のワークロードを実行できます。企業全体のデータサイエンスのサイロが統合されるため、モデルの構築、導入、監視を1箇所で行うことができます。同様に、ネットアップのハイブリッドクラウドデータ管理機能を使用すれば、実行されている場所に関係なく、ジョブやワークスペースにデータを移動できます。Domino NexusをNetAppとペアリングすると、データの可用性を心配することなく、複数の環境にわたってワークロードのスケジュールを柔軟に設定できます。つまり、ワークロードとデータを適切なコンピューティング環境に送信できるため、AIの導入を高速化しながら、データプライバシーとデータ主権に関する規制に対応できます。

この解決策では、オンプレミスのKubernetesクラスタとAmazon Web Services（AWS）で実行されるElastic Kubernetes Service（EKS）クラスタを組み込んだ統合MLOpsコントロールプレーンの導入について説明します。



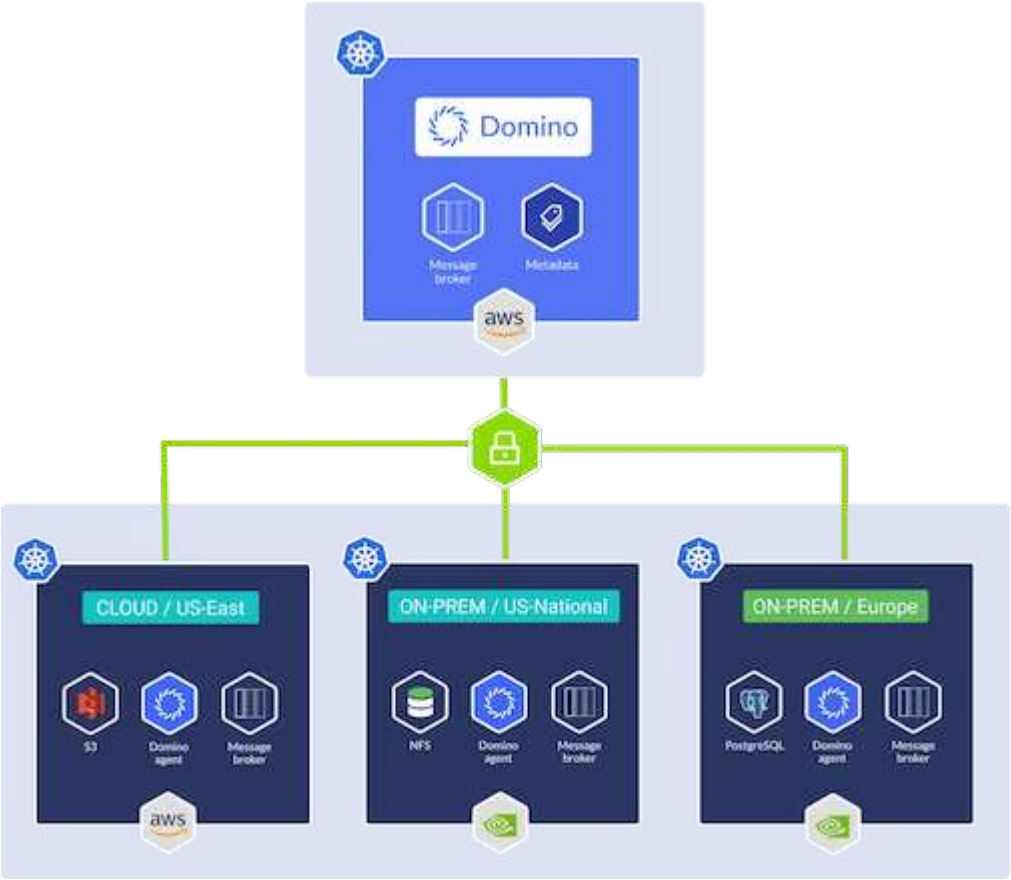
テクノロジの概要

Dominoデータラボ

Domino Data Labは、Fortune 100企業の20%以上から信頼されている業界をリードするエンタープライズAIプラットフォームでモデル主導型ビジネスを支援しています。Dominoは、コラボレーションとガバナンスを強化しながら、データサイエンスの開発と導入を加速します。Dominoを使用することで、世界中の企業はより良い医薬品を開発し、より生産的な作物を栽培し、より良い車を製造することができます。2013年に設立されたドミノは、Coatue Management、Great Hill Partners、Highland Capital、Sequoia Capitalなどの主要な投資家の支援を受けています。

Dominoを使用すると、企業とそのデータサイエンティストは、統合されたエンドツーエンドのプラットフォーム上でAIの構築、導入、管理を高速で、責任を持って対費用効果の高い方法で行うことができます。チームは、環境に関係なく、必要なすべてのデータ、ツール、コンピューティング、モデル、プロジェクトにアクセスできるため、コラボレーション、過去の作業の再利用、本番環境でのモデルの追跡による精度の向上、ベストプラクティスの標準化、AIの責任と管理が可能になります。

- *オープンで柔軟性：*オープンソース、商用ツール、およびインフラストラクチャの広範なエコシステムにアクセスして、最高のイノベーションを実現し、ベンダーロックインを回避します。
- *記録システム：*企業全体のAI運用と知識のための中央ハブであり、ベストプラクティス、部門横断的なコラボレーション、迅速なイノベーション、効率化を実現します。
- *統合：*統合されたワークフローと自動化は、企業のプロセス、制御、ガバナンスのために構築されており、コンプライアンスと規制のニーズを満たします。
- *ハイブリッドマルチクラウド：*オンプレミス、ハイブリッド、あらゆるクラウド、マルチクラウドなど、データの近くでAIワークロードを実行できるため、コストを削減し、パフォーマンスとコンプライアンスを最適化できます。



Domino Nexus

Domino Nexusは、クラウド、リージョン、オンプレミスなど、あらゆるコンピューティングクラスタにわたってデータサイエンスと機械学習のワークロードを実行できる単一のコンソールです。企業全体のデータサイエンスのサイロが統合されるため、モデルの構築、導入、監視を1箇所で行うことができます。

NetApp BlueXP

NetApp BlueXPは、ネットアップのすべてのストレージサービスとデータサービスを1つのツールに統合し、ハイブリッドマルチクラウドのデータ資産を構築、保護、管理できます。オンプレミス環境とクラウド環境に

わたってストレージとデータサービスのエクスペリエンスを統一し、柔軟な消費パラメータと、今日のクラウド主導の世界に必要な統合された保護機能を備えたAIOpsによる運用の簡易化を実現します。

NetApp ONTAP

ネットアップが提供する最新世代のストレージ管理ソフトウェアONTAP 9を使用すれば、インフラを最新化し、クラウド対応のデータセンターに移行できます。ONTAP は、業界をリードするデータ管理機能を活用して、データの格納場所に関係なく、単一のツールセットでデータの管理と保護を実現します。エッジ、コア、クラウドなど、必要な場所に自由にデータを移動することもできます。ONTAP 9には、データ管理の簡易化、重要なデータの高速化と保護、ハイブリッドクラウドアーキテクチャ全体で次世代インフラ機能を実現する多数の機能が搭載されています。

データ管理を簡易化

データ管理は、AIアプリケーションの運用やAI / MLデータセットのトレーニングに適切なリソースを使用できるように、エンタープライズIT運用とデータサイエンティストにとって非常に重要です。以下に記載するネットアップテクノロジに関する追加情報 は、この検証の対象外ですが、導入環境によっては関連性がある場合もあります。

ONTAP データ管理ソフトウェアには、運用を合理化および簡易化し、総運用コストを削減するための次の機能が含まれています。

- インラインデータコンパクション、強化された重複排除：データコンパクションはストレージブロック内の無駄なスペースを削減し、重複排除は実効容量を大幅に増やします。この環境データはローカルに格納され、データはクラウドに階層化されます。
- 最小、最大、アダプティブのQuality of Service（AQoS）。きめ細かいサービス品質（QoS）管理機能により、高度に共有された環境で重要なアプリケーションのパフォーマンスレベルを維持できます。
- NetApp FabricPool の略。Amazon Web Services（AWS）、Azure、NetApp StorageGRID ストレージ解決策 など、パブリッククラウドとプライベートクラウドのストレージオプションへコールドデータを自動的に階層化します。FabricPool の詳細については、を参照してください "TR-4598：『FabricPool best bests』"。

データの高速化と保護

ONTAP は、卓越したパフォーマンスとデータ保護を実現し、以下の方法でこれらの機能を拡張します。

- パフォーマンスとレイテンシの低下：ONTAP は、可能なかぎり最小のレイテンシで最高のスループットを提供します。
- データ保護ONTAP には、組み込みのデータ保護機能が用意されており、すべてのプラットフォームを共通の管理機能で管理できます。
- NetApp Volume Encryption（NVE）：ONTAP は、オンボードと外部キー管理の両方をサポートし、ボリュームレベルでのネイティブな暗号化を実現します。
- マルチテナンシーおよび多要素認証ONTAP を使用すると、最高レベルのセキュリティでインフラリソースを共有できます。

将来のニーズにも対応できるインフラ

ONTAP は、次の機能を備えており、要件が厳しく、絶えず変化するビジネスニーズに対応できます。

- シームレスな拡張とノンストップオペレーションONTAP を使用すると、既存のコントローラとスケール

アウトクラスタに無停止で容量を追加できます。NVMe や 32Gb FC などの最新テクノロジへのアップグレードも、コストのかかるデータ移行やシステム停止を行わずに実行できます。

- クラウドへの接続：ONTAPは、ほとんどのクラウドに対応したストレージ管理ソフトウェアで、すべてのパブリッククラウドでSoftware-Defined Storageとクラウドネイティブインスタンスを選択できます。
- 新しいアプリケーションとの統合：ONTAP は、既存のエンタープライズアプリケーションをサポートするインフラを使用して、自律走行車、スマートシティ、インダストリー4.0などの次世代プラットフォームやアプリケーション向けにエンタープライズクラスのデータサービスを提供します。

NetApp ONTAP 対応の Amazon FSX

Amazon FSx for NetApp ONTAPは、ネットアップが人気のONTAPファイルシステムを基盤に構築された、信頼性、拡張性、パフォーマンス、機能豊富なファイルストレージを提供する、ファーストパーティのフルマネージドAWSサービスです。FSX for ONTAP は、ネットアップファイルシステムの使い慣れた機能、パフォーマンス、機能、API操作に、AWSのフルマネージドサービスならではの即応性、拡張性、シンプルさを兼ね備えています。

ネットアップアストラト Trident

Astra Tridentでは、パブリッククラウドやオンプレミスにあるONTAP（AFF、NetApp FAS、Select、Cloud、Amazon FSx for NetApp ONTAP）、Elementソフトウェア（NetApp HCI、SolidFire）、Azure NetApp Filesサービス、Cloud Volumes Service on Google CloudAstra Tridentは、Kubernetesとネイティブに統合される、コンテナストレージインターフェイス（CSI）に準拠した動的ストレージオーケストレーションツールです。

Kubernetes

Kubernetes は、 Google が当初設計した、オープンソースの分散型コンテナオーケストレーションプラットフォームであり、 Cloud Native Computing Foundation （ CNCF ）によって管理されています。Kubernetes は、コンテナ化されたアプリケーションの導入、管理、拡張の自動化機能を可能にし、エンタープライズ環境における主要なコンテナオーケストレーションプラットフォームです。

Amazon Elastic Kubernetes Service（EKS）

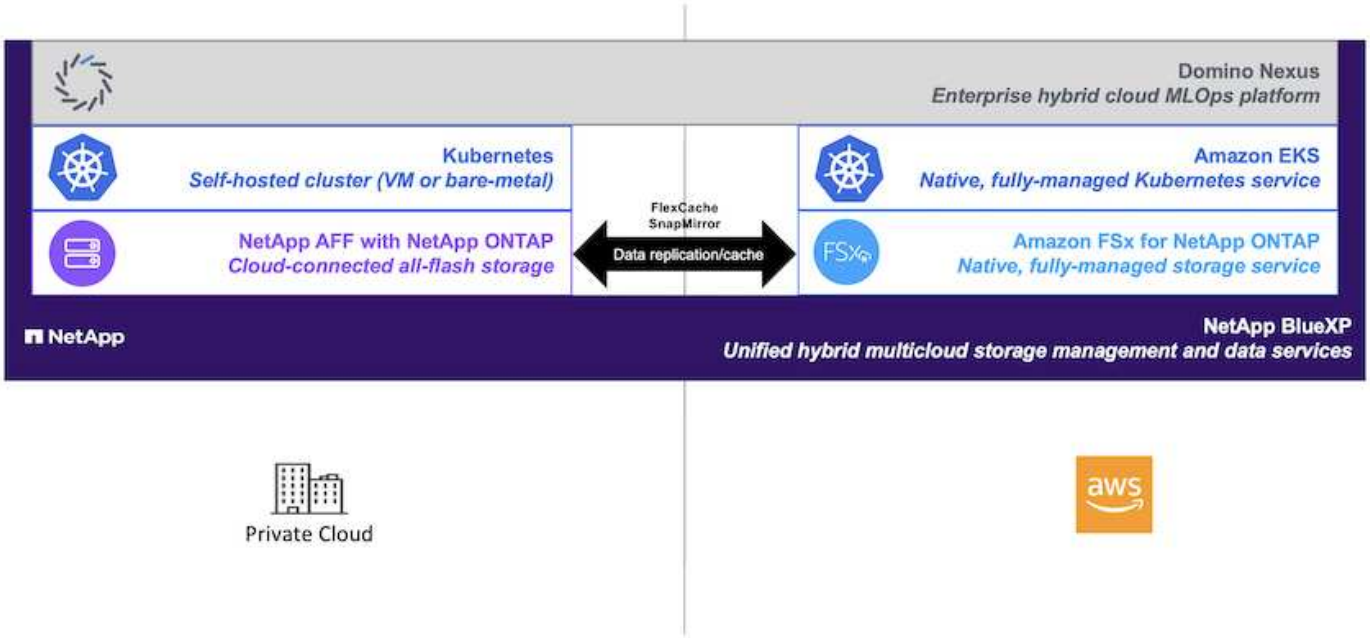
Amazon Elastic Kubernetes Service（Amazon EKS）は、AWSクラウドで運用されるマネージドKubernetesサービスです。Amazon EKSは、コンテナのスケジュール設定、アプリケーションの可用性の管理、クラスタデータの格納、その他の重要なタスクを担当するKubernetesコントロールプレーンノードの可用性と拡張性を自動的に管理します。Amazon EKSを使用すると、AWSインフラのパフォーマンス、拡張性、信頼性、可用性のすべてを活用できるだけでなく、AWSのネットワークサービスやセキュリティサービスとの統合も利用できます。

アーキテクチャ

この解決策は、Domino Nexusのハイブリッドマルチクラウドワークロードのスケジューリング機能とNetAppデータサービスを組み合わせて、ハイブリッドクラウドの統合MLOpsプラットフォームを構築します。詳細については、次の表を参照してください。

コンポーネント	名前	環境
MLOpsコントロールプレーン	"Domino Nexusを使用したDomino Enterprise AIプラットフォーム"	AWS

コンポーネント	名前	環境
MLOpsプラットフォームのコンピューティング環境	"Domino Nexusデータプレーン"	AWS、オンプレミスデータセンター
オンプレミスのコンピューティングプラットフォーム	"Kubernetes" を使用 "ネットアップアストラト Trident"	オンプレミスのデータセンター
クラウドコンピューティングプラットフォーム	"Amazon Elastic Kubernetes Service（EKS）" を使用 "ネットアップアストラト Trident"	AWS
オンプレミスのデータプラットフォーム	"NetAppストレージアプライアンス" 搭載： "NetApp ONTAP"	オンプレミスのデータセンター
クラウドデータプラットフォーム	"NetApp ONTAP 対応の Amazon FSX"	AWS



初期セットアップ

ここでは、オンプレミスのデータセンターとAWSを組み合わせたハイブリッド環境でDomino NexusとNetAppデータサービスを活用するために必要な初期セットアップタスクについて説明します。

前提条件

このセクションで説明する手順を実行する前に、次のタスクがすでに実行されていることを前提としています。

- オンプレミスのNetApp ONTAPストレージプラットフォームの導入と設定が完了している。詳細については、を参照してください "ネットアップの製品マニュアル"。
- AWSでAmazon FSx for NetApp ONTAPインスタンスをすでにプロビジョニングしておきます。詳細については、を参照してください "Amazon FSx for NetApp ONTAPの製品ページ"。
- オンプレミスのデータセンターにKubernetesクラスタをプロビジョニング済みである。詳細については、

を参照してください "Domino管理者ガイド"。

- AWSでAmazon EKSクラスタをプロビジョニング済みである。詳細については、を参照してください "Domino管理者ガイド"。
- オンプレミスのKubernetesクラスタにNetApp Astra Tridentをインストールしました。また、ストレージリソースのプロビジョニングと管理にオンプレミスのNetApp ONTAPストレージプラットフォームを使用するようにTridentインスタンスを設定しました。詳細については、を参照してください "ネットアップ Astra Trident のドキュメント"。
- NetApp Astra TridentをAmazon EKSクラスタにインストールしておきます。また、ストレージリソースのプロビジョニングと管理にAmazon FSx for NetApp ONTAPインスタンスを使用するようにTridentインスタンスを設定しました。詳細については、を参照してください "ネットアップ Astra Trident のドキュメント"。
- オンプレミスのデータセンターとAWSの仮想プライベートクラウド（VPC）の間に双方向のネットワーク接続が必要です。これを実装するためのさまざまなオプションの詳細については、を参照してください。"Amazon Virtual Private Network（VPN）ドキュメント"。

Domino Enterprise AI PlatformをAWSにインストールする

Domino Enterprise MLOps PlatformをAWSにインストールするには、に記載されている手順に従います。"Domino管理者ガイド"。Dominoは、以前にプロビジョニングしたのと同じAmazon EKSクラスタに導入する必要があります。また、このEKSクラスタにNetApp Astra Tridentがインストールおよび設定されている必要があります。また、Domino.ymlインストール構成ファイルで、Tridentが管理するストレージクラスを共有ストレージクラスとして指定する必要があります。

> を参照してください "Dominoインストール構成リファレンスガイド" Domino.ymlインストール構成ファイルで共有ストレージクラスを指定する方法の詳細については、を参照してください。

> "テクニカルレポートTR-4952" Amazon FSx for NetApp ONTAPを使用したAWSでのDominoの導入について説明します。発生した問題のトラブルシューティングに役立つ参考資料になります。

Domino Nexusの有効化

次に、Domino Nexusを有効にする必要があります。を参照してください "Domino管理者ガイド" を参照してください。

オンプレミスのデータセンターにDominoデータプレーンを導入

次に、オンプレミスのデータセンターにDominoデータプレーンを導入する必要があります。このデータプレーンは、以前にプロビジョニングしたオンプレミスのKubernetesクラスタに導入する必要があります。また、このKubernetesクラスタにNetApp Astra Tridentがインストールおよび設定されている必要があります。を参照してください "Domino管理者ガイド" を参照してください。

既存のNetAppボリュームをDominoに公開

このセクションでは、既存のNetApp ONTAP NFSボリュームをDomino MLOpsプラットフォームに公開するために実行する必要があるタスクについて説明します。オンプレミスとAWSの両方で、同じ手順を実行します。

NetApp ONTAP VolumeをDominoに公開する理由

NetAppボリュームとDominoを併用すると、次のようなメリットがあります。

- NetApp ONTAPのスケールアウト機能を活用することで、きわめて大規模なデータセットに対してワークロードを実行できます。
- 複数のコンピューティングノード間でワークロードを実行できます。個 々 のノードにデータをコピーする必要はありません。
- ネットアップのハイブリッドマルチクラウドデータ移動機能と同期機能を活用して、複数のデータセンターやクラウドにわたってデータにアクセスできます。
- 別のデータセンターやクラウドにデータのキャッシュをすばやく簡単に作成したいと考えています。

Astra Tridentでプロビジョニングされていない既存のNFSボリュームを公開

既存のNetApp ONTAP NFSボリュームがAstra Tridentでプロビジョニングされていなかった場合は、このサブセクションで説明する手順を実行します。

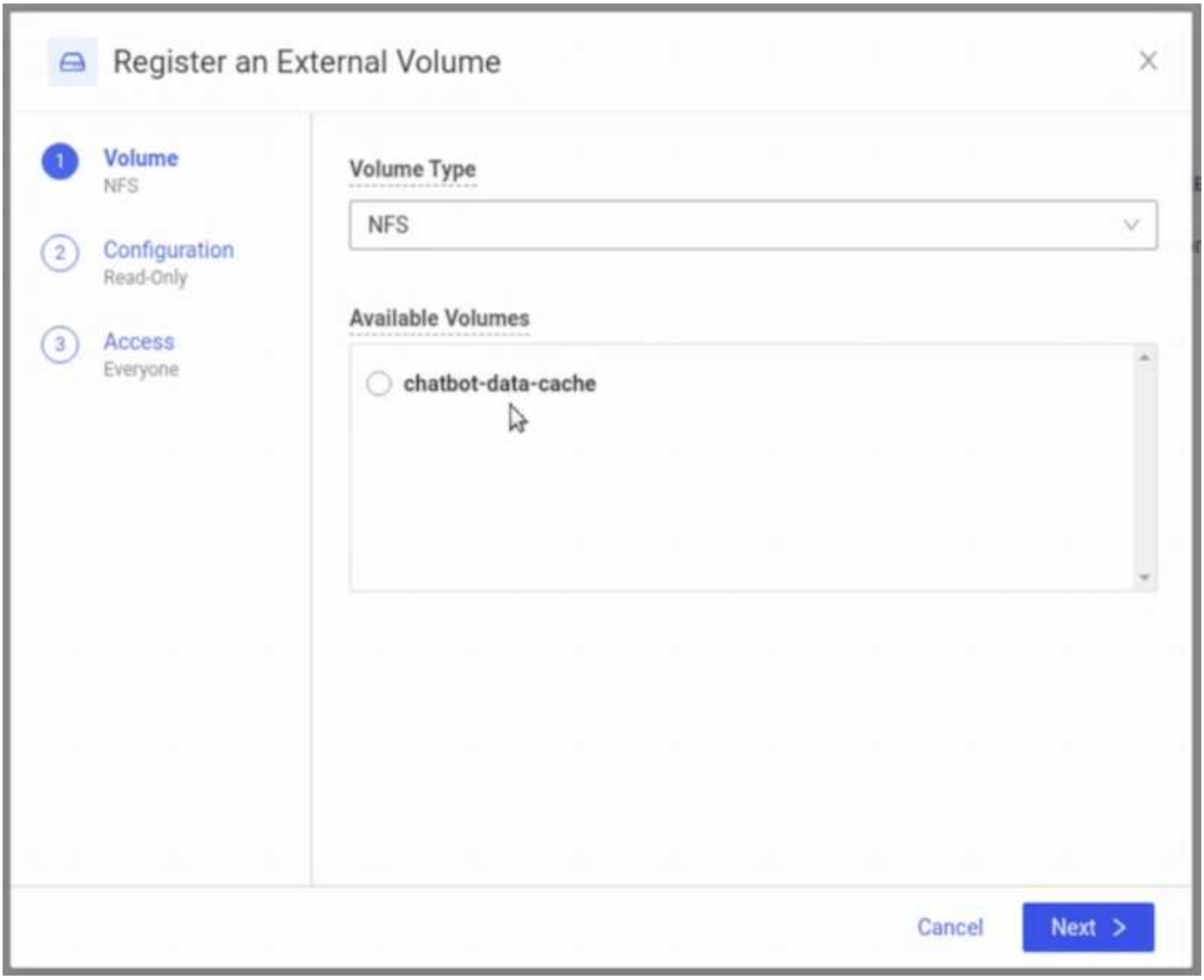
KubernetesでPVとPVCを作成

> オンプレミスボリュームの場合は、オンプレミスのKubernetesクラスタにPVとPVCを作成します。Amazon FSx for NetApp ONTAPボリュームの場合は、Amazon EKSでPVとPVCを作成します。

まず、Kubernetesクラスタに永続ボリューム（PV）と永続ボリューム要求（PVC）を作成する必要があります。PVおよびPVCを作成するには、 "NFS PV/PVCの例" Domino管理者ガイドから、環境に合わせて値を更新します。の正しい値を指定してください `namespace`、 `nfs.path`および `nfs.server` フィールド。また、PVとPVCには、対応するONTAP NFSボリュームに格納されているデータの性質を表す一意の名前を付けることを推奨します。たとえば、ボリュームに製造上の欠陥の画像が含まれている場合は、PVという名前を付けることができます。 `pv-mfg-defect-images`および`PVC` `pvc-mfg-defect-images`。

Dominoへの外部データボリュームの登録

次に、Dominoに外部データボリュームを登録する必要があります。外部データボリュームを登録するには、 "手順" Domino管理者ガイドを参照してください。ボリュームを登録するときは、[ボリュームタイプ]ドロップダウンメニューから[NFS]を選択してください。[NFS]を選択すると、[Available Volumes]リストにPVCが表示されます。



Astra Tridentでプロビジョニングされた既存ボリュームを公開

既存のボリュームがAstra Tridentでプロビジョニングされていた場合は、このサブセクションで説明する手順に従います。

既存の**PVC**の編集

ボリュームがAstra Tridentによってプロビジョニングされていた場合は、ボリュームに対応する永続的ボリューム要求（PVC）がすでに設定されています。このボリュームをDominoに公開するには、PVCを編集し、次のラベルを `metadata.labels` フィールド：

```
"dominodatalab.com/external-data-volume": "Generic"
```

Dominoへの外部データボリュームの登録

次に、Dominoに外部データボリュームを登録する必要があります。外部データボリュームを登録するには、 "手順" Domino管理者ガイドを参照してください。ボリュームを登録する際は、必ず「ボリュームタイプ」ドロップダウンメニューから「汎用」を選択してください。[Generic]を選択すると、[Available Volumes]リスト

にPVCが表示されます。

異なる環境間で同じデータにアクセス

このセクションでは、異なるコンピューティング環境間で同じデータにアクセスするために実行する必要があるタスクについて説明します。Domino MLOpsプラットフォームでは、コンピューティング環境を「データプレーン」と呼びます。 データがあるデータプレーンのNetAppボリュームに格納されていて、別のデータプレーンでアクセスする必要がある場合は、このセクションで説明するタスクを実行します。このタイプのシナリオは、「バースト」またはデスティネーション環境がクラウドの場合は「クラウドバースト」と呼ばれます。 この機能は、限られたコンピューティングリソースやオーバーサブスクライブされたコンピューティングリソースを扱うときに必要になることがよくあります。たとえば、オンプレミスのコンピューティングクラスタがオーバーサブスクライブされている場合、ワークロードをすぐに開始できるクラウドにスケジュールすることができます。

別のデータプレーンにあるNetAppボリュームにアクセスする場合に推奨される方法は2つあります。これらのオプションについては、次のサブセクションで説明します。特定の要件に応じて、これらのオプションのいずれかを選択します。次の表に、2つのオプションの利点と欠点を示します。

オプション	利点	欠点
オプション1 -キャッシュ	-シンプルなワークフロー -ニーズに基づいてデータのサブセットをキャッシュする機能 -ソースにデータを書き戻す機能 -管理するリモートコピーがない	-キャッシュがハイドレーションされると、初期データアクセスのレイテンシが増加します。
オプション2 -ミラーリング	-ソースボリュームのフルコピー -キャッシュハイドレーションによるレイテンシ増加なし（ミラー処理完了後）	-データにアクセスする前にミラー処理が完了するまで待つ必要がある -リモートコピーの管理が必要 -ソースに書き戻す機能がない

オプション**1 -**別のデータプレーンに存在するボリュームのキャッシュを作成する

を使用 "NetApp FlexCache テクノロジ"では、別のデータプレーンにあるNetAppボリュームのキャッシュを作成できます。たとえば、オンプレミスのデータプレーンにNetAppボリュームがあり、そのボリュームにAWSデータプレーンでアクセスする必要がある場合は、AWSにボリュームのキャッシュを作成できます。このセクションでは、別のデータプレーンにあるNetAppボリュームのキャッシュを作成するために実行する必要があるタスクの概要を説明します。

デスティネーション環境での**FlexCache**ボリュームの作成

> デスティネーション環境がオンプレミスのデータセンターの場合は、オンプレミスのONTAPシステムにFlexCacheボリュームを作成します。デスティネーション環境がAWSの場合は、Amazon FSx for NetApp ONTAPインスタンスにFlexCacheボリュームを作成します。

最初に、デスティネーション環境にFlexCacheボリュームを作成する必要があります。

FlexCacheボリュームの作成にはBlueXPを使用することを推奨します。BlueXPでFlexCacheボリュームを作

成するには、 "BlueXPのボリュームキャッシュに関するドキュメント"。

BlueXPを使用しない場合は、ONTAPシステムマネージャまたはONTAP CLIを使用してFlexCacheボリュームを作成できます。System ManagerでFlexCacheボリュームを作成する手順については、 "ONTAP のドキュメント"。ONTAP CLIを使用してFlexCacheボリュームを作成するには、 "ONTAP のドキュメント"。

このプロセスを自動化するには、 "BlueXP API"、 "ONTAP REST API"または "ONTAP Ansibleコレクション"。

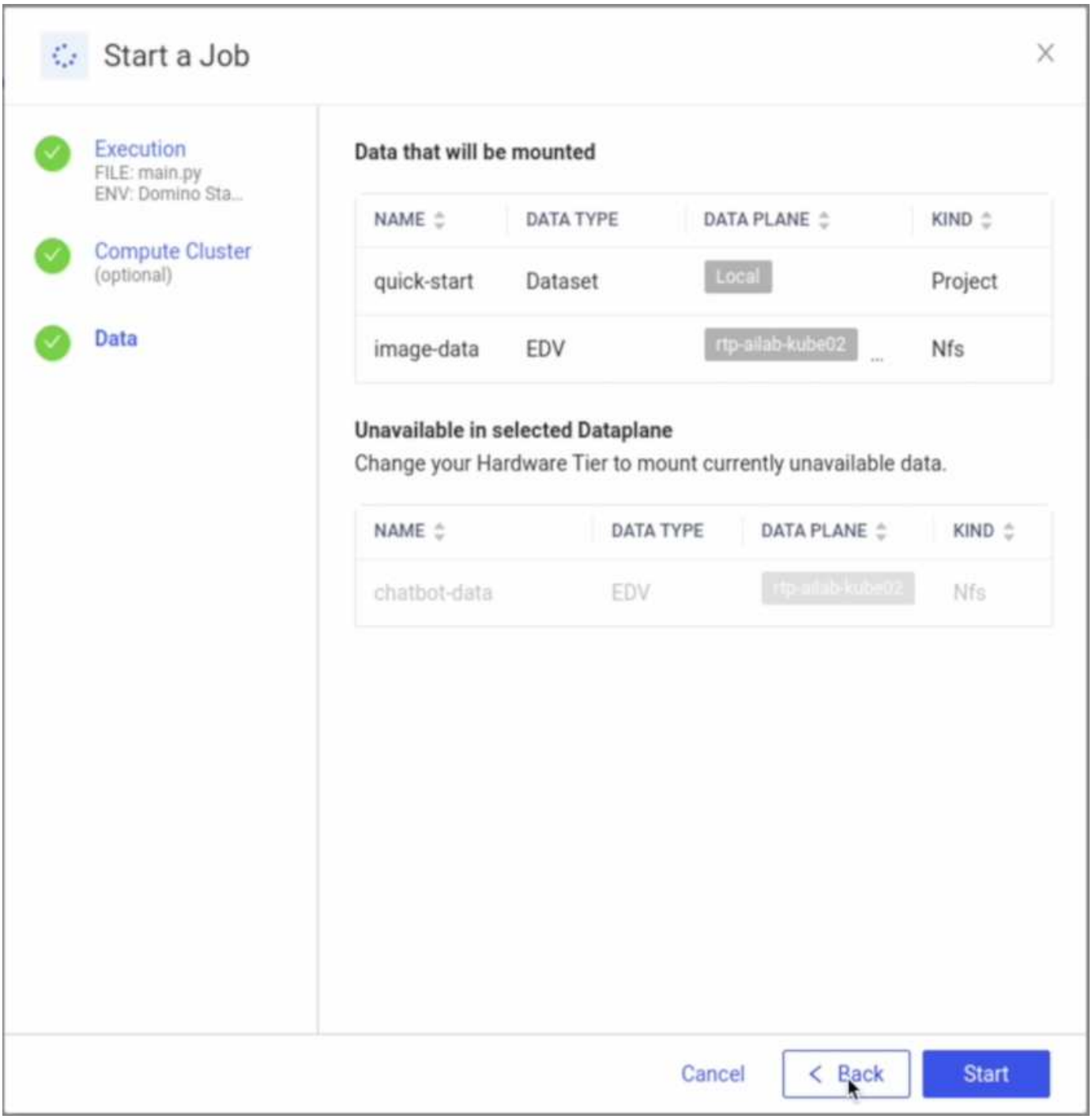
System ManagerはAmazon FSx for NetApp ONTAPでは使用できません。

FlexCacheボリュームを**Domino**に公開

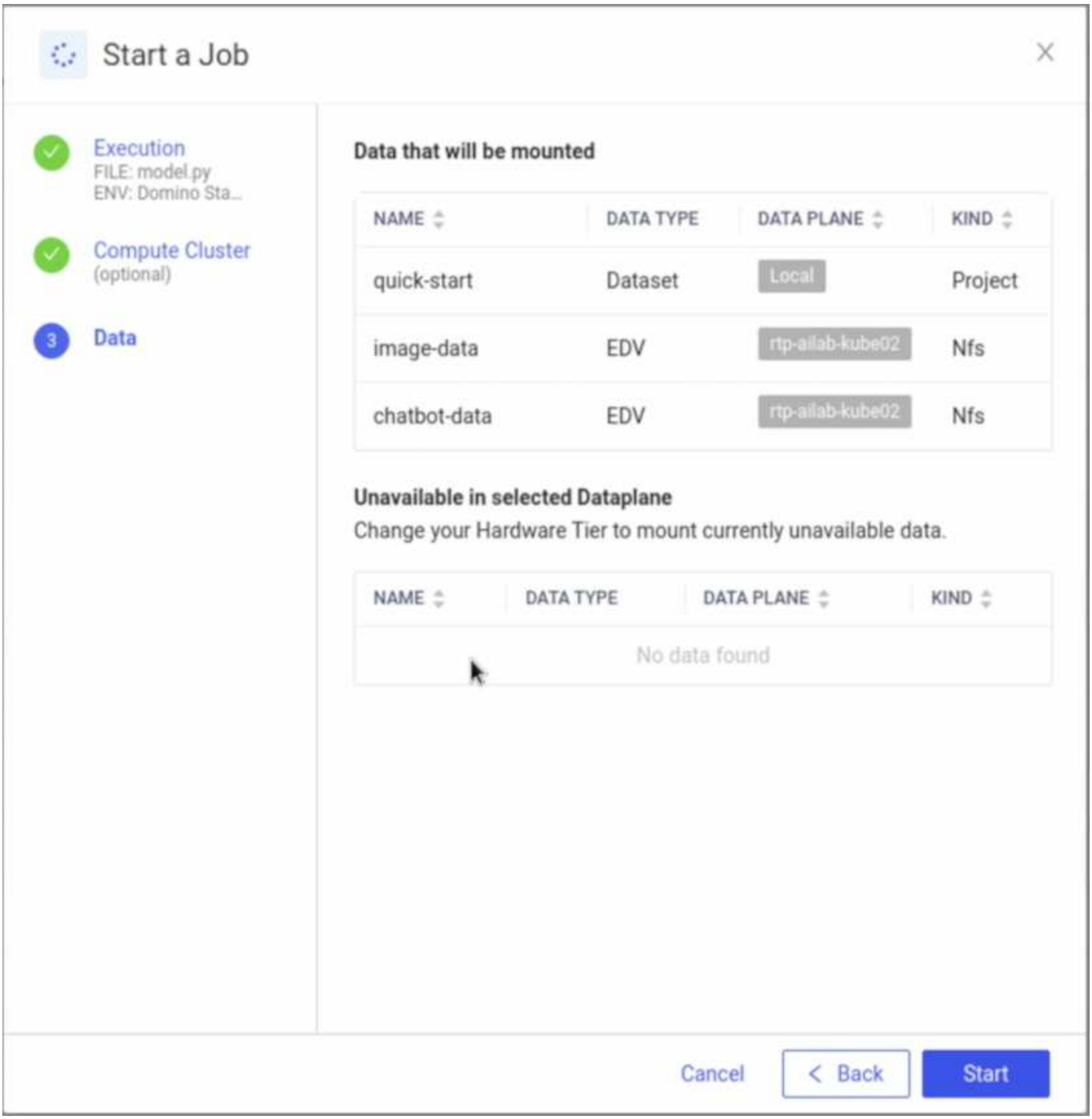
次に、FlexCacheボリュームをDomino MLOpsプラットフォームに公開する必要があります。FlexCacheボリュームをDominoに公開するには、次のセクションの「Astra Tridentでプロビジョニングされていない既存のNFSボリュームを公開する」に記載されている手順に従ってください。 "[既存のNetAppボリュームをDominoに公開するセクション"] （この解決策の）。

これで、次のスクリーンショットに示すように、デスティネーションデータプレーンでジョブやワークスペースを起動するときにFlexCacheボリュームをマウントできるようになります。

FlexCacheボリュームを作成する前に



FlexCacheボリュームを**Domino**に公開した後



オプション**2 -**別のデータプレーンに存在するボリュームをレプリケートする

を使用 "NetApp SnapMirrorデータレプリケーションテクノロジ"では、別のデータプレーンにあるNetAppボリュームのコピーを作成できます。たとえば、オンプレミスのデータプレーンにNetAppボリュームがあり、AWSデータプレーンでそのボリュームにアクセスする必要がある場合は、AWSでボリュームのコピーを作成できます。このセクションでは、別のデータプレーンにあるNetAppボリュームのコピーを作成するために実行する必要があるタスクの概要を説明します。

SnapMirror 関係を作成

まず、ソースボリュームとデスティネーション環境の新しいデスティネーションボリュームの間にSnapMirror関係を作成する必要があります。デスティネーションボリュームは、SnapMirror関係の作成プロセスの一環と

して作成されます。

SnapMirror関係の作成にはBlueXPを使用することを推奨します。BlueXPを使用してSnapMirror関係を作成するには、 "BlueXPのレプリケーションに関するドキュメント"。

BlueXPを使用しない場合は、ONTAPシステムマネージャまたはONTAP CLIを使用してSnapMirror関係を作成できます。System ManagerとのSnapMirror関係を作成するには、 "ONTAP のドキュメント"。ONTAP CLIを使用してSnapMirror関係を作成するには、 "ONTAP のドキュメント"。

このプロセスを自動化するには、 "BlueXP API"、 "ONTAP REST API"または "ONTAP Ansibleコレクション"。

System ManagerはAmazon FSx for NetApp ONTAPでは使用できません。

SnapMirror 関係を解除します

次に、データアクセス用にデスティネーションボリュームをアクティブ化するために、SnapMirror関係を解除する必要があります。最初のレプリケーションが完了するまで待ってから、この手順を実行します。

レプリケーションが完了したかどうかは、BlueXP、ONTAPシステムマネージャ、またはONTAP CLIでミラー状態を確認することで確認できます。レプリケーションが完了すると、ミラー状態は「snapmirrored」になります。

SnapMirror関係の解除にはBlueXPを使用することを推奨します。BlueXPとのSnapMirror関係を解除するには、 "BlueXPのレプリケーションに関するドキュメント"。

BlueXPを使用しない場合は、ONTAP System ManagerまたはONTAP CLIを使用してSnapMirror関係を解除できます。System ManagerとのSnapMirror関係を解除するには、 "ONTAP のドキュメント"。ONTAP CLIとのSnapMirror関係を解除するには、 "ONTAP のドキュメント"。

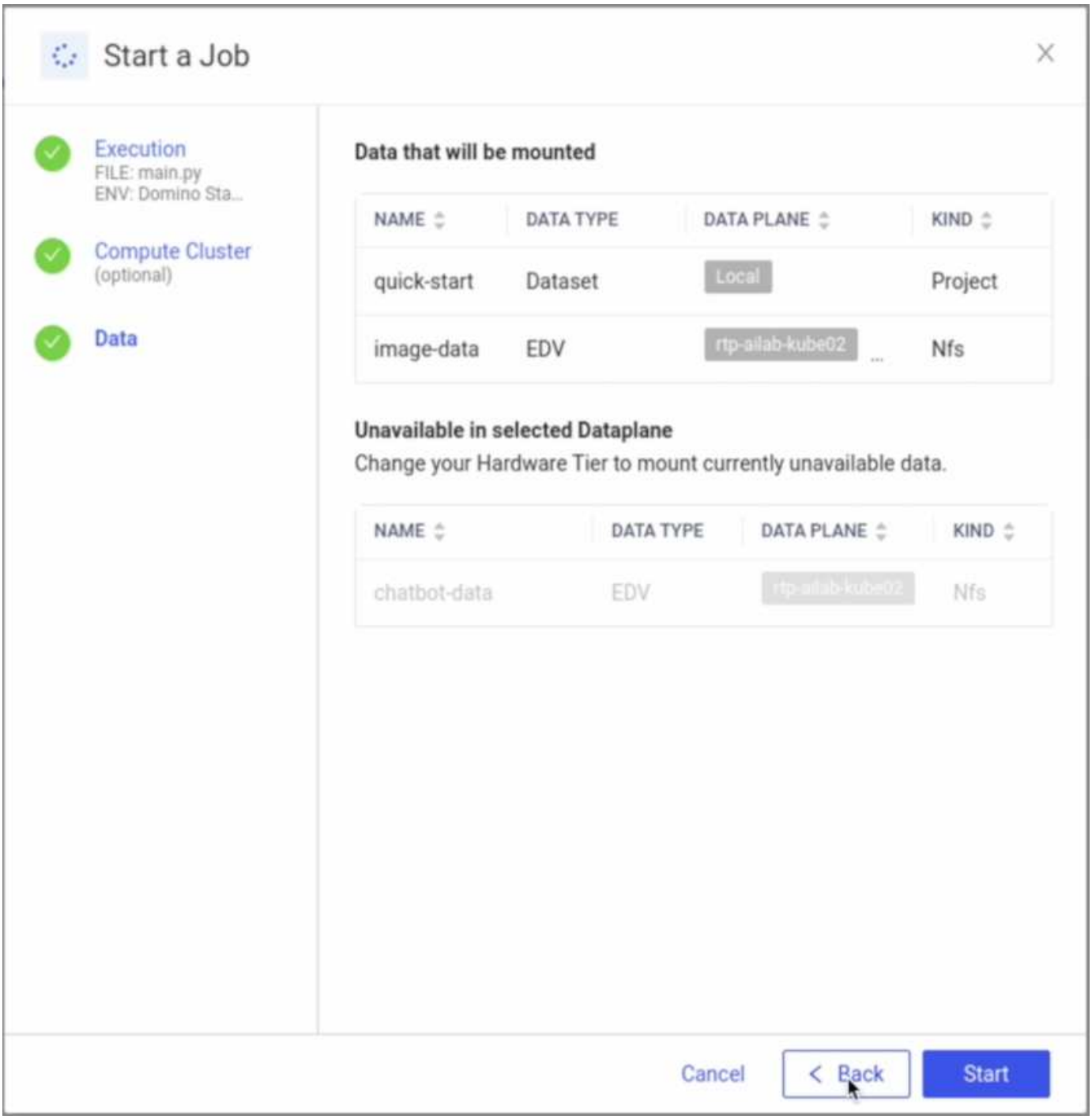
このプロセスを自動化するには、 "BlueXP API"、 "ONTAP REST API"または "ONTAP Ansibleコレクション"。

宛先ボリュームを**Domino**に公開

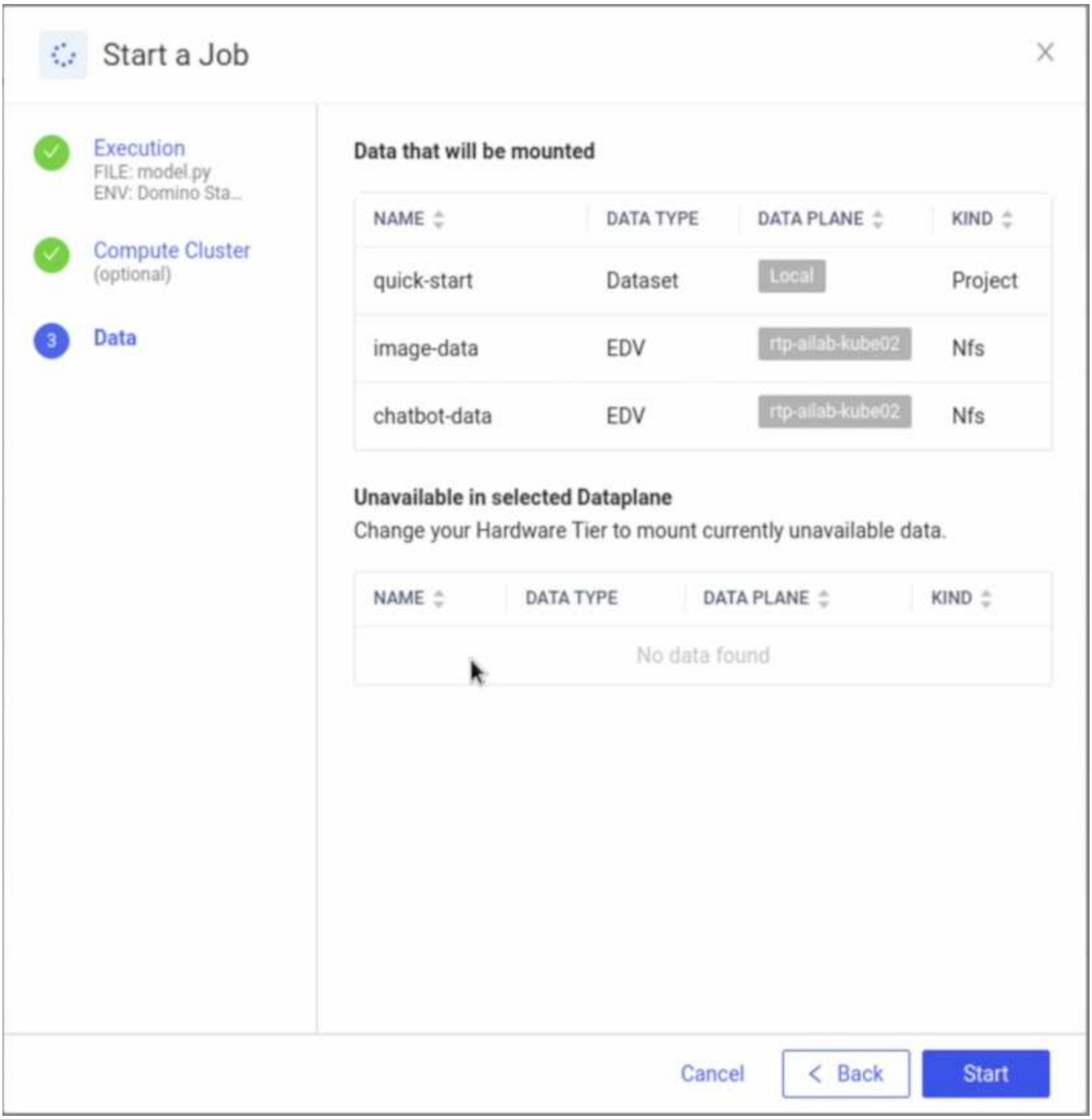
次に、デスティネーションボリュームをDomino MLOpsプラットフォームに公開する必要があります。デスティネーションボリュームをDominoに公開するには、次のセクションの「Astra Tridentでプロビジョニングされていない既存のNFSボリュームを公開する」の手順に従います。 "[既存のNetAppボリュームをDominoに公開するセクション"] （この解決策の）。

これで、次のスクリーンショットに示すように、デスティネーションデータプレーンでジョブやワークスペースを起動するときに、デスティネーションボリュームをマウントできるようになります。

SnapMirror関係を作成する前に



宛先ボリュームを**Domino**に公開した後



追加情報の検索場所

このドキュメントに記載されている情報の詳細については、以下のドキュメントや Web サイトを参照してください。

- Dominoデータラボ

  "https://domino.ai"

- Domino Nexus

  "https://domino.ai/platform/nexus"

- NetApp BlueXP

  "https://bluexp.netapp.com"

- NetApp ONTAP データ管理ソフトウェア

  "https://www.netapp.com/data-management/ontap-data-management-software/"

- NetApp AIソリューション

  "https://www.netapp.com/artificial-intelligence/"

謝辞

- Domino Data Lab Tech Alliances SAディレクターJosh Mineroff氏
- Domino Data LabフィールドCTO Nicholas Jablonski氏
- NetApp解決策アーキテクトPrabu Arjunan氏
- NetAppテクノロジアライアンスパートナーグローバルアライアンスディレクターBrian Young氏

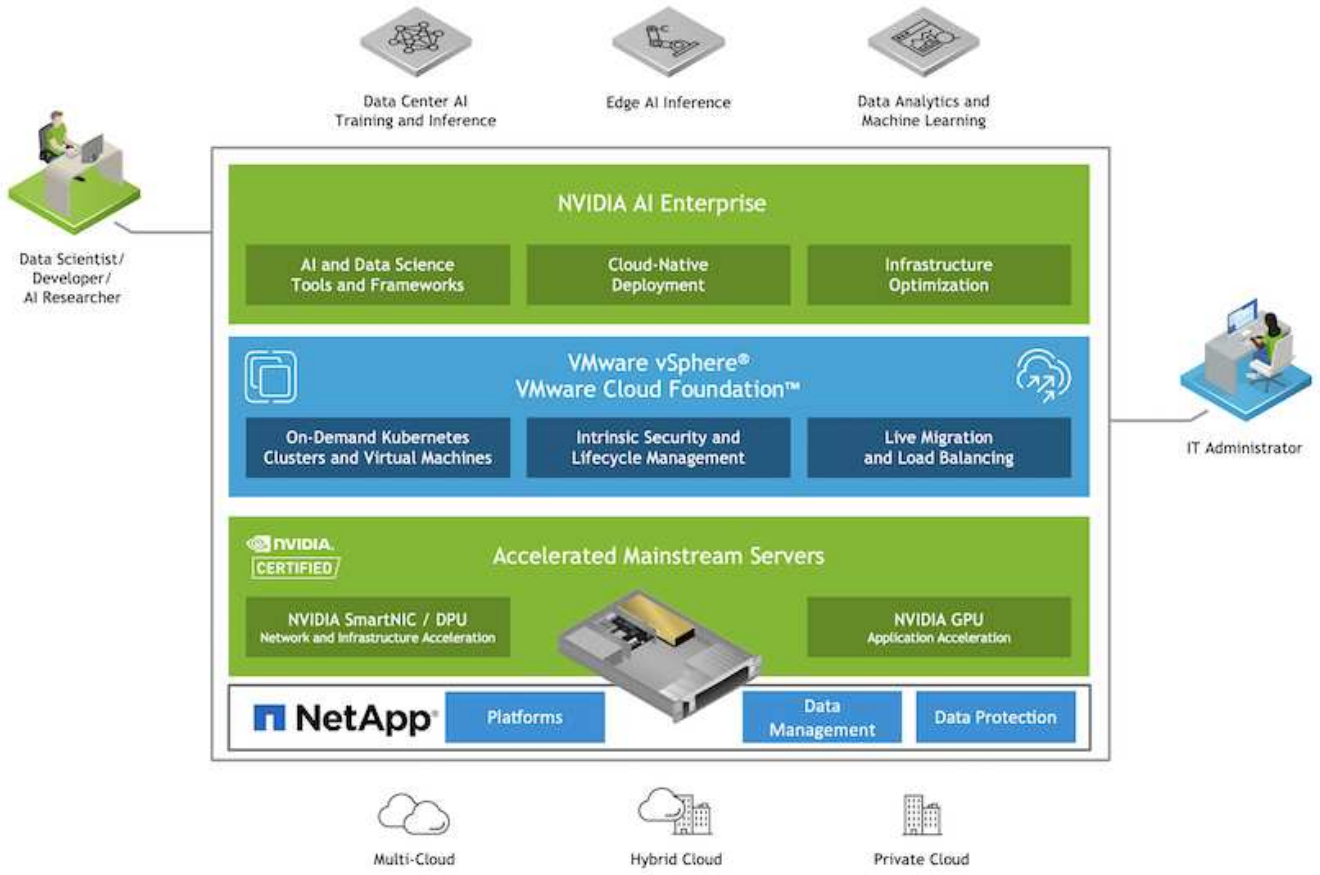
ネットアップと**VMware**による**NVIDIA AI Enterprise**

ネットアップと**VMware**による**NVIDIA AI Enterprise**

ネットアップ、 Mike Oglesby

ITアーキテクトや管理者にとって、AIツールの構築は複雑で、馴染みのない作業です。さらに、多くのAIプラットフォームはエンタープライズ対応ではありません。ネットアップとVMwareを基盤とするNVIDIA AI Enterpriseは、合理化されたエンタープライズクラスのAIアーキテクチャを提供するために開発されました。

NVIDIA AI Enterpriseは、NVIDIA認定システムを搭載したVMware vSphere上で動作するようにNVIDIAによって最適化、認定、サポートされている、AIとデータ分析のためのエンドツーエンドのクラウドネイティブスイートです。AIワークロードの導入、管理、拡張を簡易化し、最新のハイブリッドクラウド環境で容易に実行できます。ネットアップとVMwareを基盤とするNVIDIA AI Enterpriseは、シンプルで使いやすいパッケージで、エンタープライズクラスのAIワークロードとデータ管理を実現します。



テクノロジの概要

NVIDIA AIエンタープライズ

NVIDIA AI Enterpriseは、NVIDIA認定システムを搭載したVMware vSphere上で動作するようにNVIDIAによって最適化、認定、サポートされている、AIとデータ分析のためのエンドツーエンドのクラウドネイティブスイートです。AIワークロードの導入、管理、拡張を簡易化し、最新のハイブリッドクラウド環境で容易に実行できます。

NVIDIA GPU Cloud （NGC）

NVIDIA NGCは、AIソリューションの開発に従事するAI担当者向けに最適化されたGPUソフトウェアのカタログをホスティングします。また、モデルトレーニング用のNVIDIA Baseコマンド、モデルを導入および監視するためのNVIDIA Fleetコマンド、独自のAIソフトウェアに安全にアクセスして管理するためのNGC Private Registryなど、さまざまなAIサービスにアクセスできます。また、NVIDIA AI Enterpriseをご利用のお客様は、NGCポータルからサポートをリクエストできます。

VMware vSphere の場合

VMware vSphereは、CPU、ストレージ、およびネットワークリソースを含む集約されたコンピューティングインフラにデータセンターを変革する、VMwareの仮想化プラットフォームです。vSphereは、これらのインフラを統合運用環境として管理し、管理者がその環境に含まれるデータセンターを管理するためのツールを提供します。

vSphereの中核となるコンポーネントは、ESXiとvCenter Serverの2つです。ESXiは、管理者が仮想マシンと仮想アプライアンスを作成して実行する仮想化プラットフォームです。vCenter Serverは、管理者がネットワ

ークに接続された複数のホストを管理し、ホストリソースをプールするためのサービスです。

NetApp ONTAP

ネットアップが提供する最新世代のストレージ管理ソフトウェアONTAP 9を使用すれば、インフラを最新化し、クラウド対応のデータセンターに移行できます。ONTAP は、業界をリードするデータ管理機能を活用して、データの格納場所に関係なく、単一のツールセットでデータの管理と保護を実現します。エッジ、コア、クラウドなど、必要な場所に自由にデータを移動することもできます。ONTAP 9には、データ管理の簡易化、重要なデータの高速化と保護、ハイブリッドクラウドアーキテクチャ全体で次世代インフラ機能を実現する多数の機能が搭載されています。

データ管理を簡易化

データ管理は、AIアプリケーションの運用やAI / MLデータセットのトレーニングに適切なリソースを使用できるように、エンタープライズIT運用とデータサイエンティストにとって非常に重要です。以下に記載するネットアップテクノロジに関する追加情報 は、この検証の対象外ですが、導入環境によっては関連性がある場合もあります。

ONTAP データ管理ソフトウェアには、運用を合理化および簡易化し、総運用コストを削減するための次の機能が含まれています。

- インラインデータコンパクション、強化された重複排除：データコンパクションはストレージブロック内の無駄なスペースを削減し、重複排除は実効容量を大幅に増やします。この環境データはローカルに格納され、データはクラウドに階層化されます。
- 最小、最大、アダプティブのQuality of Service（AQoS）。きめ細かいサービス品質（QoS）管理機能により、高度に共有された環境で重要なアプリケーションのパフォーマンスレベルを維持できます。
- NetApp FabricPool の略。Amazon Web Services（AWS）、Azure、NetApp StorageGRID ストレージ解決策 など、パブリッククラウドとプライベートクラウドのストレージオプションへコールドデータを自動的に階層化します。FabricPool の詳細については、を参照してください "TR-4598：『FabricPool best bests』"。

データの高速化と保護

ONTAP は、卓越したパフォーマンスとデータ保護を実現し、以下の方法でこれらの機能を拡張します。

- パフォーマンスとレイテンシの低下：ONTAP は、可能なかぎり最小のレイテンシで最高のスループットを提供します。
- データ保護ONTAP には、組み込みのデータ保護機能が用意されており、すべてのプラットフォームを共通の管理機能で管理できます。
- NetApp Volume Encryption（NVE）：ONTAP は、オンボードと外部キー管理の両方をサポートし、ボリュームレベルでのネイティブな暗号化を実現します。
- マルチテナンシーおよび多要素認証ONTAP を使用すると、最高レベルのセキュリティでインフラリソースを共有できます。

将来のニーズにも対応できるインフラ

ONTAP は、次の機能を備えており、要件が厳しく、絶えず変化するビジネスニーズに対応できます。

- シームレスな拡張とノンストップオペレーションONTAP を使用すると、既存のコントローラとスケールアウトクラスタに無停止で容量を追加できます。NVMe や 32Gb FC などの最新テクノロジへのアップグレードも、コストのかかるデータ移行やシステム停止を行わずに実行できます。

- クラウドへの接続：ONTAP は、すべてのパブリッククラウドでSoftware-Defined Storage（ONTAP Select ）とクラウドネイティブインスタンス（NetApp Cloud Volumes Service ）のオプションを選択できる、マルチクラウドに対応した最もクラウド対応のストレージ管理ソフトウェアです。
- 新しいアプリケーションとの統合：ONTAP は、既存のエンタープライズアプリケーションをサポートするインフラを使用して、自律走行車、スマートシティ、インダストリー4.0などの次世代プラットフォームやアプリケーション向けにエンタープライズクラスのデータサービスを提供します。

NetApp DataOps ツールキット

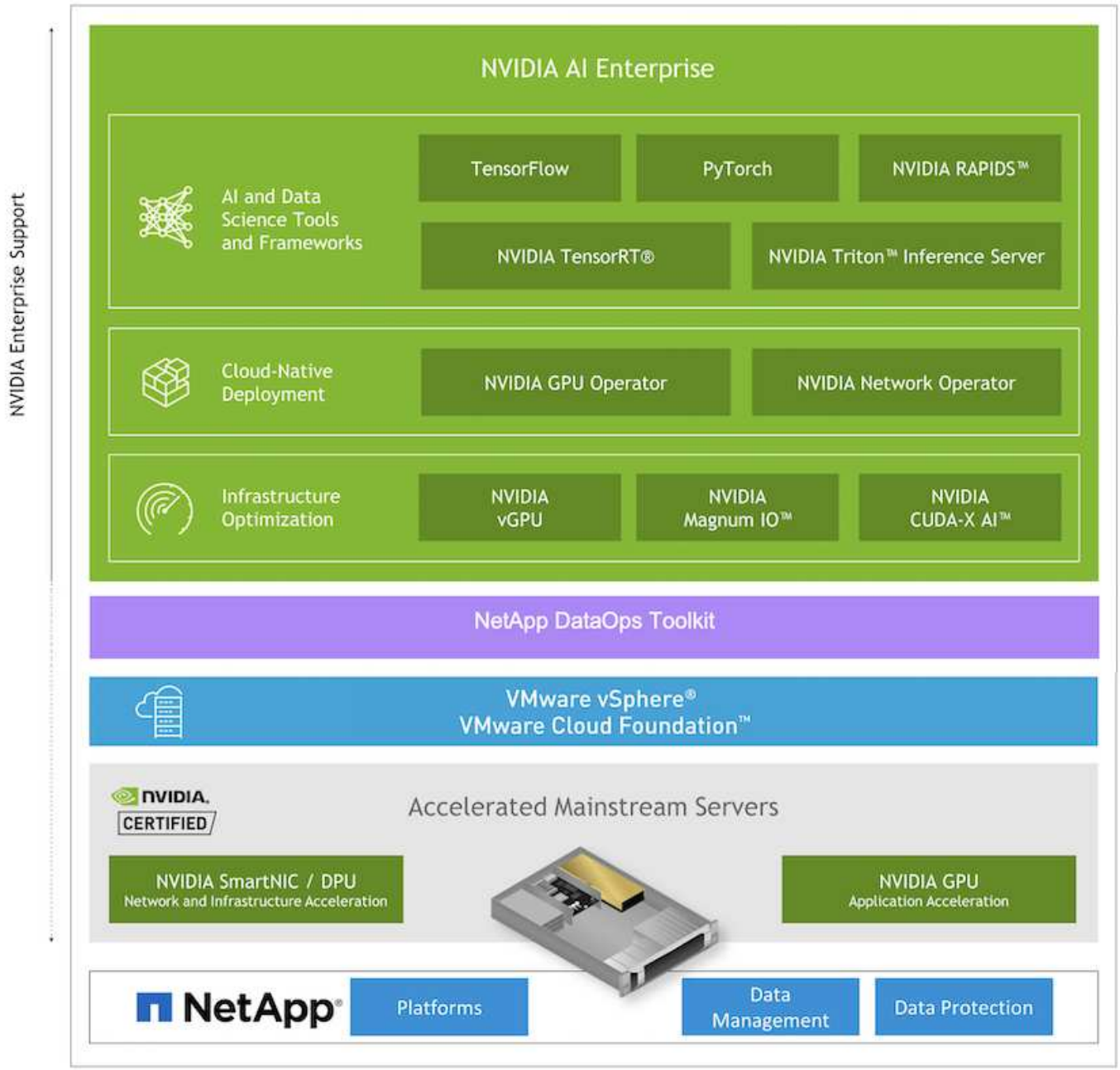
NetApp DataOpsツールキットは、高性能なスケールアウトネットアップストレージを基盤とする開発/トレーニング用ワークスペースと推論サーバの管理を簡易化するPythonベースのツールです。主な機能は次のとおりです。

- ハイパフォーマンスでスケールアウト可能なネットアップストレージを基盤とする、大容量のJupyterLabワークスペースを迅速にプロビジョニングできます。
- エンタープライズクラスのネットアップストレージを基盤とする新しいNVIDIA Triton Inference Serverインスタンスを迅速にプロビジョニング
- 実験的または迅速な反復を可能にするために、大容量のJupyterLabワークスペースをほぼインスタンス化してクローニングします。
- 大容量JupyterLabワークスペースのスナップショットをほぼインスタンスで保存し、バックアップやトレーサビリティ/ベースライン設定を実現します。
- ほぼインスタンス化することなく、大容量でハイパフォーマンスなデータボリュームをプロビジョニング、クローニング、およびSnapshot作成する。

アーキテクチャ

この解決策 は、ネットアップ、VMware、NVIDIA認定システムを使用した、実績のある馴染みのあるアーキテクチャを基盤としています。詳細については、次の表を参照してください。

コンポーネント	詳細
AIとデータ分析ソフトウェア	"NVIDIA AI Enterprise for VMware"
仮想化プラットフォーム	"VMware vSphere の場合"
コンピューティングプラットフォーム	"NVIDIA認定システム"
Data Management Platformの略	"NetApp ONTAP"



初期セットアップ

このセクションでは、ネットアップとVMwareでNVIDIA AI Enterpriseを活用するために必要な初期セットアップタスクについて説明します。

前提条件

ここで説明する手順を実行する前に、VMware vSphereとNetApp ONTAP が導入済みであることを前提としています。を参照してください "NVIDIA AI Enterprise製品サポートマトリックス" サポートされているvSphereのバージョンの詳細については、を参照を参照してください "ネットアップとVMware解決策 のドキュメント" NetApp ONTAP を使用したVMware vSphereの導入の詳細については、を参照してください。

NVIDIA AI Enterprise Host Softwareをインストールします

NVIDIA AI Entrpriseホストソフトウェアをインストールするには、のセクション1~4に記載されている手順に従います "NVIDIA AI Enterpriseクイックスタートガイド"。

NVIDIA NGCソフトウェアを利用

このセクションでは、NVIDIA AI Enterprise環境でNVIDIA NGCエンタープライズソフトウェアを利用するために実行する必要があるタスクについて説明します。

セットアップ（Setup）

このセクションでは、NVIDIA AI Enterprise環境でNVIDIA NGCエンタープライズソフトウェアを利用するために実行する必要がある初期セットアップタスクについて説明します。

前提条件

このセクションで説明する手順を実行する前に、に記載されている手順に従ってNVIDIA AI Entrpriseホストソフトウェアがすでに導入されていることを前提としています "初期セットアップ" ページ

vGPUを使用してUbuntuゲストVMを作成します

まず、vGPUを使用してUbuntu 20.04ゲストVMを作成する必要があります。vGPUを使用してUbuntu 20.04ゲストVMを作成する場合は、の手順に従います "NVIDIA AI Enterprise導入ガイド"。

NVIDIA Guest Softwareをダウンロードしてインストールします

次に、前の手順で作成したゲストVMに必要なNVIDIAゲストソフトウェアをインストールします。必要なNVIDIAゲストソフトウェアをゲストVMにダウンロードしてインストールするには、のセクション5.1-5.4に記載されている手順に従います "NVIDIA AI Enterpriseクイックスタートガイド"。

> セクション5.4で説明した検証タスクを実行するときは、ガイドの作成後にCUDAコンテナイメージが更新されているため、別のCUDAコンテナイメージバージョンタグを使用する必要がある場合があります。今回の検証では「nvidia / CUDA：11.0.3-base-ubuntu20.04」を使用しました。

AI /分析フレームワークコンテナをダウンロード

次に、NVIDIA NGCからAIまたは分析フレームワークのコンテナイメージをダウンロードして、ゲストVM内で利用できるようにする必要があります。ゲストVM内でフレームワークコンテナをダウンロードするには、の手順に従います "NVIDIA AI Enterprise導入ガイド"。

NetApp DataOpsツールキットをインストールして設定します

次に、ゲストVM内で従来の環境にNetApp DataOpsツールキットをインストールする必要があります。NetApp DataOpsツールキットを使用すると、ONTAP システム上のスケールアウトデータボリュームをゲストVM内の端末から直接管理できます。ゲストVMにNetApp DataOpsツールキットをインストールするには、次のタスクを実行します。

1. pipをインストールします。

```
$ sudo apt update
$ sudo apt install python3-pip
$ python3 -m pip install netapp-dataops-traditional
```

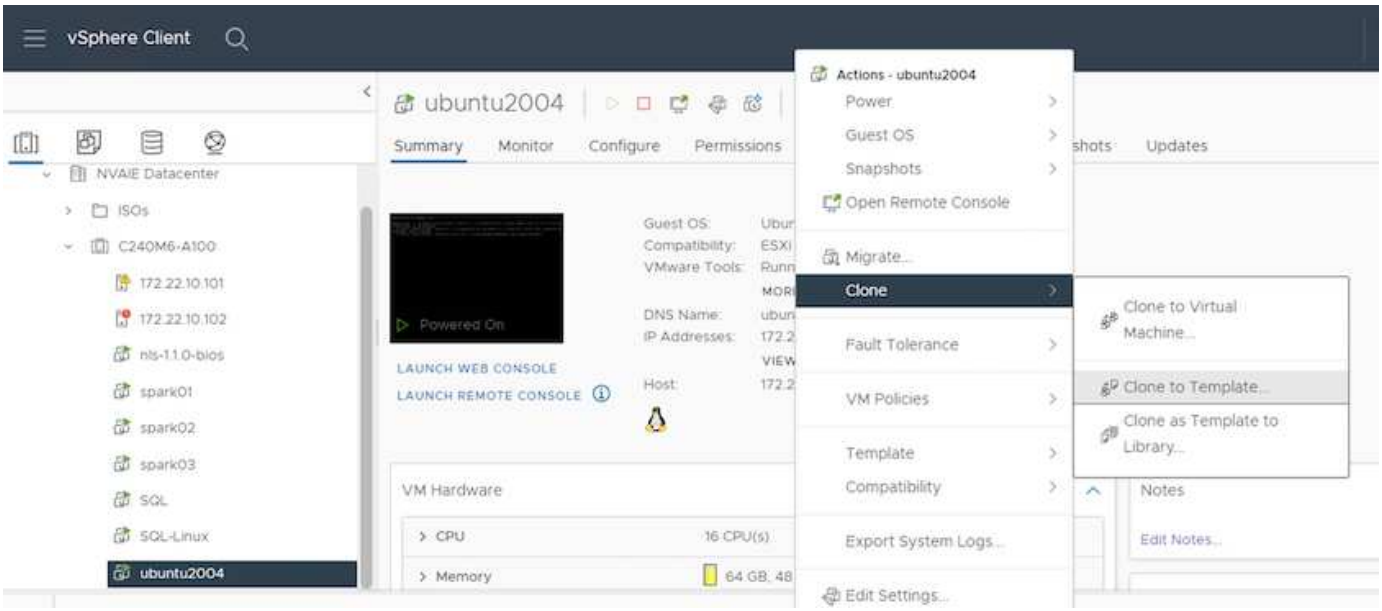
2. ゲストVM端末からログアウトし、再度ログインします。
3. NetApp DataOpsツールキットを設定する。この手順を完了するには、ONTAP システムのAPIアクセスの詳細が必要です。これらはストレージ管理者から入手する必要があります。

```
$ netapp_dataops_cli.py config

Enter ONTAP management LIF hostname or IP address (Recommendation: Use
SVM management interface): 172.22.10.10
Enter SVM (Storage VM) name: NVAIE-client
Enter SVM NFS data LIF hostname or IP address: 172.22.13.151
Enter default volume type to use when creating new volumes
(flexgroup/flexvol) [flexgroup]:
Enter export policy to use by default when creating new volumes
[default]:
Enter snapshot policy to use by default when creating new volumes
[none]:
Enter unix filesystem user id (uid) to apply by default when creating
new volumes (ex. '0' for root user) [0]:
Enter unix filesystem group id (gid) to apply by default when creating
new volumes (ex. '0' for root group) [0]:
Enter unix filesystem permissions to apply by default when creating new
volumes (ex. '0777' for full read/write permissions for all users and
groups) [0777]:
Enter aggregate to use by default when creating new FlexVol volumes:
aff_a400_01_NVME_SSD_1
Enter ONTAP API username (Recommendation: Use SVM account): admin
Enter ONTAP API password (Recommendation: Use SVM account):
Verify SSL certificate when calling ONTAP API (true/false): false
Do you intend to use this toolkit to trigger BlueXP Copy and Sync
operations? (yes/no): no
Do you intend to use this toolkit to push/pull from S3? (yes/no): no
Created config file: '/home/user/.netapp_dataops/config.json'.
```

ゲスト**VM**テンプレートを作成します

最後に、ゲストVMに基づいてVMテンプレートを作成する必要があります。このテンプレートを使用すると、NVIDIA NGCソフトウェアを使用するゲストVMをすばやく作成できます。

ゲストVMに基づいてVMテンプレートを作成するには、VMware vSphereにログインし、ゲストVM名をクリックして「Clone」を選択し、「Clone to Template…」を選択して、ウィザードに従います。



使用例**- TensorFlow**トレーニングジョブ

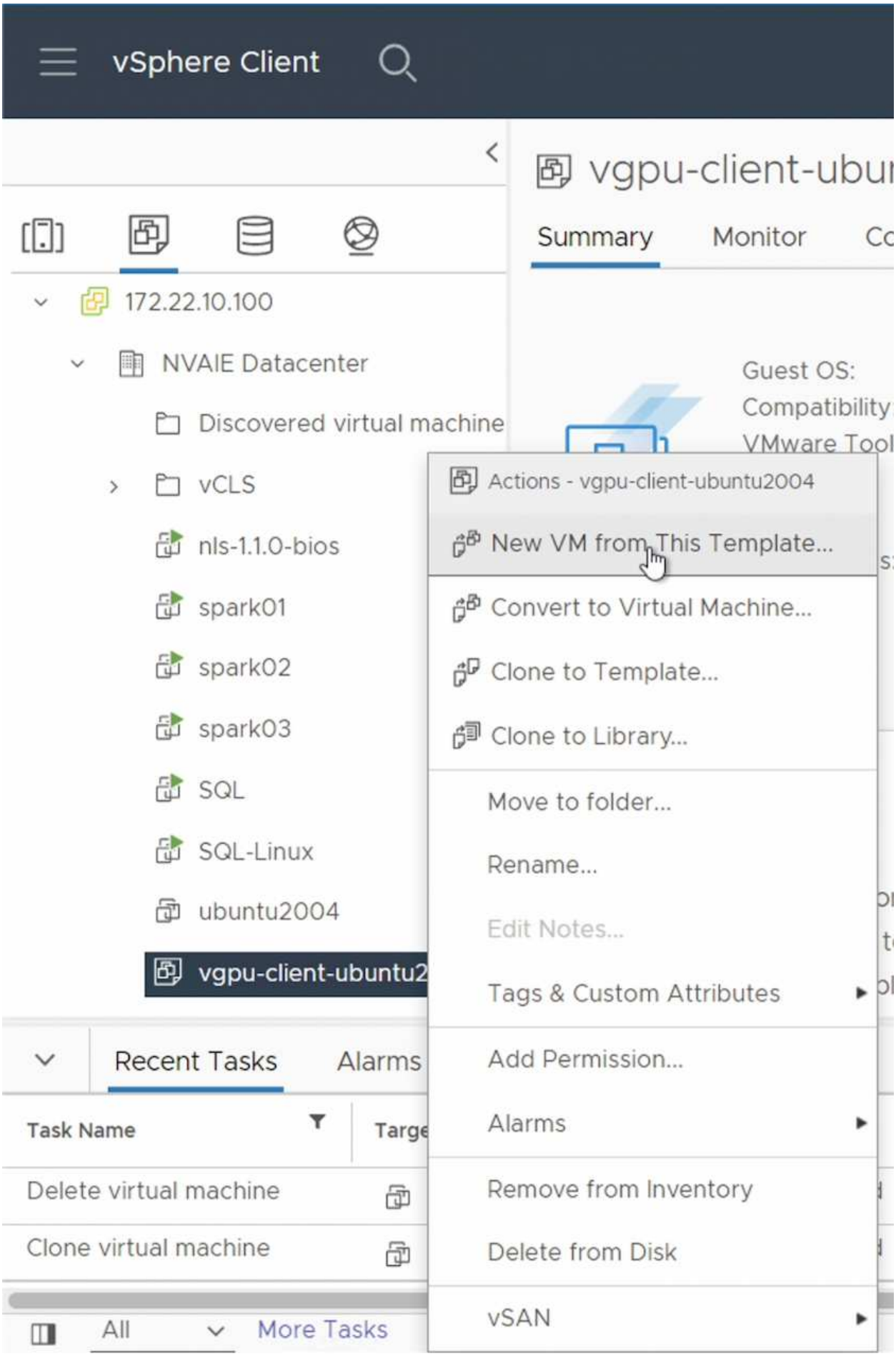
このセクションでは、NVIDIA AI Enterprise環境内でTensorFlowトレーニングジョブを実行するために実行する必要があるタスクについて説明します。

前提条件

ここで説明する手順を実行する前に、に記載されている手順に従ってゲストVMテンプレートを作成済みであることを前提としています "セットアップ（Setup）" ページ

テンプレートからゲスト**VM**を作成します

最初に、前のセクションで作成したテンプレートから新しいゲストVMを作成する必要があります。テンプレートから新しいゲストVMを作成するには、VMware vSphereにログインし、テンプレート名を右クリックして「このテンプレートからVMを新規作成...」を選択し、ウィザードに従います。


vSphere Client
vgpu-client-ubu
Summary
Monitor
172.22.10.100
NVAIE Datacenter
Discovered virtual machine
vCLS
nls-1.1.0-bios
spark01
spark02
spark03
SQL
SQL-Linux
ubuntu2004
vgpu-client-ubuntu2
Guest OS:
Compatibility
VMware Tool
Actions - vgpu-client-ubuntu2004
New VM from This Template...
Convert to Virtual Machine...
Clone to Template...
Clone to Library...
Move to folder...
Rename...
Edit Notes...
Tags & Custom Attributes
Add Permission...
Alarms
Remove from Inventory
Delete from Disk
vSAN
Recent Tasks
Alarms
Task Name
Delete virtual machine
Clone virtual machine
All
More Tasks

データボリュームを作成してマウント

次に、トレーニングデータセットを格納する新しいデータボリュームを作成する必要があります。NetApp DataOpsツールキットを使用して、新しいデータボリュームを簡単に作成できます。次のコマンド例は、「ImageNet」という名前のボリュームを作成し、容量を2TBにしています。

```
$ netapp_dataops_cli.py create vol -n imagenet -s 2TB
```

データボリュームにデータを入力する前に、ゲストVM内でデータボリュームをマウントする必要があります。NetApp DataOpsツールキットを使用して、データボリュームを簡単にマウントできます。次のコマンド例は、前の手順で作成したボリュームをアンマウントしています。

```
$ sudo -E netapp_dataops_cli.py mount vol -n imagenet -m ~/imagenet
```

データボリュームの取り込み

新しいボリュームのプロビジョニングとマウントが完了したら、トレーニングデータセットをソースの場所から取得して、新しいボリュームに配置できます。通常はS3またはHadoopのデータレイクからデータを取得する必要があり、場合によってはデータエンジニアの支援も必要になります。

TensorFlowトレーニングジョブを実行する

これで、TensorFlowトレーニングジョブを実行する準備が整いました。TensorFlowトレーニングジョブを実行するには、次のタスクを実行します。

1. NVIDIA NGC Enterprise TensorFlowコンテナイメージを取得します。

   ```
   $ sudo docker pull nvcr.io/nvaie/tensorflow-2-1:22.05-tf1-nvaie-2.1-py3
   ```

2. NVIDIA NGCエンタープライズTensorFlowコンテナのインスタンスを起動します。「-v」オプションを使用して、データボリュームをコンテナに接続します。

   ```
   $ sudo docker run --gpus all -v ~/imagenet:/imagenet -it --rm
   nvcr.io/nvaie/tensorflow-2-1:22.05-tf1-nvaie-2.1-py3
   ```

3. コンテナ内でTensorFlowトレーニングプログラムを実行します。次のコマンド例は、コンテナイメージに含まれるResNet-50トレーニングプログラムの実行例を示しています。

   ```
   $ python ./nvidia-examples/cnn/resnet.py --layers 50 -b 64 -i 200 -u
   batch --precision fp16 --data_dir /imagenet/data
   ```

追加情報の検索場所

このドキュメントに記載されている情報の詳細については、以下のドキュメントや Web サイトを参照してください。

- NetApp ONTAP データ管理ソフトウェア—ONTAP 情報ライブラリ

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

- NetApp DataOps ツールキット

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

- NVIDIA AI Enterprise with VMware

  https://www.nvidia.com/en-us/data-center/products/ai-enterprise/vmware/^]

謝辞

- Bobby Oommen、Sr.ネットアップ、マネージャー
- ネットアップ、システム管理者、Ramesh Isaac氏
- ネットアップ、テクニカルマーケティングエンジニア、Roney Daniel氏

TR-4851：自動運転ワークロード向けのネットアップ**StorageGRID** データレイク**-**解決策 設計

ネットアップ、David Arnette

TR-4851は、機械学習（ML）とディープラーニング（DL）のソフトウェア開発のためのデータリポジトリおよび管理システムとして、NetApp StorageGRID オブジェクトストレージを使用する方法を示しています。このホワイトペーパーでは、自律走行車ソフトウェア開発におけるデータフローと要件、およびデータライフサイクルを合理化するStorageGRID 機能について説明します。この 解決策 環境 は、MLおよびDL開発プロセスで一般的なマルチステージのデータパイプラインワークフローです。

"TR-4851：自動運転ワークロード向けのネットアップStorageGRID データレイク-解決策 設計"

NetApp AI コントロールプレーン

TR-4798 ：『 NetApp AI Control Plane 』

ネットアップ、 Mike Oglesby

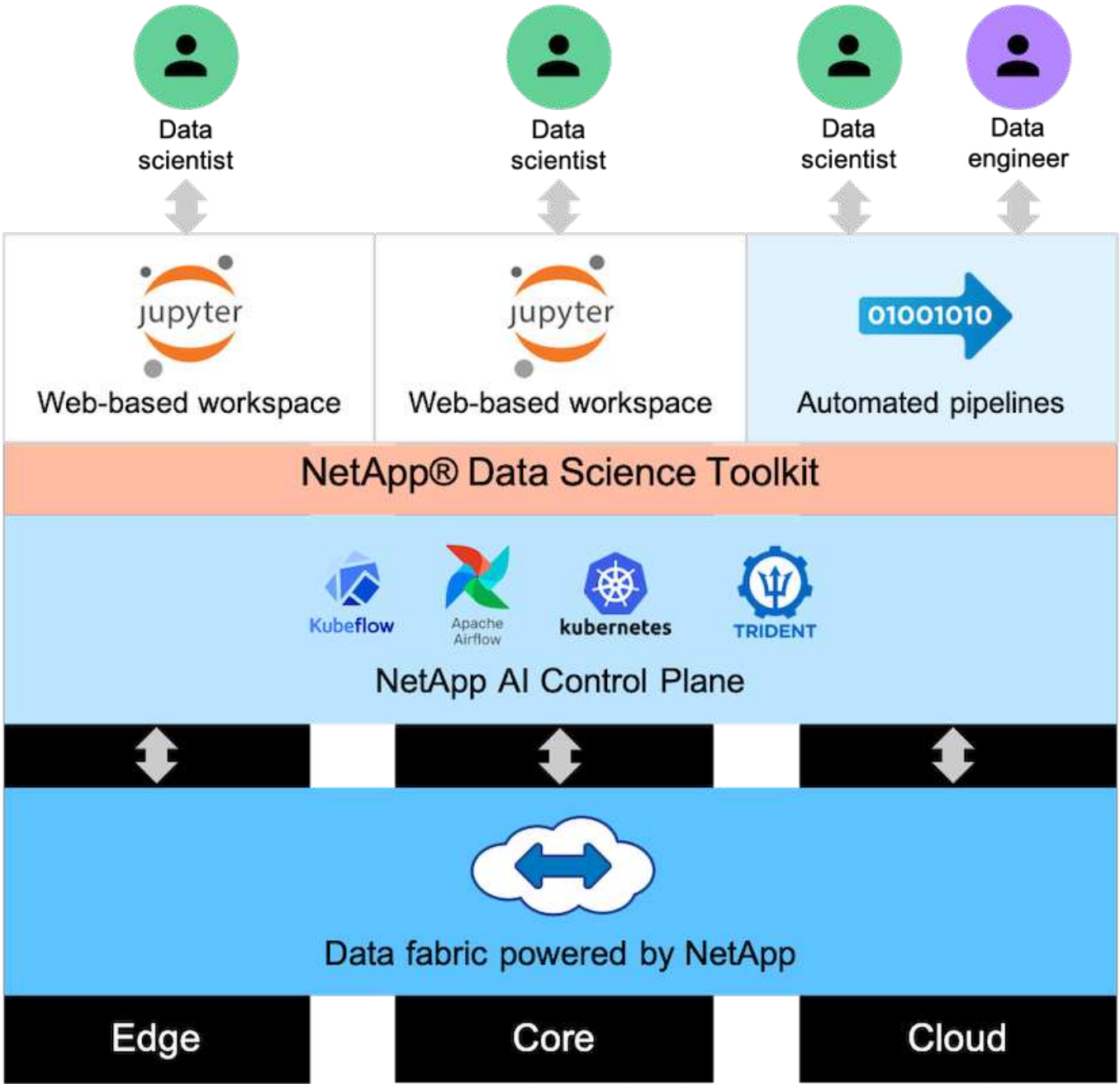
あらゆる規模の企業や組織が、多くの業界で、人工知能（ AI ）、機械学習（ ML ）、ディープラーニング（ DL ）を導入して、現実世界の問題を解決し、革新的な製品やサービスを提供し、競争が激化する市場で優位に立つことになりつつあります。AI 、 ML 、 DL の利用が増えるにつれ、ワークロードの拡張性やデータの可用性など、多くの課題に直面しています。このドキュメントでは、ネットアップのデータ管理機能と一般的なオ

ープンソースのツールやフレームワークとのペアリングを行う解決策である NetApp AI コントロールプレーンを使用して、これらの課題に対処する方法について説明します。

このレポートでは、データネームスペースを迅速にクローニングする方法について説明します。また、サイトやリージョン間でデータをシームレスにレプリケートし、統合された AI / ML / DL データパイプラインを構築する方法も示します。さらに、トレーサビリティとバージョン管理のためにデータやモデルベースラインをほぼ瞬時に作成する AI 、 ML 、 DL トレーニングワークフローの定義と実装についても説明します。この解決策を使用すると、すべてのモデルトレーニングを、モデルのトレーニングや検証に使用したデータセットまでトレースできます。最後に、このドキュメントでは、 Jupyter Notebook ワークスペースを、大規模なデータセットにすばやくプロビジョニングする方法を説明します。

注：同じデータセットへの共有アクセスを必要とする多数の GPU サーバを対象とした大規模な HPC スタイルの分散トレーニング、または並列ファイルシステムを必要とする場合は、チェックアウトしてください "TR-4890"。本テクニカルレポートでは、この情報を記載する方法について説明します "ネットアップのフルサポートの並列ファイルシステム解決策 BeeGFS" ネットアップの AI コントロールプレーンの一部として提供されます。この解決策は、数台の NVIDIA DGX A100 システムで構成される 140 ノードの SuperPOD まで拡張できるように設計されています。

ネットアップの AI コントロールプレーンは、データサイエンティストやデータエンジニアをターゲットとするため、ネットアップや ONTAP ® に関する専門知識は最小限に抑えられます。この解決策を使用すると、シンプルで使い慣れたツールやインターフェイスを使用してデータ管理機能を実行できます。ネットアップストレージがすでにある環境では、ネットアップの AI コントロールプレーンを今すぐテストできます。解決策のテストドライブを希望していても、ネットアップストレージがない場合は、にアクセスしてください "cloud.netapp.com"クラウドベースのネットアップストレージ解決策を使用すれば、数分で稼働を開始できます。次の図に、解決策を視覚的に示します。



コンセプトとコンポーネント

人工知能

AI とは、人間の心の認識機能を模倣するためにコンピュータが訓練されているコンピュータ科学分野です。AI 開発者は、人間に似た方法、または人間に比べて優れた方法で、コンピュータをトレーニングして問題を解決します。ディープラーニングと機械学習は AI のサブフィールドです。組織は、重要なビジネスニーズに対応するために、 AI 、 ML 、 DL を導入する傾向に迫られています。次に例を示します。

- 未知のビジネスに大量のデータを分析しています 分析
- 自然言語処理を使用して顧客と直接やり取りする
- さまざまなビジネスプロセスと機能を自動化します。

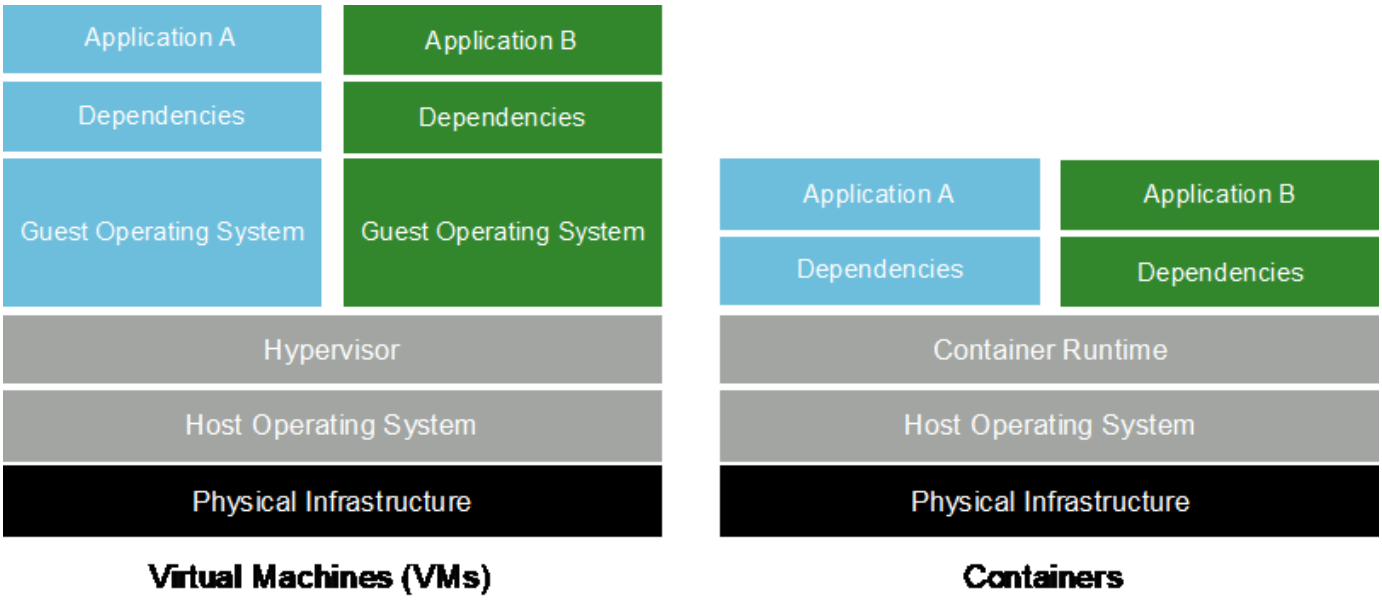
最新の AI トレーニングと推論のワークロードには、超並列処理機能が必要です。そのため、 GPU の並列処

理機能は汎用 CPU よりもはるかに優れているため、 GPU を使用した AI 処理も増えています。

コンテナ

コンテナは、共有ホストオペレーティングシステムカーネル上で実行される独立したユーザスペースインスタンスです。コンテナの採用が急速に増加しています。コンテナは、仮想マシン（ VM ）が提供するものと同じアプリケーションのサンドボックス化のメリットの多くを提供します。ただし、 VM が依存するハイパーバイザーレイヤとゲストオペレーティングシステムレイヤが排除されているため、コンテナの軽量化が大幅に向上しています。次の図に、仮想マシンとコンテナを視覚的に示します。

コンテナを使用すると、アプリケーションの依存関係や実行時間などをアプリケーションで直接効率的にパッケージングできます。最も一般的に使用されるコンテナパッケージ形式は Docker コンテナです。Docker コンテナ形式でコンテナ化されたアプリケーションは、 Docker コンテナを実行できる任意のマシンで実行できます。これは、アプリケーションの依存関係がマシンに存在しない場合でも当てはまります。これは、すべての依存関係がコンテナ自体にパッケージ化されているためです。詳細については、を参照してください "Docker Web サイト"。



Kubernetes

Kubernetes は、 Google が当初設計した、オープンソースの分散型コンテナオーケストレーションプラットフォームであり、 Cloud Native Computing Foundation （ CNCF ）によって管理されています。Kubernetes を使用すると、コンテナ化されたアプリケーションの導入、管理、拡張の機能を自動化できます。近年、Kubernetes は主要なコンテナオーケストレーションプラットフォームとして登場しています。他のコンテナパッケージ化形式や実行時間もサポートされていますが、 Kubernetes は Docker コンテナ用のオーケストレーションシステムとして最もよく使用されます。詳細については、を参照してください "Kubernetes Web サイト"。

NetApp Trident

Trident は、ネットアップが開発および管理しているオープンソースのストレージオーケストレーションツールで、 Kubernetes ワークロード向けの永続的ストレージの作成、管理、使用を大幅に簡易化します。Trident は Kubernetes ネイティブのアプリケーションであり、 Kubernetes クラスタ内で直接実行されます。Trident を使用すると、 Kubernetes のユーザ（開発者、データサイエンティスト、 Kubernetes 管理者など）は、使い慣れた標準的な Kubernetes 形式で永続ストレージボリュームを作成、管理、操作できます。同時に、ネットアップの高度なデータ管理機能と、ネットアップテクノロジを基盤とするデータファブリックを活用できま

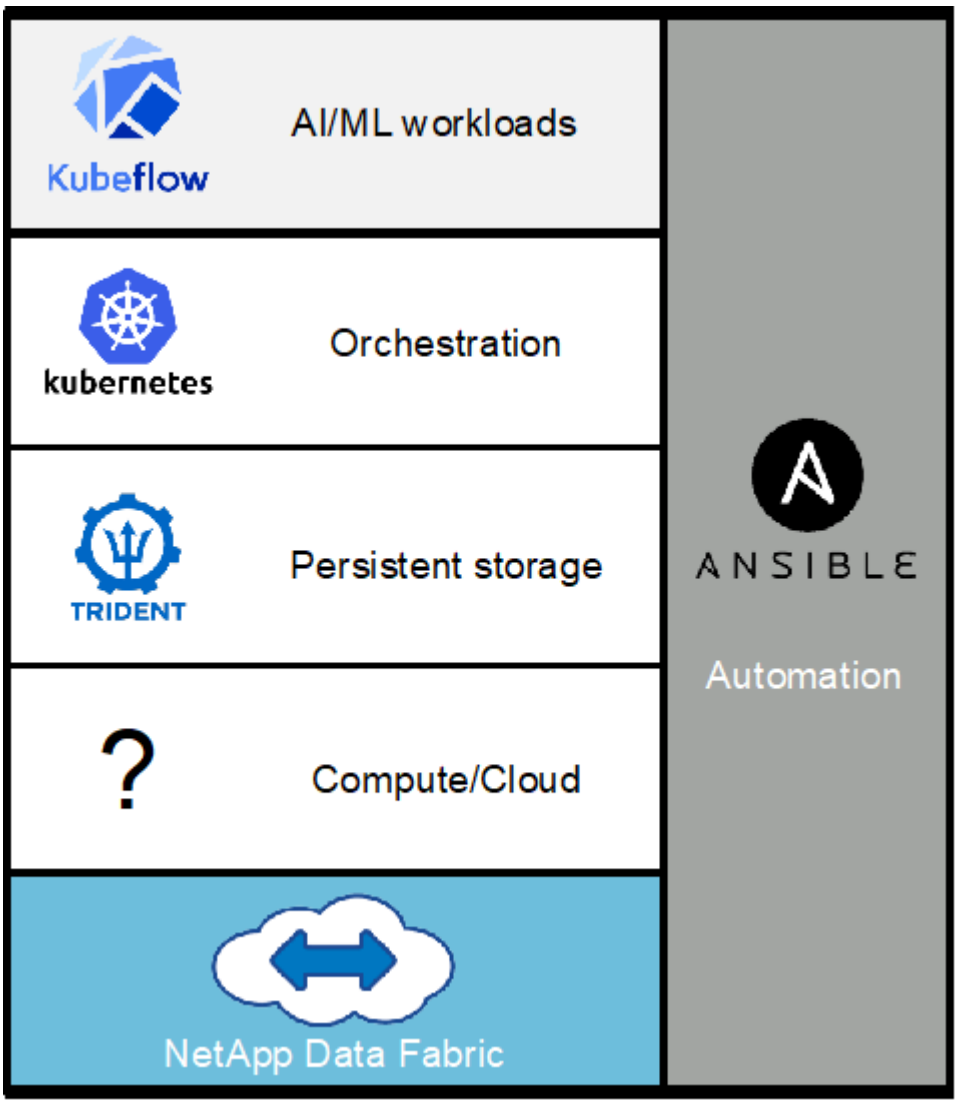
す。Trident は、複雑な永続的ストレージを抽象化して、消費を簡易化します。詳細については、を参照してください "Trident の Web サイト"。

NVIDIA DeepOps のことです

DeepOps は NVIDIA が開発したオープンソースプロジェクトです。 Ansible を使用することで、ベストプラクティスに従って GPU サーバクラスタの導入を自動化できます。DeepOps はモジュール方式であり、さまざまな導入タスクに使用できます。このドキュメントとこの検証の演習では、 DeepOps を使用して、 GPU サーバワーカーノードで構成される Kubernetes クラスタを導入します。詳細については、を参照してください "DeepOps の Web サイト"。

クビフロー

Kubeflow は Kubernetes 向けのオープンソースの AI / ML ツールキットで、 Google が開発したものです。Kubeflow プロジェクトでは、 Kubernetes での AI ワークフローと ML ワークフローの導入を、シンプル、ポータブル、拡張性に優れた方法で実施します。Kubeflow は Kubernetes の複雑さを抽象化し、データサイエンティストがデータサイエンスのベストプラクティスに集中できるようにします。表示については、次の図を参照してください。Kubernetes で企業の IT 部門の標準化が進むにつれて、 Kubeflow は大きな牽引力を発揮してきました。詳細については、を参照してください "Kubeflow の Web サイト"。



Kubeflow パイプライン

Kubeflow Pipelines は Kubeflow の主要コンポーネントです。Kubeflow Pipelines は、移植性と拡張性に優れた AI および ML ワークフローを定義、導入するためのプラットフォームと標準です。詳細については、を参照してください "Kubeflow の公式ドキュメント"。

Jupyter Notebook Server の 2 つのツールを使用

Jupyter Notebook Server はオープンソースの Web アプリケーションで、データサイエンティストは Jupyter Notebook と呼ばれる Wiki 形式のドキュメントを作成できます。このドキュメントには、ライブコードと説明的なテストが含まれています。Jupyter Notebook は、 AI プロジェクトと ML プロジェクトを文書化、保存、共有する手段として、 AI と ML のコミュニティで広く使用されています。Kubeflow を使用すると、Kubernetes での Jupyter Notebook Server のプロビジョニングと導入が簡単になります。Jupyter Notebook の詳細については、を参照してください "Jupyter のウェブサイト"。Kubeflow のコンテキスト内の Jupyter Notebook の詳細については、を参照してください "Kubeflow の公式ドキュメント"。

Apache の通気

Apache Airflow は、複雑なエンタープライズワークフローのプログラムによるオーサリング、スケジューリング、監視を可能にするオープンソースのワークフロー管理プラットフォームです。ETL やデータパイプラインのワークフローを自動化する目的でよく使用されますが、こうした種類のワークフローに限定されるわけではありません。Airflow プロジェクトは Airbnb が開始しましたが、業界で非常に人気があり、現在は Apache Software Foundation の後援を受けています。空気の流れは Python で書かれており、 Python スクリプトを使用して空気の流れが作られています。また、空気の流れは、「コードとしての設定」という原則に基づいて設計されています。 現在、多くの企業のエアフローユーザが Kubernetes の上で通気を実行しています。

ダイレクト非周期グラフ（ DAG ）

エアーフローでは、ワークフローは Directed Acyclic Graphs （ DAG ）と呼ばれます。DAG は、 DAG の定義に応じて、順番に実行されるタスク、並列タスク、またはその組み合わせで実行されるタスクで構成されます。エアーフロースケジューラは、 DAG 定義で指定されているタスクレベルの依存関係を維持しながら、一連のワーカーに対して個々のタスクを実行します。DAG は Python スクリプトを使用して定義および作成されます。

NetApp ONTAP 9.

NetApp ONTAP 9 はネットアップが提供する最新世代のストレージ管理ソフトウェアです。お客様のような企業がインフラを刷新し、クラウド対応のデータセンターに移行できるようにします。業界をリードするデータ管理機能を備えた ONTAP では、データの格納場所に関係なく、単一のツールセットでデータの管理と保護を行うことができます。エッジ、コア、クラウドなど、必要な場所に自由にデータを移動することもできます。ONTAP 9 には、データ管理を簡易化し、重要なデータを高速化、保護し、ハイブリッドクラウドアーキテクチャ全体で将来のニーズに対応できるインフラを実現する、多数の機能が搭載されています。

データ管理を簡易化

データ管理は、アプリケーションやデータセットに適切なリソースを使用できるようにするために、企業の IT 運用にとって非常に重要です。ONTAP には、運用を合理化および簡易化し、総運用コストを削減するための次の機能が含まれています。

- * インラインデータコンパクションと重複排除の強化。 * データコンパクションはストレージブロック内の無駄なスペースを削減し、重複排除は実効容量を大幅に増やします。
- * 最小、最大、アダプティブの Quality of Service （ QoS ；サービス品質）。 * きめ細かい QoS 管理機能により、高度に共有された環境で重要なアプリケーションのパフォーマンスレベルを維持できます。

- * StorageGRID 。 * この機能は、 Amazon Web Services （ AWS ）、 Azure 、 NetApp ONTAP FabricPool オブジェクトベースストレージなどのパブリックおよびプライベートクラウドストレージオプションへのコールドデータの自動階層化を提供します。

データの高速化と保護

ONTAP は、卓越したパフォーマンスとデータ保護を実現し、以下の機能を通じてこれらの機能を拡張します。

- * ハイパフォーマンスと低レイテンシ。 * ONTAP は、可能な限り低いレイテンシで最高のスループットを提供します。
- * NetApp ONTAP FlexGroup テクノロジ。 * FlexGroup ボリュームは、最大 20PB と 4 、 000 億ファイルまでリニアに拡張可能な高性能データコンテナで、データ管理を簡易化する単一のネームスペースを提供します。
- * データ保護。 * ONTAP は、組み込みのデータ保護機能を提供し、すべてのプラットフォームで共通の管理を実現します。
- * NetApp Volume Encryption* ONTAP は、オンボードと外部の両方のキー管理をサポートし、ボリュームレベルのネイティブ暗号化を実現します。

将来のニーズにも対応できるインフラ

ONTAP 9 は、要件が厳しく、絶えず変化するビジネスニーズに対応します。

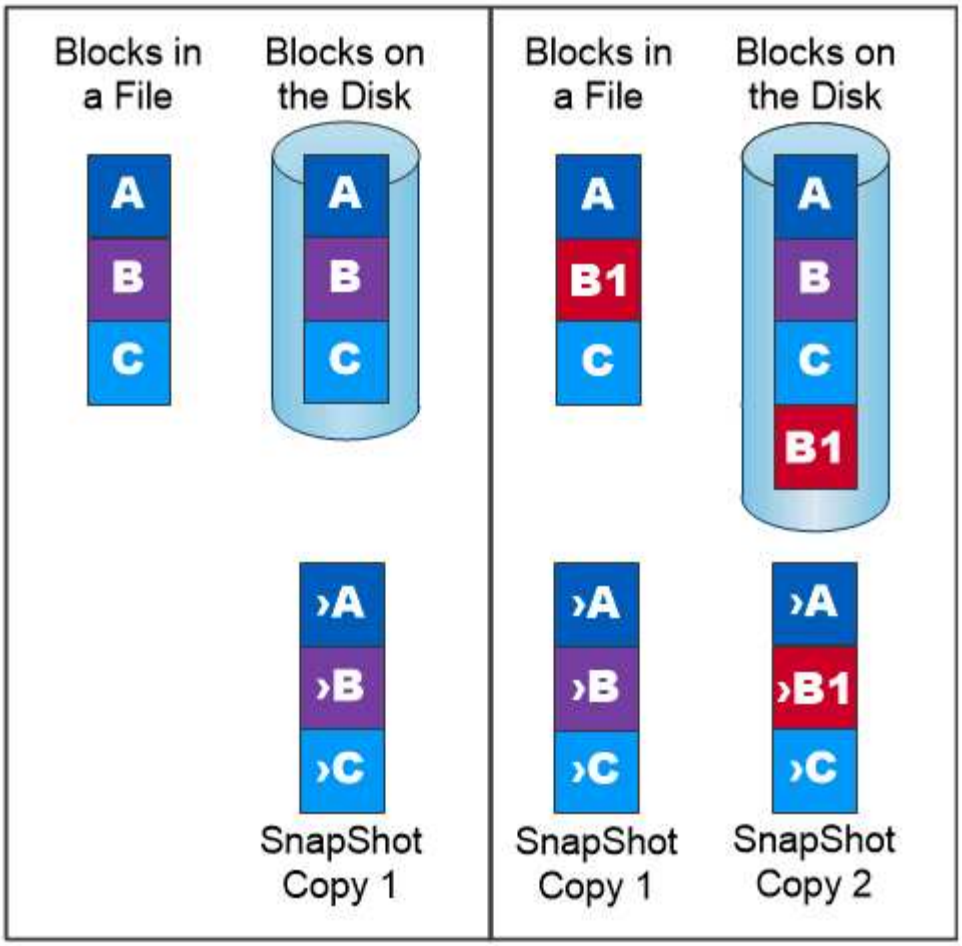
- * シームレスな拡張とノンストップオペレーション。 * ONTAP は、既存のコントローラとスケールアウトクラスタに無停止で容量を追加できます。NVMe や 32Gb FC などの最新テクノロジへのアップグレードも、コストのかかるデータ移行やシステム停止を行わずに実行できます。
- * クラウドへの接続。 * ONTAP は、すべてのパブリッククラウドで Software-Defined Storage （ ONTAP Select ）とクラウドネイティブインスタンス（ NetApp Cloud Volumes Service ）を選択できる、最もクラウドに接続されたストレージ管理ソフトウェアの 1 つです。
- * 新しいアプリケーションとの統合。 * 既存のエンタープライズアプリケーションをサポートする同じインフラを使用して、 ONTAP は、 OpenStack 、 Hadoop 、 MongoDB などの次世代プラットフォームやアプリケーションにエンタープライズクラスのデータサービスを提供します。

NetApp Snapshot コピー

NetApp Snapshot コピーは、ボリュームの読み取り専用のポイントインタイムイメージです。次の図に示すように、イメージには Snapshot コピーが最後に作成されたあとに作成されたファイルへの変更だけが記録されるため、ストレージスペースは最小限しか消費せず、パフォーマンスのオーバーヘッドもわずかです。

Snapshot コピーの効率性は、 ONTAP の中核的なストレージ仮想化テクノロジである Write Anywhere File Layout （ WAFL ）によって実現します。WAFL は、データベースと同様に、メタデータを使用してディスク上の実際のデータブロックを参照します。ただし、データベースとは異なり、 WAFL は既存のブロックを上書きしません。更新されたデータは新しいブロックに書き込まれ、メタデータが変更されます。ONTAP では、 Snapshot コピーの作成時にデータブロックをコピーするのではなくメタデータを参照するため、非常に効率的です。他のシステムと違ってコピーするブロックを探すシーク時間もなければ、コピー自体を作成するコストもかかりません。

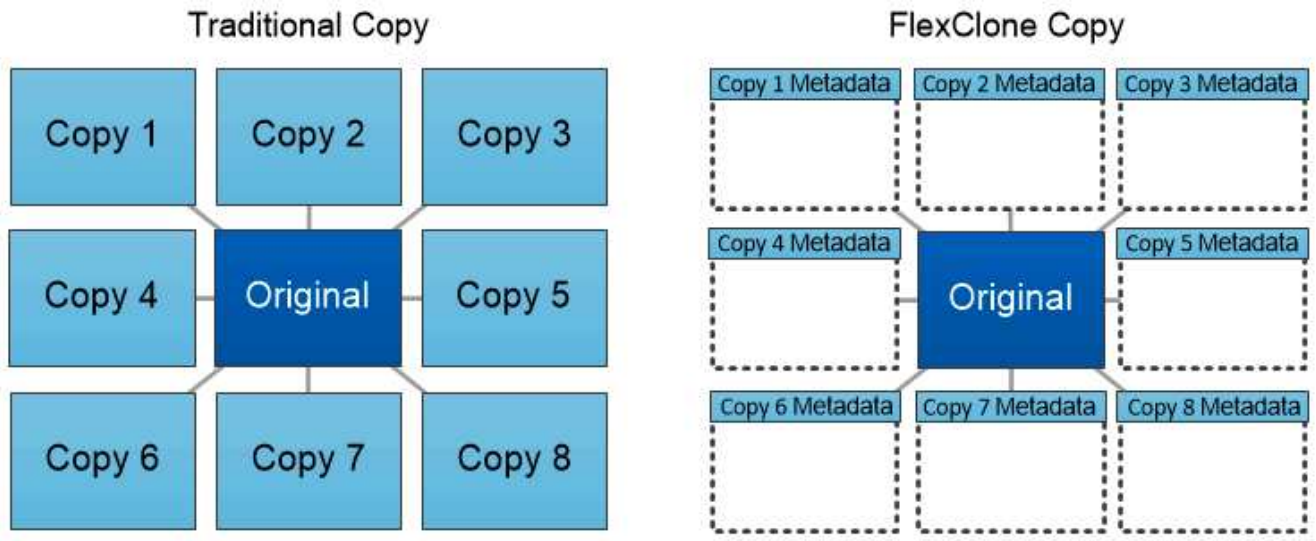
Snapshot コピーを使用して、個々のファイルまたは LUN をリカバリしたり、ボリュームの内容全体をリストアしたりできます。ONTAP は、 Snapshot コピーのポインタ情報をディスク上のデータと比較することで、ダウンタイムや多大なパフォーマンスコストなしで損失オブジェクトや破損オブジェクトを再構築します。



A Snapshot copy records only changes to the active file system since the last Snapshot copy.

NetApp FlexClone テクノロジ

NetApp FlexClone テクノロジは、 Snapshot メタデータを参照してボリュームの書き込み可能なポイントインタイムコピーを作成します。コピーと親でデータブロックが共有されるため、次の図に示すように、コピーに変更が書き込まれるまではメタデータに必要な分しかストレージは消費されません。従来の手法でコピーを作成すると数分から数時間かかりますが、 FlexClone ソフトウェアを使用すれば大規模なデータセットのコピーもほぼ瞬時に作成できます。そのため、同じデータセットのコピーが複数必要な状況（開発用ワークスペースなど）や一時的にデータセットのコピーが必要な状況（本番環境のデータセットでアプリケーションをテストする場合など）に適しています。

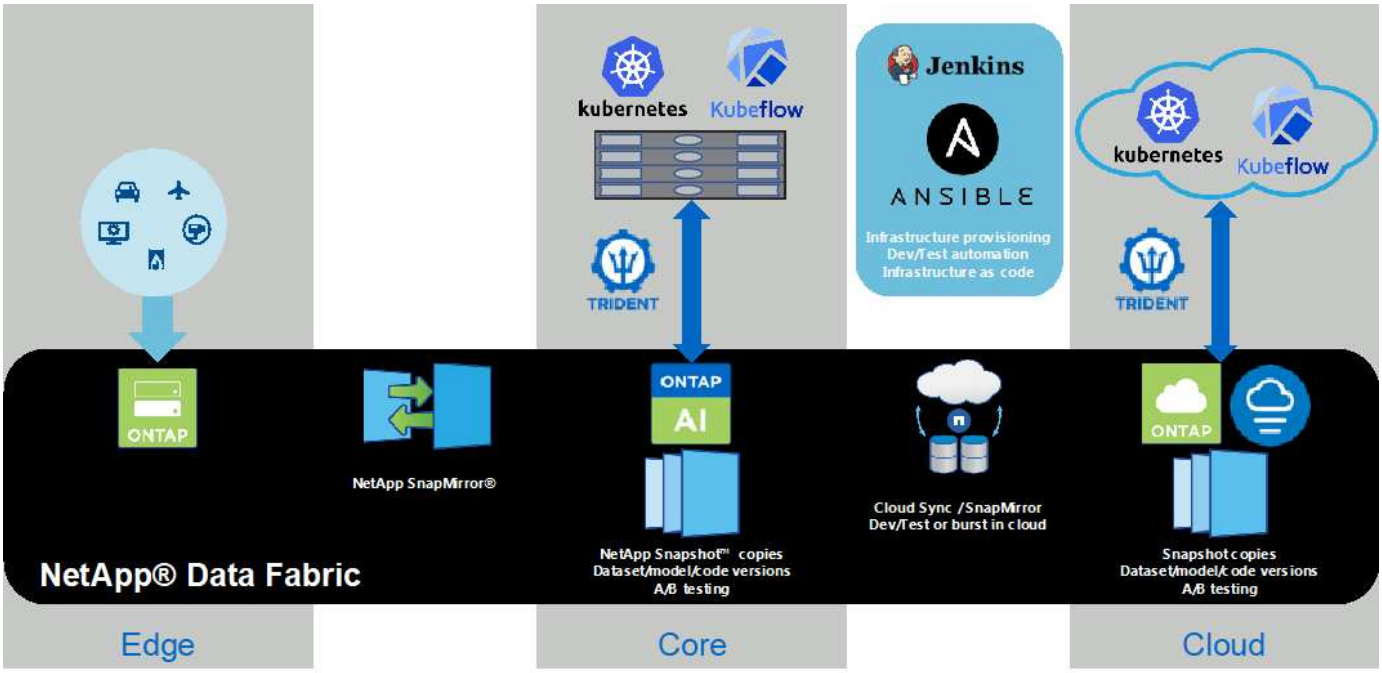


FlexClone copies share data blocks with their parents, consuming no storage except what is required for metadata.

NetApp SnapMirror データレプリケーションテクノロジ

NetApp SnapMirror ソフトウェアは、データファブリック全体にわたる、コスト効率に優れた使いやすいユニファイドレプリケーション解決策です。LAN または WAN 経由でデータを高速で複製します。仮想環境と従来の環境の両方でビジネスクリティカルなアプリケーションを含む、あらゆるタイプのアプリケーションに対し、高いデータ可用性と高速なデータレプリケーションを提供します。1 つ以上のネットアップストレージシステムにデータをレプリケートし、セカンダリデータを継続的に更新すると、データが最新の状態に保たれ、必要なときにいつでも使用できます。外部レプリケーションサーバは必要ありません。SnapMirror テクノロジを利用したアーキテクチャの例については、次の図を参照してください。

SnapMirror ソフトウェアは、変更されたブロックのみをネットワーク経由で送信することで、 NetApp ONTAP の Storage Efficiency 機能を活用します。SnapMirror ソフトウェアには、組み込みのネットワーク圧縮機能も使用して、データ転送を高速化し、ネットワーク帯域幅の使用量を最大 70% 削減します。SnapMirror テクノロジを使用すると、 1 つのシンレプリケーションデータストリームを利用して単一のリポジトリを作成し、アクティブなミラーと以前のポイントインタイムコピーの両方を保持できるため、ネットワークトラフィックを最大 50% 削減できます。



NetApp BlueXPのコピーと同期

BlueXPのコピーと同期は、迅速かつセキュアなデータ同期を実現するNetAppサービスです。オンプレミスのNFSまたはSMBファイル共有（NetApp StorageGRID、NetApp ONTAP S3、NetApp Cloud Volumes Service、Azure NetApp Files、AWS S3、AWS EFS、Azure Blob）間でファイルを転送する必要があるかどうか Google Cloud Storage（IBM Cloud Object Storage）のBlueXP Copy and Syncは、必要な場所に迅速かつ安全にファイルを移動します。

転送されたデータは、ソースとターゲットの両方で完全に使用できます。BlueXPのCopy and Syncは、更新がトリガーされたときにオンデマンドでデータを同期したり、事前定義されたスケジュールに基づいてデータを継続的に同期したりできます。いずれにせよ、BlueXPのCopy and Syncは差分のみを移動するため、データレプリケーションにかかる時間とコストを最小限に抑えることができます。

BlueXPのCopy and Syncは、セットアップと使用が非常に簡単なソフトウェアサービス（SaaS）ツールです。BlueXPのCopyとSyncによってトリガーされるデータ転送は、データブローカーによって実行されます。BlueXPのCopy and Syncデータブローカーは、AWS、Azure、Google Cloud Platform、オンプレミスに導入できます。

NetApp XCP

NetApp XCP は、ネットアップとネットアップ間のデータ移行およびファイルシステムに関する分析情報を提供するクライアントベースのソフトウェアです。XCP は、大量のデータセットとハイパフォーマンスな移行を処理するために、利用可能なすべてのシステムリソースを活用することで、最大限のパフォーマンスを実現するように設計されています。ファイルシステムを完全に可視化するために XCP を使用すると、レポート生成オプションが利用できます。

NetApp XCP は、 NFS プロトコルと SMB プロトコルをサポートする単一パッケージで提供されます。NFS データセット用の Linux バイナリと SMB データセット用の Windows 実行可能ファイルが XCP に含まれています。

NetApp XCP File Analytics は、ファイル共有を検出し、ファイルシステム上でスキャンを実行し、ファイル分析用のダッシュボードを提供するホストベースのソフトウェアです。XCP File Analytics は、ネットアップシステムと他社システムの両方に対応し、 Linux ホストまたは Windows ホストで動作して、 NFS および

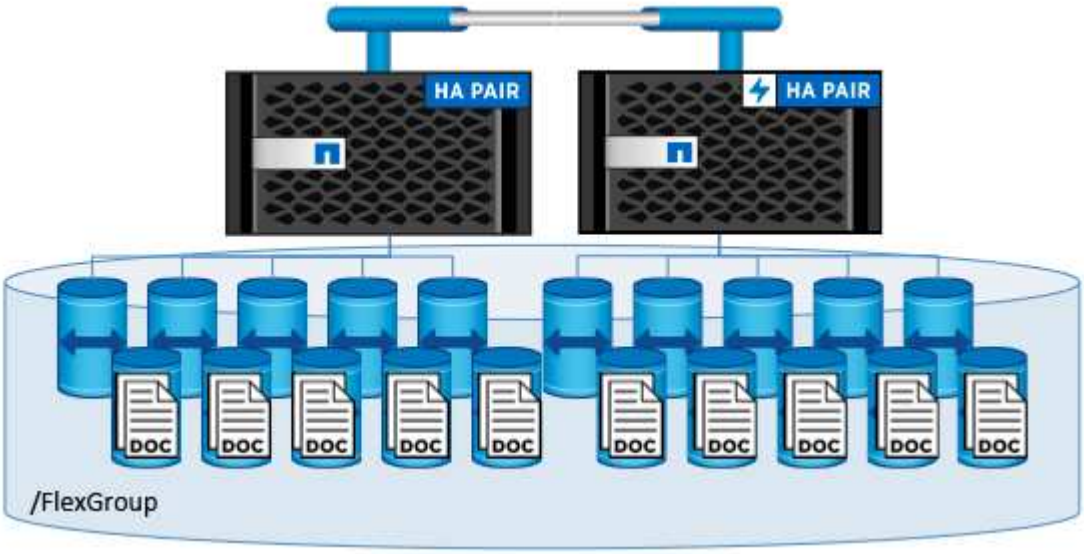
SMB エクスポートファイルシステムの分析を提供します。

NetApp ONTAP FlexGroup Volume の略

トレーニングデータセットは、数十億に及ぶ可能性のあるファイルの集まりです。ファイルには、テキスト、オーディオ、ビデオなどの形式の非構造化データを含めることができます。これらのデータは、並行して読み込まれるように保存して処理する必要があります。ストレージシステムは、多数の小さなファイルを格納し、シーケンシャル I/O とランダム I/O でそれらのファイルを並行して読み取る必要があります

FlexGroup ボリュームは、次の図に示すように、複数のコンスティチュエントメンバーボリュームで構成される単一のネームスペースです。ストレージ管理者の視点で見ると、 FlexGroup ボリュームは管理され、NetApp FlexVol ボリュームのように機能します。FlexGroup ボリューム内のファイルは、個々のメンバーボリュームに割り当てられ、複数のボリュームやノードにまたがってストライプされることはありません。次の機能が有効になります。

- FlexGroup ボリュームは、数ペタバイトの容量と、メタデータ比率の高いワークロード向けの予測可能な低レイテンシを提供します。
- 同じネームスペースで最大 4 、 000 億個のファイルをサポートします。
- CPU 、ノード、アグリゲート、コンスティチュエント FlexVol ボリューム全体で NAS ワークロードの並列処理をサポートします。



ハードウェアとソフトウェアの要件

NetApp AI コントロールプレーン解決策は、このような特定のハードウェアには依存しません。解決策は、 Trident でサポートされている、任意のネットアップ物理ストレージアプライアンス、ソフトウェア定義インスタンス、クラウドサービスと互換性があります。例としては、ネットアップ AFF ストレージシステム、 Azure NetApp Files 、ネットアップ Cloud Volumes Service 、ネットアップ ONTAP Select ソフトウェアで定義されるストレージインスタンス、ネットアップ Cloud Volumes ONTAP インスタンスなどがあります。また、使用する Kubernetes のバージョンが Kubeflow および NetApp Trident でサポートされていれば、どの Kubernetes クラスタにも解決策を実装できます。Kubeflow でサポートされる Kubernetes バージョンの一覧については、を参照して

ください "Kubeflow の公式ドキュメント"。Trident でサポートされている Kubernetes のバージョンのリストについては、を参照してください "Trident のドキュメント"。解決策の検証に使用した環境の詳細については、次の表を参照してください。

インフラストラクチャコンポーネント	数量	詳細	オペレーティングシステム
展開ジャンプホスト	1.	VM	Ubuntu 20.04.2 LTS の場合は
Kubernetes マスターノード	1.	VM	Ubuntu 20.04.2 LTS の場合は
Kubernetes ワーカーノード	2.	VM	Ubuntu 20.04.2 LTS の場合は
Kubernetes GPU ワーカーノード	2.	NVIDIA DGX-1 （ベアメタル）	NVIDIA DGX OS 4.0.5 （Ubuntu 18.04.2 LTS に基づく）
ストレージ	1 つの HA ペア	NetApp AFF A220	NetApp ONTAP 9.7 P6

ソフトウェアコンポーネント	バージョン
Apache の通気	2.0.1
Apache Airflow Helm チャート	8.0.8
Docker です	19.03.12
クビフロー	1/2
Kubernetes	1.18.9
NetApp Trident	21.01.2.
NVIDIA DeepOps のことです	コミット時点でマスターブランチから Trident 導入機能を利用できるようになりました "61898cdfda"; バージョン 21.03 の他のすべての機能

サポート

ネットアップは、 Apache Airflow 、 Docker 、 Kubeflow 、 Kubernetes 、 NVIDIA DeepOps のエンタープライズサポートを提供していません。ネットアップの AI コントロールプレーン解決策と同様の機能を備えた、完全にサポートされている解決策に関心がある場合： "ネットアップにお問い合わせください" ネットアップがパートナー様と共同で提供する、完全にサポートされている AI / ML ソリューションについて説明します。

Kubernetes の導入

このセクションでは、 NetApp AI コントロールプレーン解決策を実装する Kubernetes クラスタを導入するために完了しておく必要があるタスクについて説明します。Kubernetes クラスタをすでにお持ちの場合は、 Kubeflow と NetApp Trident でサポートされるバージョンの Kubernetes を実行していれば、このセクションをスキップできます。Kubeflow でサポートされる Kubernetes バージョンの一覧については、を参照してください "Kubeflow の公式ドキュメント"。Trident でサポートされている Kubernetes のバージョンのリストについては、を参照してください "Trident のドキュメ

ント"。

NVIDIA GPU を搭載したベアメタルノードを統合したオンプレミスの Kubernetes 環境では、 NVIDIA の DeepOps Kubernetes 導入ツールの使用を推奨します。このセクションでは、 DeepOps を使用した Kubernetes クラスタの導入について説明します。

前提条件

このセクションで説明する導入の演習を行う前に、次の作業をすでに実行していることを前提としています。

1. 標準的な構成手順に従って、ベアメタルの Kubernetes ノード（ ONTAP AI ポッドに含まれる NVIDIA DGX システムなど）を設定済みであること。
2. すべての Kubernetes マスターノードとワーカーノード、および導入ジャンプホストに、サポートされているオペレーティングシステムをインストールしておきます。DeepOps でサポートされているオペレーティングシステムのリストについては、を参照してください "DeepOps GitHub サイト"。

NVIDIA DeepOps を使用して Kubernetes をインストールおよび設定します

NVIDIA DeepOps で Kubernetes クラスタを導入および設定するには、導入ジャンプホストから次のタスクを実行します。

1. の手順に従って、 NVIDIA DeepOps をダウンロードします "「はじめに」ページ" NVIDIA DeepOps GitHub サイトで入手できます。
2. の手順に従って、クラスタに Kubernetes を導入します 。 "Kubernetes 導入ガイドのページ" NVIDIA DeepOps GitHub サイトで入手できます。

NetApp Trident の導入と設定

NetApp Trident の導入と設定

このセクションでは、 Kubernetes クラスタに NetApp Trident をインストールして設定するために必要な作業について説明します。

前提条件

このセクションで説明する導入の演習を行う前に、次の作業をすでに実行していることを前提としています。

1. Kubernetes クラスタはすでに稼働しており、 Trident でサポートされるバージョンの Kubernetes を実行している。サポートされているバージョンの一覧については、を参照してください "Trident のドキュメント"。
2. すでに稼働しているネットアップストレージアプライアンス、ソフトウェア定義インスタンス、クラウドストレージサービスで、 Trident でサポートされているものを使用している。

Trident をインストール

Kubernetes クラスタに NetApp Trident をインストールして設定するには、導入ジャンプホストから次のタスクを実行します。

1. Trident は、次のいずれかの方法で導入できます。
    - NVIDIA DeepOps を使用して Kubernetes クラスタを導入した場合は、 NVIDIA DeepOps を使用して

Kubernetes クラスタに Trident を導入することもできます。Trident で DeepOps を導入する方法については、を参照してください "Trident の導入手順" NVIDIA DeepOps GitHub サイトで入手できます。

- NVIDIA DeepOps を使用して Kubernetes クラスタを導入していない場合や Trident を手動で導入する場合は、次の手順で Trident を導入できます "導入手順" Trident のドキュメントの設定方法の詳細については、Tridentバックエンドと少なくとも1つのKubernetes StorageClassを作成してください "バックエンド" および "ストレージクラス" ネットアップドキュメントのリンクされたサブセクションを参照してください。

> 解決策 AI ポッドにネットアップ AI コントロールプレーン ONTAP を導入する場合は、を参照してください "ONTAP AI 導入向け Trident バックエンドの例" さまざまな Trident バックエンドの例を紹介します とを作成します "ONTAP AI 導入向けの Kubernetes StorageClasses の例" を参照してください。

ONTAP AI 導入向け **Trident** バックエンドの例

Trident を使用して Kubernetes クラスタ内のストレージリソースを動的にプロビジョニングするには、 Trident バックエンドを 1 つ以上作成する必要があります。以下に示す例は、 ONTAP AI ポッドにネットアップ AI コントロールプレーン解決策を導入する場合に作成するバックエンドのタイプを表しています。バックエンドの詳細については、を参照してください "Trident のドキュメント"。

1. ネットアップでは、 NetApp AFF システムで使用する各データ LIF （データアクセスを提供する論理ネットワークインターフェイス）に対して、 FlexGroup 対応の Trident バックエンドを作成することを推奨します。これにより、 LIF 間でボリュームマウントを分散させることができます

   以下のコマンド例は、同じ ONTAP Storage Virtual Machine （ SVM ）に関連付けられている 2 つの異なるデータ LIF を対象に、 FlexGroup 対応の Trident バックエンドを 2 つ作成する方法を示しています。これらのバックエンドは 'ONTAP-NAS-flexgroup ストレージ・ドライバを使用しますONTAP では、 FlexVol と FlexGroup の 2 つの主要なデータボリュームタイプがサポートされます。FlexVol ボリュームのサイズは限られています（現時点では、最大サイズは環境によって異なります）。一方、 FlexGroup ボリュームは最大 20PB 、 4 、 000 億ファイルまでリニアに拡張でき、データ管理を大幅に簡易化する単一のネームスペースを提供します。そのため、 FlexGroup ボリュームは、大量のデータに依存する AI や ML のワークロードに最適です。

   少量のデータを処理していて、 FlexGroup ボリュームではなく FlexVol ボリュームを使用する場合は、「 ONTAP-NAS-flexgroup 」ストレージドライバの代わりに「 ONTAP-NAS' ストレージドライバ」を使用する Trident バックエンドを作成できます。

```
$ cat << EOF > ./trident-backend-ontap-ai-flexgroups-iface1.json
{
    "version": 1,
    "storageDriverName": "ontap-nas-flexgroup",
    "backendName": "ontap-ai-flexgroups-iface1",
    "managementLIF": "10.61.218.100",
    "dataLIF": "192.168.11.11",
    "svm": "ontapai_nfs",
    "username": "admin",
    "password": "ontapai"
```

```
}
EOF
$ tridentctl create backend -f ./trident-backend-ontap-ai-flexgroups-
iface1.json -n trident
+----------------------------+---------------------
+--------------------------------------+--------+---------+
|            NAME            |   STORAGE DRIVER    |
UUID                 | STATE  | VOLUMES |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
| ontap-ai-flexgroups-iface1 | ontap-nas-flexgroup | b74cbddb-e0b8-40b7-
b263-b6da6dec0bdd | online |       0 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
$ cat << EOF > ./trident-backend-ontap-ai-flexgroups-iface2.json
{
    "version": 1,
    "storageDriverName": "ontap-nas-flexgroup",
    "backendName": "ontap-ai-flexgroups-iface2",
    "managementLIF": "10.61.218.100",
    "dataLIF": "192.168.12.12",
    "svm": "ontapai_nfs",
    "username": "admin",
    "password": "ontapai"
}
EOF
$ tridentctl create backend -f ./trident-backend-ontap-ai-flexgroups-
iface2.json -n trident
+----------------------------+---------------------
+--------------------------------------+--------+---------+
|            NAME            |   STORAGE DRIVER    |
UUID                 | STATE  | VOLUMES |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
| ontap-ai-flexgroups-iface2 | ontap-nas-flexgroup | 61814d48-c770-436b-
9cb4-cf7ee661274d | online |       0 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
$ tridentctl get backend -n trident
+----------------------------+---------------------
+--------------------------------------+--------+---------+
|            NAME            |   STORAGE DRIVER    |
UUID                 | STATE  | VOLUMES |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
| ontap-ai-flexgroups-iface1 | ontap-nas-flexgroup | b74cbddb-e0b8-40b7-
```

```
b263-b6da6dec0bdd | online |       0 |
| ontap-ai-flexgroups-iface2 | ontap-nas-flexgroup | 61814d48-c770-436b-
9cb4-cf7ee661274d | online |       0 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
```

2. また、 FlexVol 対応の Trident バックエンドも 1 つ以上作成することを推奨します。FlexGroup ボリュームをデータセットストレージのトレーニングに使用する場合は、結果、出力、デバッグ情報などの格納に FlexVol ボリュームを使用できます。FlexVol ボリュームを使用する場合は、 FlexVol 対応の Trident バックエンドを 1 つ以上作成する必要があります。次に示すコマンド例は、単一のデータ LIF を使用する、FlexVol 対応の Trident バックエンドの作成例を示しています。

```
$ cat << EOF > ./trident-backend-ontap-ai-flexvols.json
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "ontap-ai-flexvols",
    "managementLIF": "10.61.218.100",
    "dataLIF": "192.168.11.11",
    "svm": "ontapai_nfs",
    "username": "admin",
    "password": "ontapai"
}
EOF
$ tridentctl create backend -f ./trident-backend-ontap-ai-flexvols.json -n
trident
+----------------------------+---------------------
+--------------------------------------+--------+---------+
|            NAME            |   STORAGE DRIVER    |                 UUID
| STATE  | VOLUMES |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
| ontap-ai-flexvols          | ontap-nas           | 52bdb3b1-13a5-4513-
a9c1-52a69657fabe | online |       0 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
$ tridentctl get backend -n trident
+----------------------------+---------------------
+--------------------------------------+--------+---------+
|            NAME            |   STORAGE DRIVER    |                 UUID
| STATE  | VOLUMES |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
| ontap-ai-flexvols          | ontap-nas           | 52bdb3b1-13a5-4513-
a9c1-52a69657fabe | online |       0 |
| ontap-ai-flexgroups-iface1 | ontap-nas-flexgroup | b74cbddb-e0b8-40b7-
b263-b6da6dec0bdd | online |       0 |
| ontap-ai-flexgroups-iface2 | ontap-nas-flexgroup | 61814d48-c770-436b-
9cb4-cf7ee661274d | online |       0 |
+----------------------------+---------------------
+--------------------------------------+--------+---------+
```

ONTAP AI 導入向けの Kubernetes StorageClasses の例

Trident を使用して Kubernetes クラスタ内のストレージリソースを動的にプロビジョニングするには、 Kubernetes StorageClasses を 1 つ以上作成する必要があります。以下に示す例は、 ONTAP AI ポッドにネットアップ AI コントロールプレーン解決策を導入す

る場合に作成する、さまざまなタイプのストレージクラスを表しています。StorageClasses の詳細については、を参照してください "Trident のドキュメント"。

1. FlexGroup が有効な Trident ごとに別々のストレージクラスを作成することを推奨します セクションで作成したバックエンド "ONTAP AI 導入向け Trident バックエンドの例"、手順 1.これらの Granular StorageClasses を使用すると、 StorageClass 仕様ファイルで指定されている特定のバックエンドとして、特定の LIF （ Trident バックエンドの作成時に指定した LIF ）に対応する NFS マウントを追加できます。以降のコマンド例では、 2 つのを作成しています StorageClasses を使用して、バックエンドの 2 つの例に対応しています セクションで作成されます "ONTAP AI 導入向け Trident バックエンドの例"、手順 1.StorageClasses の詳細については、を参照してください "Trident のドキュメント"。

   対応する PersistentVolumeClaim （ PVC ）が削除されたときに永続ボリュームが削除されないようにするため、次の例では「 Retain 」の「 ReclaimPolicy 」の値を使用しています。「 ReclaimPolicy 」フィールドの詳細については、公式を参照してください "Kubernetes のドキュメント"。

```
$ cat << EOF > ./storage-class-ontap-ai-flexgroups-retain-iface1.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-ai-flexgroups-retain-iface1
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas-flexgroup"
  storagePools: "ontap-ai-flexgroups-iface1:.*"
reclaimPolicy: Retain
EOF
$ kubectl create -f ./storage-class-ontap-ai-flexgroups-retain-
iface1.yaml
storageclass.storage.k8s.io/ontap-ai-flexgroups-retain-iface1 created
$ cat << EOF > ./storage-class-ontap-ai-flexgroups-retain-iface2.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-ai-flexgroups-retain-iface2
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas-flexgroup"
  storagePools: "ontap-ai-flexgroups-iface2:.*"
reclaimPolicy: Retain
EOF
$ kubectl create -f ./storage-class-ontap-ai-flexgroups-retain-
iface2.yaml
storageclass.storage.k8s.io/ontap-ai-flexgroups-retain-iface2 created
$ kubectl get storageclass
NAME                                  PROVISIONER         AGE
ontap-ai-flexgroups-retain-iface1   netapp.io/trident   0m
ontap-ai-flexgroups-retain-iface2   netapp.io/trident   0m
```

2. に対応するストレージクラスを作成することも推奨します セクションで作成した FlexVol 対応の Trident バックエンド "ONTAP AI 導入向け Trident バックエンドの例"、ステップ 2 。以下のコマンド例は、FlexVol ボリューム用の単一のストレージクラスの作成を示しています。

   次の例では、 FlexVol 対応の Trident バックエンドが 1 つしか作成されていないため、 StorageClass 定義ファイルで特定のバックエンドが指定されていません。Kubernetes を使用してこの StorageClass を使用するボリュームを管理すると、 Trident は「 ONTAP-NAS' 」ドライバを使用するバックエンドで利用可能なものを使用しようとします。

```
$ cat << EOF > ./storage-class-ontap-ai-flexvols-retain.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-ai-flexvols-retain
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas"
reclaimPolicy: Retain
EOF
$ kubectl create -f ./storage-class-ontap-ai-flexvols-retain.yaml
storageclass.storage.k8s.io/ontap-ai-flexvols-retain created
$ kubectl get storageclass
NAME                                  PROVISIONER         AGE
ontap-ai-flexgroups-retain-iface1   netapp.io/trident   1m
ontap-ai-flexgroups-retain-iface2   netapp.io/trident   1m
ontap-ai-flexvols-retain            netapp.io/trident   0m
```

3. また、 FlexGroup ボリューム用の汎用の StorageClass を作成することを推奨します。次のコマンド例は、 FlexGroup ボリューム用の汎用の StorageClass を 1 つ作成する方法を示しています。

   特定のバックエンドが StorageClass 定義ファイルで指定されていないことに注意してください。したがって、 Kubernetes を使用してこのストレージクラスを使用するボリュームを管理する場合、 Trident は「 ONTAP-NAS-flexgroup 」ドライバを使用する利用可能なバックエンドを使用しようとします。

```
$ cat << EOF > ./storage-class-ontap-ai-flexgroups-retain.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-ai-flexgroups-retain
provisioner: netapp.io/trident
parameters:
  backendType: "ontap-nas-flexgroup"
reclaimPolicy: Retain
EOF
$ kubectl create -f ./storage-class-ontap-ai-flexgroups-retain.yaml
storageclass.storage.k8s.io/ontap-ai-flexgroups-retain created
$ kubectl get storageclass
NAME                                  PROVISIONER         AGE
ontap-ai-flexgroups-retain          netapp.io/trident   0m
ontap-ai-flexgroups-retain-iface1   netapp.io/trident   2m
ontap-ai-flexgroups-retain-iface2   netapp.io/trident   2m
ontap-ai-flexvols-retain            netapp.io/trident   1m
```

Kubeflow の導入

このセクションでは、 Kubernetes クラスタに Kubeflow を導入するために実行する必要のあるタスクについて説明します。

前提条件

このセクションで説明する導入の演習を行う前に、次の作業をすでに実行していることを前提としています。

1. Kubernetes クラスタはすでに稼働しており、 Kubeflow でサポートされるバージョンの Kubernetes を実行しているとします。サポートされているバージョンの一覧については、を参照してください "Kubeflow の公式ドキュメント"。
2. 内に NetApp Trident のインストールと設定が完了している必要があります Kubernetes クラスタ（を参照） "Trident の導入と設定"。

デフォルトの **Kubernetes StorageClass** を設定します

Kubeflow を導入する前に、 Kubernetes クラスタ内でデフォルトの StorageClass を指定する必要があります。Kubeflow の導入プロセスでは、デフォルトの StorageClass を使用して新しい永続ボリュームのプロビジョニングが試行されます。StorageClass がデフォルトの StorageClass として指定されていない場合、導入は失敗します。クラスタ内のデフォルトの StorageClass を指定するには、導入ジャンプホストから次のタスクを実行します。クラスタ内ですでにデフォルトの StorageClass を指定している場合は、この手順を省略できます。

1. 既存のストレージクラスの 1 つをデフォルトのストレージクラスとして指定します。以降のコマンド例では、「 ontap/ai-sFLEXs-retain 」という名前の StorageClass がデフォルトの StorageClass として指定されています。

> 「 ONTAP-NAS-flexgroup 」の Trident バックエンドタイプには、かなり大きな最小 PVC サイズがあります。デフォルトでは、 Kubeflow はサイズが数 GB しかない PVC のプロビジョニングを試みます。したがって、 Kubeflow の導入の目的で、「 ONTAP-NAS-flexgroup 」バックエンドタイプをデフォルトの StorageClass として使用する StorageClass を指定しないでください。

```
$ kubectl get sc
NAME                                  PROVISIONER             AGE
ontap-ai-flexgroups-retain            csi.trident.netapp.io   25h
ontap-ai-flexgroups-retain-iface1     csi.trident.netapp.io   25h
ontap-ai-flexgroups-retain-iface2     csi.trident.netapp.io   25h
ontap-ai-flexvols-retain              csi.trident.netapp.io   3s
$ kubectl patch storageclass ontap-ai-flexvols-retain -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
storageclass.storage.k8s.io/ontap-ai-flexvols-retain patched
$ kubectl get sc
NAME                                   PROVISIONER             AGE
ontap-ai-flexgroups-retain             csi.trident.netapp.io   25h
ontap-ai-flexgroups-retain-iface1      csi.trident.netapp.io   25h
ontap-ai-flexgroups-retain-iface2      csi.trident.netapp.io   25h
ontap-ai-flexvols-retain (default)     csi.trident.netapp.io   54s
```

NVIDIA DeepOps を使用して Kubeflow を導入します

NVIDIA DeepOps が提供する Kubeflow 導入ツールを使用することを推奨します。DeepOps 導入ツールを使用して Kubernetes クラスタに Kubeflow を導入するには、導入ジャンプホストから次のタスクを実行します。

> または、を使用して手動で Kubeflow を導入することもできます "インストール手順" Kubeflow の公式ドキュメントにあります

1. クラスタに Kubeflow を導入するには、に従ってください "Kubeflow の導入手順" NVIDIA DeepOps GitHub サイトで入手できます。
2. Kubeflow Dashboard URL をメモしてください。 DeepOps Kubeflow 導入ツールによって出力されます。

```
$ ./scripts/k8s/deploy_kubeflow.sh -x
…
INFO[0007] Applied the configuration Successfully!
filename="cmd/apply.go:72"
Kubeflow app installed to: /home/ai/kubeflow
It may take several minutes for all services to start. Run 'kubectl get
pods -n kubeflow' to verify
To remove (excluding CRDs, istio, auth, and cert-manager), run:
./scripts/k8s_deploy_kubeflow.sh -d
To perform a full uninstall : ./scripts/k8s_deploy_kubeflow.sh -D
Kubeflow Dashboard (HTTP NodePort): http://10.61.188.111:31380
```

3. Kubeflow ネームスペース内に展開されているすべてのポッドに「ステータス」が「実行中」であることを確認し、ネームスペース内に展開されているコンポーネントがエラー状態でないことを確認します。すべてのポッドが起動するまでに数分かかることがあります。

```
$ kubectl get all -n kubeflow
NAME                                                                READY
STATUS    RESTARTS   AGE
pod/admission-webhook-bootstrap-stateful-set-0                      1/1
Running   0          95s
pod/admission-webhook-deployment-6b89c84c98-vrtbh                   1/1
Running   0          91s
pod/application-controller-stateful-set-0                           1/1
Running   0          98s
pod/argo-ui-5dcf5d8b4f-m2wn4                                        1/1
Running   0          97s
pod/centraldashboard-cf4874ddc-7hcr8                                1/1
Running   0          97s
pod/jupyter-web-app-deployment-685b455447-gjhh7                     1/1
Running   0          96s
pod/katib-controller-88c97d85c-kgq66                                1/1
Running   1          95s
pod/katib-db-8598468fd8-5jw2c                                       1/1
Running   0          95s
pod/katib-manager-574c8c67f9-wtrf5                                  1/1
Running   1          95s
pod/katib-manager-rest-778857c989-fjbzn                             1/1
Running   0          95s
pod/katib-suggestion-bayesianoptimization-65df4d7455-qthmw          1/1
Running   0          94s
pod/katib-suggestion-grid-56bf69f597-98vwn                          1/1
Running   0          94s
pod/katib-suggestion-hyperband-7777b76cb9-9v6dq                     1/1
Running   0          93s
pod/katib-suggestion-nasrl-77f6f9458c-2qzxq                         1/1
Running   0          93s
pod/katib-suggestion-random-77b88b5c79-l64j9                        1/1
Running   0          93s
pod/katib-ui-7587c5b967-nd629                                       1/1
Running   0          95s
pod/metacontroller-0                                                1/1
Running   0          96s
pod/metadata-db-5dd459cc-swzkm                                      1/1
Running   0          94s
pod/metadata-deployment-6cf77db994-69fk7                            1/1
Running   3          93s
pod/metadata-deployment-6cf77db994-mpbjt                            1/1
Running   3          93s
pod/metadata-deployment-6cf77db994-xg7tz                            1/1
Running   3          94s
pod/metadata-ui-78f5b59b56-qb6kr                                    1/1
```

```
Running   0          94s
pod/minio-758b769d67-llvdr                                      1/1
Running   0          91s
pod/ml-pipeline-5875b9db95-g8t2k                                1/1
Running   0          91s
pod/ml-pipeline-persistenceagent-9b69ddd46-bt9r9                1/1
Running   0          90s
pod/ml-pipeline-scheduledworkflow-7b8d756c76-7x56s              1/1
Running   0          90s
pod/ml-pipeline-ui-79ffd9c76-fcwpd                              1/1
Running   0          90s
pod/ml-pipeline-viewer-controller-deployment-5fdc87f58-b2t9r   1/1
Running   0          90s
pod/mysql-657f87857d-l5k9z                                      1/1
Running   0          91s
pod/notebook-controller-deployment-56b4f59bbf-8bvnr             1/1
Running   0          92s
pod/profiles-deployment-6bc745947-mrdkh                         2/2
Running   0          90s
pod/pytorch-operator-77c97f4879-hmlrv                           1/1
Running   0          92s
pod/seldon-operator-controller-manager-0                        1/1
Running   1          91s
pod/spartakus-volunteer-5fdfddb779-l7qkm                        1/1
Running   0          92s
pod/tensorboard-6544748d94-nh8b2                                1/1
Running   0          92s
pod/tf-job-dashboard-56f79c59dd-6w59t                           1/1
Running   0          92s
pod/tf-job-operator-79cbfd6dbc-rb58c                            1/1
Running   0          91s
pod/workflow-controller-db644d554-cwrnb                         1/1
Running   0          97s
NAME                                                   TYPE
CLUSTER-IP      EXTERNAL-IP   PORT(S)              AGE
service/admission-webhook-service                      ClusterIP
10.233.51.169   <none>        443/TCP              97s
service/application-controller-service                 ClusterIP
10.233.4.54     <none>        443/TCP              98s
service/argo-ui                                        NodePort
10.233.47.191   <none>        80:31799/TCP         97s
service/centraldashboard                               ClusterIP
10.233.8.36     <none>        80/TCP               97s
service/jupyter-web-app-service                        ClusterIP
10.233.1.42     <none>        80/TCP               97s
service/katib-controller                               ClusterIP
```

```
10.233.25.226   <none>        443/TCP             96s
service/katib-db                                     ClusterIP
10.233.33.151   <none>        3306/TCP            97s
service/katib-manager                                ClusterIP
10.233.46.239   <none>        6789/TCP            96s
service/katib-manager-rest                           ClusterIP
10.233.55.32    <none>        80/TCP              96s
service/katib-suggestion-bayesianoptimization        ClusterIP
10.233.49.191   <none>        6789/TCP            95s
service/katib-suggestion-grid                        ClusterIP
10.233.9.105    <none>        6789/TCP            95s
service/katib-suggestion-hyperband                   ClusterIP
10.233.22.2     <none>        6789/TCP            95s
service/katib-suggestion-nasrl                       ClusterIP
10.233.63.73    <none>        6789/TCP            95s
service/katib-suggestion-random                      ClusterIP
10.233.57.210   <none>        6789/TCP            95s
service/katib-ui                                     ClusterIP
10.233.6.116    <none>        80/TCP              96s
service/metadata-db                                  ClusterIP
10.233.31.2     <none>        3306/TCP            96s
service/metadata-service                             ClusterIP
10.233.27.104   <none>        8080/TCP            96s
service/metadata-ui                                  ClusterIP
10.233.57.177   <none>        80/TCP              96s
service/minio-service                                ClusterIP
10.233.44.90    <none>        9000/TCP            94s
service/ml-pipeline                                  ClusterIP
10.233.41.201   <none>        8888/TCP,8887/TCP   94s
service/ml-pipeline-tensorboard-ui                   ClusterIP
10.233.36.207   <none>        80/TCP              93s
service/ml-pipeline-ui                               ClusterIP
10.233.61.150   <none>        80/TCP              93s
service/mysql                                        ClusterIP
10.233.55.117   <none>        3306/TCP            94s
service/notebook-controller-service                  ClusterIP
10.233.10.166   <none>        443/TCP             95s
service/profiles-kfam                                ClusterIP
10.233.33.79    <none>        8081/TCP            92s
service/pytorch-operator                             ClusterIP
10.233.37.112   <none>        8443/TCP            95s
service/seldon-operator-controller-manager-service   ClusterIP
10.233.30.178   <none>        443/TCP             92s
service/tensorboard                                  ClusterIP
10.233.58.151   <none>        9000/TCP            94s
service/tf-job-dashboard                             ClusterIP
```

```
10.233.4.17     <none>        80/TCP              94s
service/tf-job-operator                              ClusterIP
10.233.60.32    <none>        8443/TCP            94s
service/webhook-server-service                       ClusterIP
10.233.32.167   <none>        443/TCP             87s
NAME                                                          READY   UP-
TO-DATE   AVAILABLE   AGE
deployment.apps/admission-webhook-deployment                  1/1     1
1           97s
deployment.apps/argo-ui                                       1/1     1
1           97s
deployment.apps/centraldashboard                              1/1     1
1           97s
deployment.apps/jupyter-web-app-deployment                    1/1     1
1           97s
deployment.apps/katib-controller                              1/1     1
1           96s
deployment.apps/katib-db                                      1/1     1
1           97s
deployment.apps/katib-manager                                 1/1     1
1           96s
deployment.apps/katib-manager-rest                            1/1     1
1           96s
deployment.apps/katib-suggestion-bayesianoptimization         1/1     1
1           95s
deployment.apps/katib-suggestion-grid                         1/1     1
1           95s
deployment.apps/katib-suggestion-hyperband                    1/1     1
1           95s
deployment.apps/katib-suggestion-nasrl                        1/1     1
1           95s
deployment.apps/katib-suggestion-random                       1/1     1
1           95s
deployment.apps/katib-ui                                      1/1     1
1           96s
deployment.apps/metadata-db                                   1/1     1
1           96s
deployment.apps/metadata-deployment                           3/3     3
3           96s
deployment.apps/metadata-ui                                   1/1     1
1           96s
deployment.apps/minio                                         1/1     1
1           94s
deployment.apps/ml-pipeline                                   1/1     1
1           94s
deployment.apps/ml-pipeline-persistenceagent                  1/1     1
```

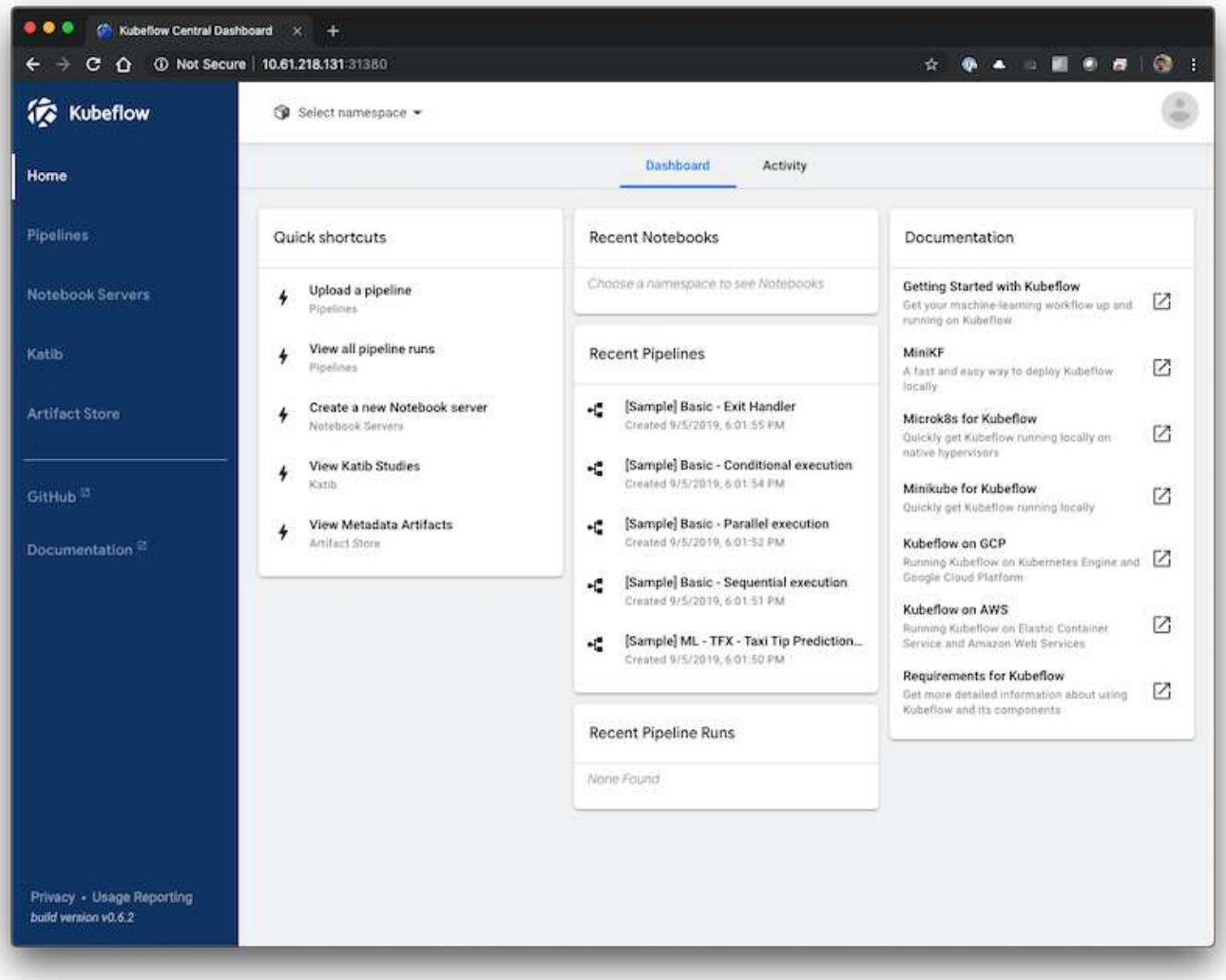
```
1           93s
deployment.apps/ml-pipeline-scheduledworkflow              1/1     1
1           93s
deployment.apps/ml-pipeline-ui                             1/1     1
1           93s
deployment.apps/ml-pipeline-viewer-controller-deployment   1/1     1
1           93s
deployment.apps/mysql                                      1/1     1
1           94s
deployment.apps/notebook-controller-deployment             1/1     1
1           95s
deployment.apps/profiles-deployment                        1/1     1
1           92s
deployment.apps/pytorch-operator                           1/1     1
1           95s
deployment.apps/spartakus-volunteer                        1/1     1
1           94s
deployment.apps/tensorboard                                1/1     1
1           94s
deployment.apps/tf-job-dashboard                           1/1     1
1           94s
deployment.apps/tf-job-operator                            1/1     1
1           94s
deployment.apps/workflow-controller                        1/1     1
1           97s
NAME
DESIRED   CURRENT   READY   AGE
replicaset.apps/admission-webhook-deployment-6b89c84c98              1
1         1       97s
replicaset.apps/argo-ui-5dcf5d8b4f                                   1
1         1       97s
replicaset.apps/centraldashboard-cf4874ddc                           1
1         1       97s
replicaset.apps/jupyter-web-app-deployment-685b455447                1
1         1       97s
replicaset.apps/katib-controller-88c97d85c                           1
1         1       96s
replicaset.apps/katib-db-8598468fd8                                  1
1         1       97s
replicaset.apps/katib-manager-574c8c67f9                             1
1         1       96s
replicaset.apps/katib-manager-rest-778857c989                        1
1         1       96s
replicaset.apps/katib-suggestion-bayesianoptimization-65df4d7455     1
1         1       95s
replicaset.apps/katib-suggestion-grid-56bf69f597                     1
```

```
1         1       95s
replicaset.apps/katib-suggestion-hyperband-7777b76cb9                1
1         1       95s
replicaset.apps/katib-suggestion-nasrl-77f6f9458c                    1
1         1       95s
replicaset.apps/katib-suggestion-random-77b88b5c79                   1
1         1       95s
replicaset.apps/katib-ui-7587c5b967                                  1
1         1       96s
replicaset.apps/metadata-db-5dd459cc                                 1
1         1       96s
replicaset.apps/metadata-deployment-6cf77db994                       3
3         3       96s
replicaset.apps/metadata-ui-78f5b59b56                               1
1         1       96s
replicaset.apps/minio-758b769d67                                     1
1         1       93s
replicaset.apps/ml-pipeline-5875b9db95                               1
1         1       93s
replicaset.apps/ml-pipeline-persistenceagent-9b69ddd46               1
1         1       92s
replicaset.apps/ml-pipeline-scheduledworkflow-7b8d756c76             1
1         1       91s
replicaset.apps/ml-pipeline-ui-79ffd9c76                             1
1         1       91s
replicaset.apps/ml-pipeline-viewer-controller-deployment-5fdc87f58   1
1         1       91s
replicaset.apps/mysql-657f87857d                                     1
1         1       92s
replicaset.apps/notebook-controller-deployment-56b4f59bbf            1
1         1       94s
replicaset.apps/profiles-deployment-6bc745947                        1
1         1       91s
replicaset.apps/pytorch-operator-77c97f4879                          1
1         1       94s
replicaset.apps/spartakus-volunteer-5fdfddb779                       1
1         1       94s
replicaset.apps/tensorboard-6544748d94                               1
1         1       93s
replicaset.apps/tf-job-dashboard-56f79c59dd                          1
1         1       93s
replicaset.apps/tf-job-operator-79cbfd6dbc                           1
1         1       93s
replicaset.apps/workflow-controller-db644d554                        1
1         1       97s
NAME                                                          READY   AGE
```

```
statefulset.apps/admission-webhook-bootstrap-stateful-set   1/1     97s
statefulset.apps/application-controller-stateful-set        1/1     98s
statefulset.apps/metacontroller                             1/1     98s
statefulset.apps/seldon-operator-controller-manager         1/1     92s
$ kubectl get pvc -n kubeflow
NAME              STATUS   VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS                AGE
katib-mysql       Bound    pvc-b07f293e-d028-11e9-9b9d-00505681a82d
10Gi       RWO            ontap-ai-flexvols-retain   27m
metadata-mysql    Bound    pvc-b0f3f032-d028-11e9-9b9d-00505681a82d
10Gi       RWO            ontap-ai-flexvols-retain   27m
minio-pv-claim    Bound    pvc-b22727ee-d028-11e9-9b9d-00505681a82d
20Gi       RWO            ontap-ai-flexvols-retain   27m
mysql-pv-claim    Bound    pvc-b2429afd-d028-11e9-9b9d-00505681a82d
20Gi       RWO            ontap-ai-flexvols-retain   27m
```

4. Web ブラウザで、手順 2 でメモした URL に移動して Kubeflow 中央ダッシュボードにアクセスします。

   デフォルトのユーザ名は「 admin@kubeflow.org 」で、デフォルトのパスワードは「 12341234 」です。追加ユーザを作成するには、の手順に従います "Kubeflow の公式ドキュメント"。



例： **Kubeflow** の操作とタスク

このセクションでは、 Kubeflow を使用して実行したいさまざまな操作とタスクの例を示します。

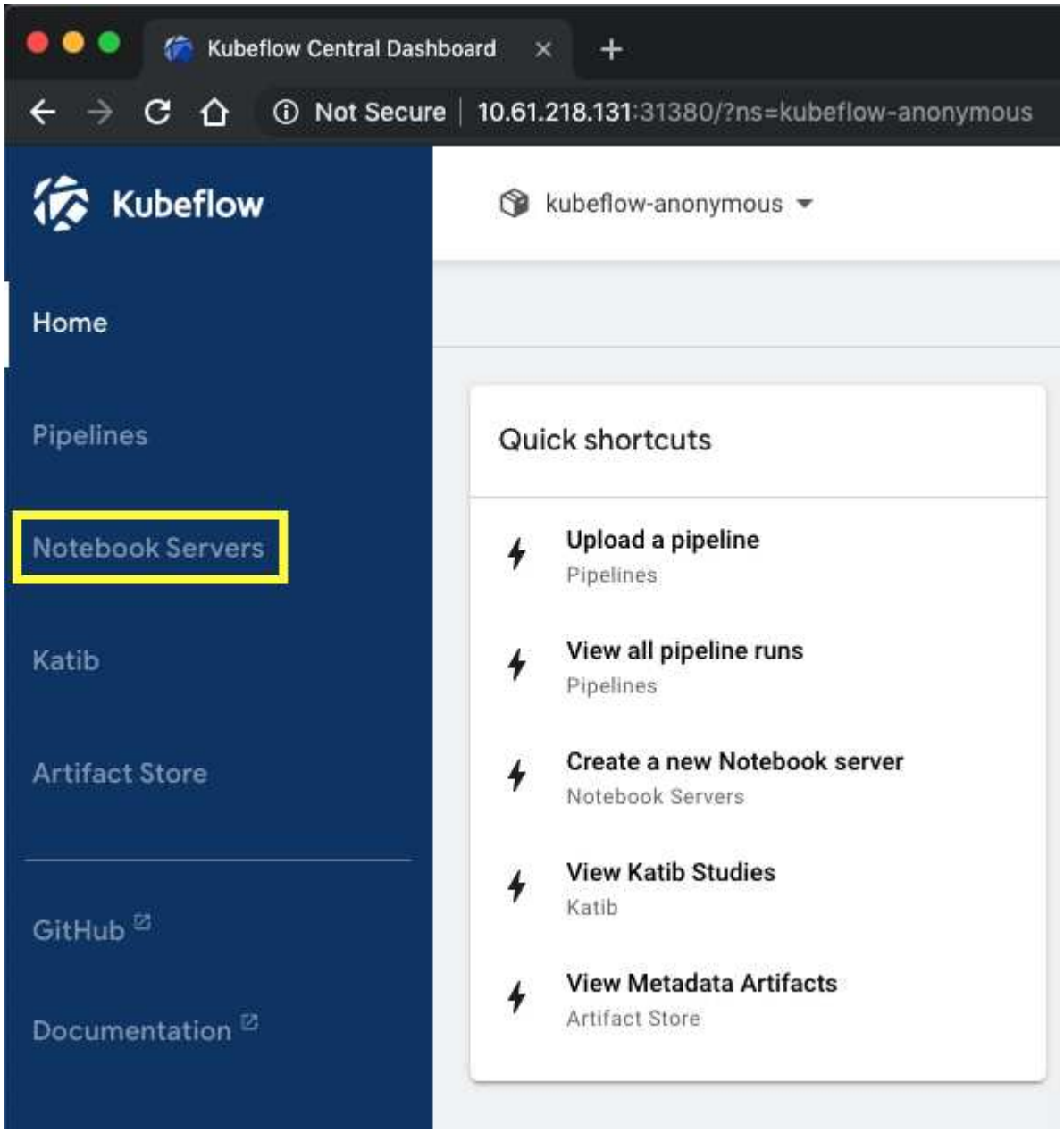
例： **Kubeflow** の操作とタスク

このセクションでは、 Kubeflow を使用して実行したいさまざまな操作とタスクの例を示します。

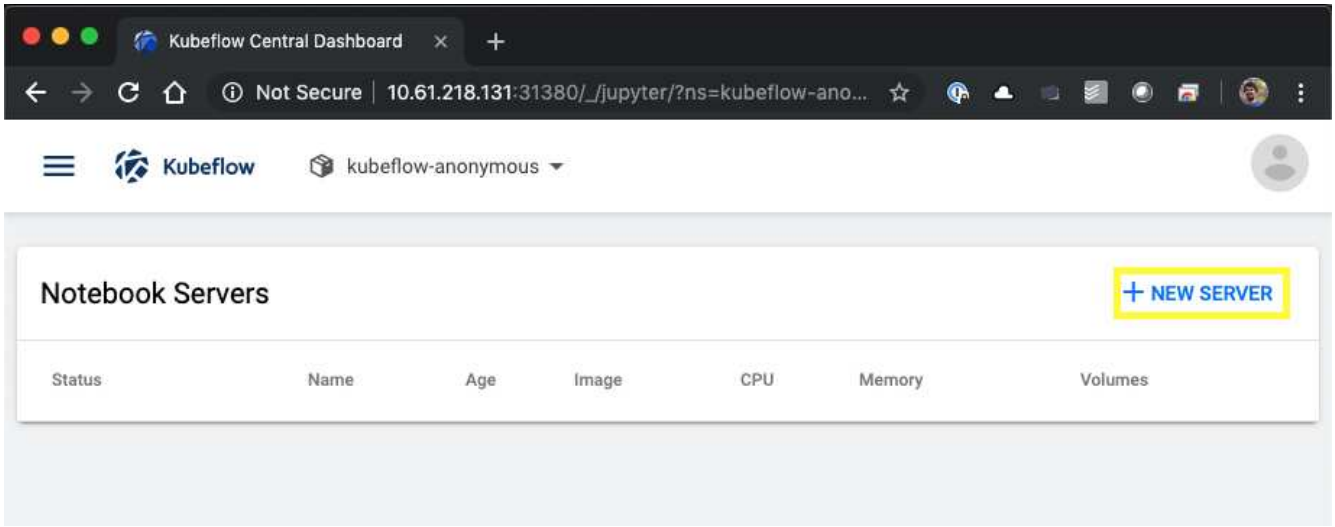
データサイエンティストまたは開発者向けに **Jupyter Notebook Workspace** をプロビジョニングします 使用

Kubeflow は、データサイエンティストのワークスペースとして機能する、新しい Jupyter Notebook サーバの迅速なプロビジョニングを可能にします。Kubeflow を使用して新しい Jupyter Notebook サーバをプロビジョニングするには、次のタスクを実行します。Kubeflow コンテキスト内の Jupyter Notebook の詳細については、を参照してください "Kubeflow の公式ドキュメント"。

1. Kubeflow 中央ダッシュボードのメインメニューで Notebook Servers をクリックして、 Jupyter Notebook サーバ管理ページに移動します。

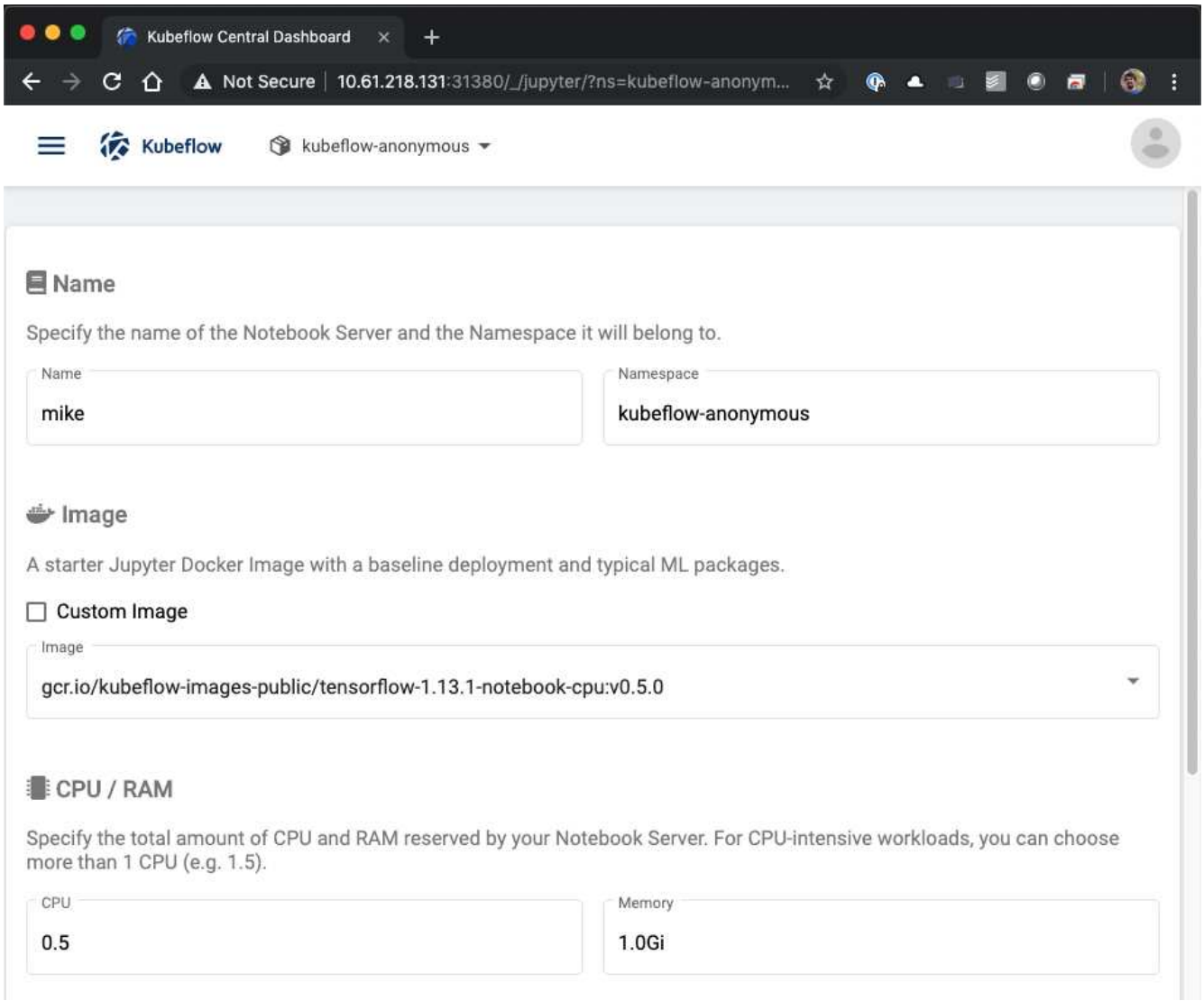


2. 新しい Jupyter Notebook サーバをプロビジョニングするには、 New Server をクリックします。

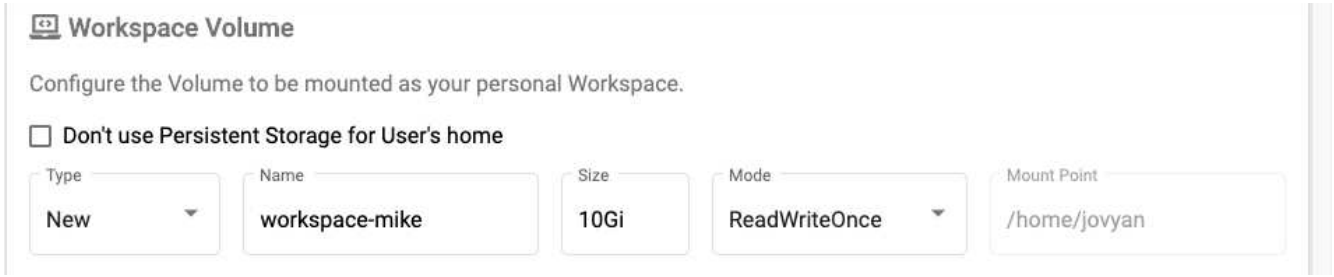


3. 新しいサーバに名前を付け、サーバのベースにする Docker イメージを選択し、サーバで予約する CPU と RAM の容量を指定します。[名前空間] フィールドが空白の場合は、ページヘッダーの [名前空間の選択] メニューを使用して名前空間を選択します。選択したネームスペースがネームスペースフィールドに自動的に入力されます。

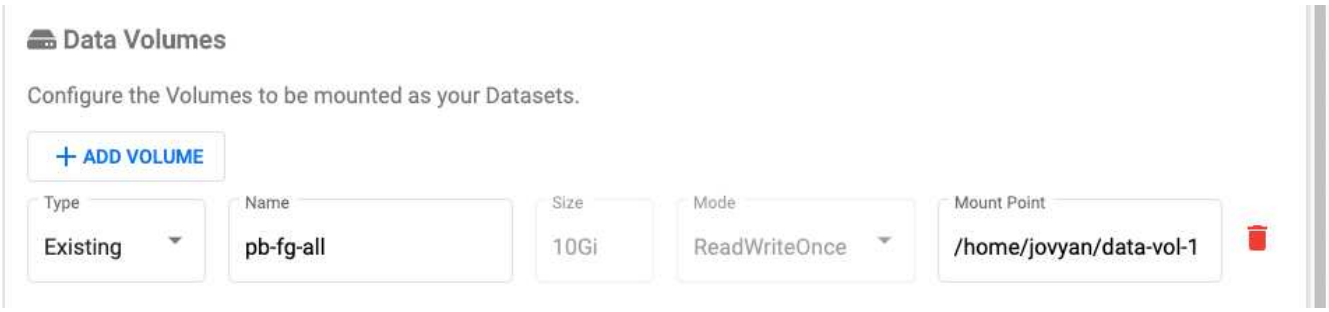
   次の例では 'kubeflow-anonymous' ネームスペースが選択されていますまた、 Docker イメージ、 CPU 、 RAM のデフォルト値も使用できます。



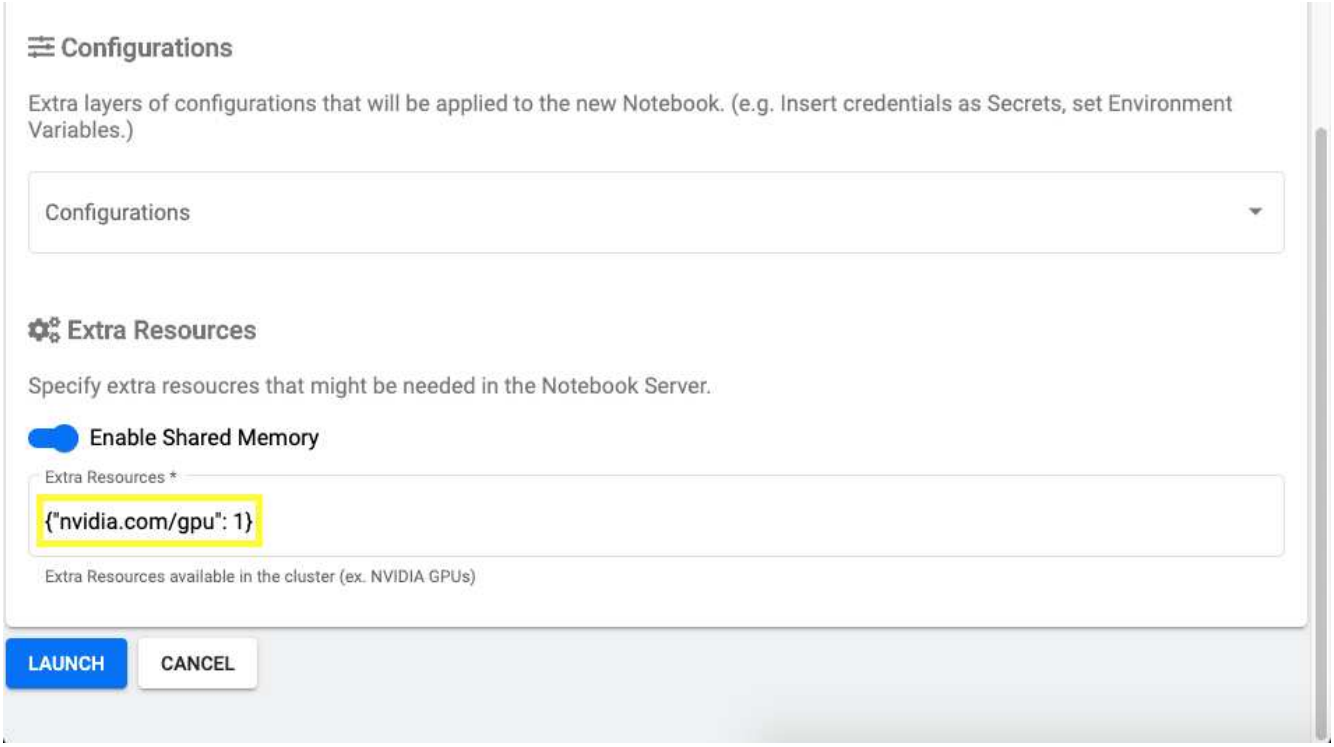
4. ワークスペースボリュームの詳細を指定します。新しいボリュームを作成するように選択した場合は、そのボリュームまたは PVC がデフォルトの StorageClass を使用してプロビジョニングされます。Trident を利用するストレージクラスがデフォルトとして指定されているため StorageClass "Kubeflow の導入"を指定した場合、ボリュームまたは PVC は Trident でプロビジョニングされます。このボリュームは、Jupyter Notebook Server コンテナ内のデフォルトワークスペースとして自動的にマウントされます。別のデータボリュームに保存されていないユーザーがサーバー上に作成したノートブックは、自動的にこのワークスペースボリュームに保存されます。そのため、ノートブックは再起動後も維持されます。



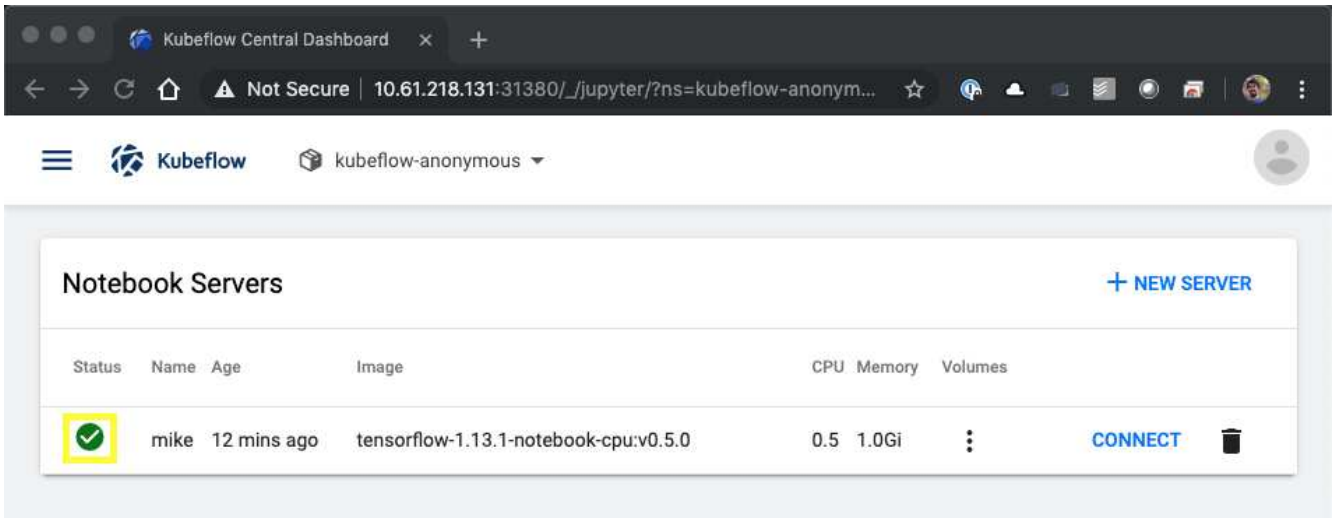
5. データボリュームを追加次に、「 pb-fg-all 」という名前の既存の PVC を指定し、デフォルトのマウントポイントを受け入れる例を示します。



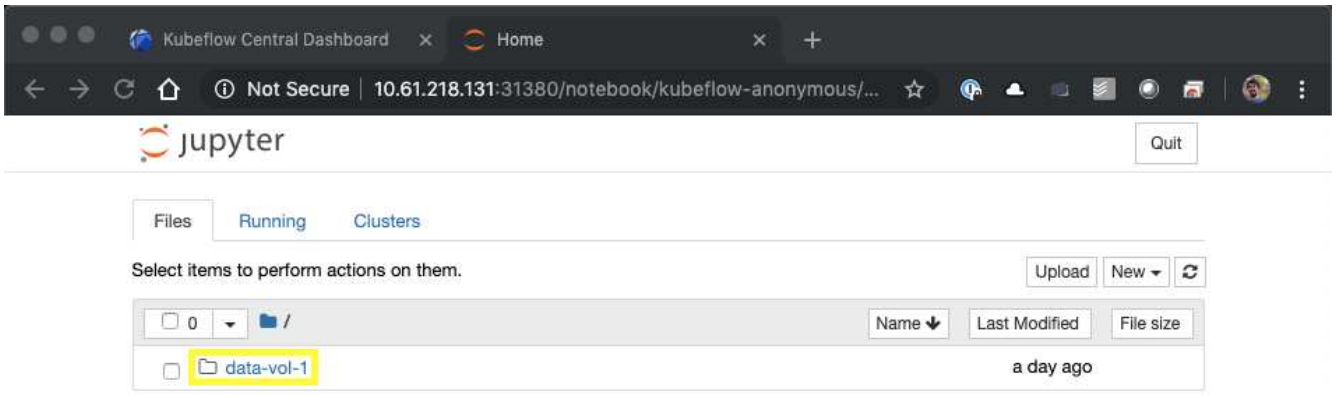
6. * オプション： * 希望する数の GPU をノートブックサーバーに割り当てるよう要求します。次の例では、GPU が 1 つ要求されています。

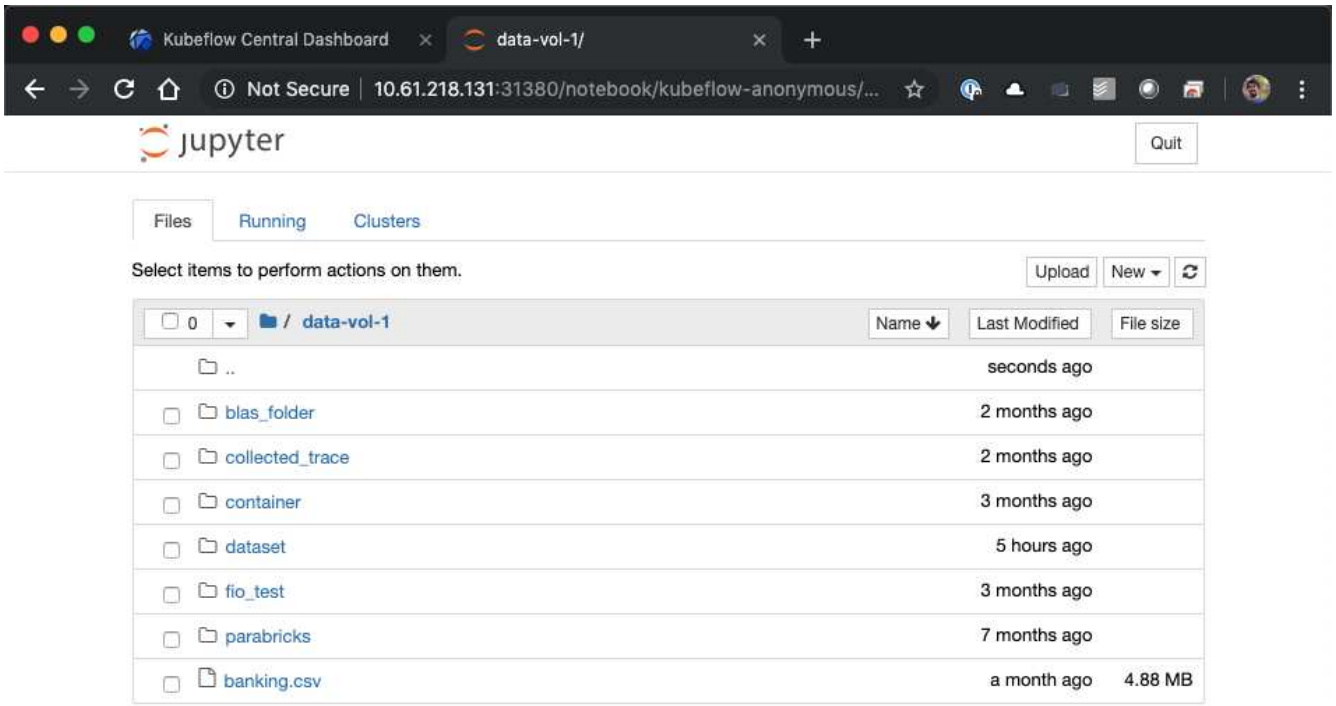


7. [起動] をクリックして、新しいノートブックサーバーをプロビジョニングします。
8. ノートブックサーバーが完全にプロビジョニングされるまで待ちます。指定した Docker イメージを使用してサーバをプロビジョニングしたことがない場合は、イメージのダウンロードが必要になるため、これには数分かかることがあります。サーバーが完全にプロビジョニングされると、 Jupyter Notebook サーバー管理ページの [ステータス] 列に緑色のチェックマークが表示されます。



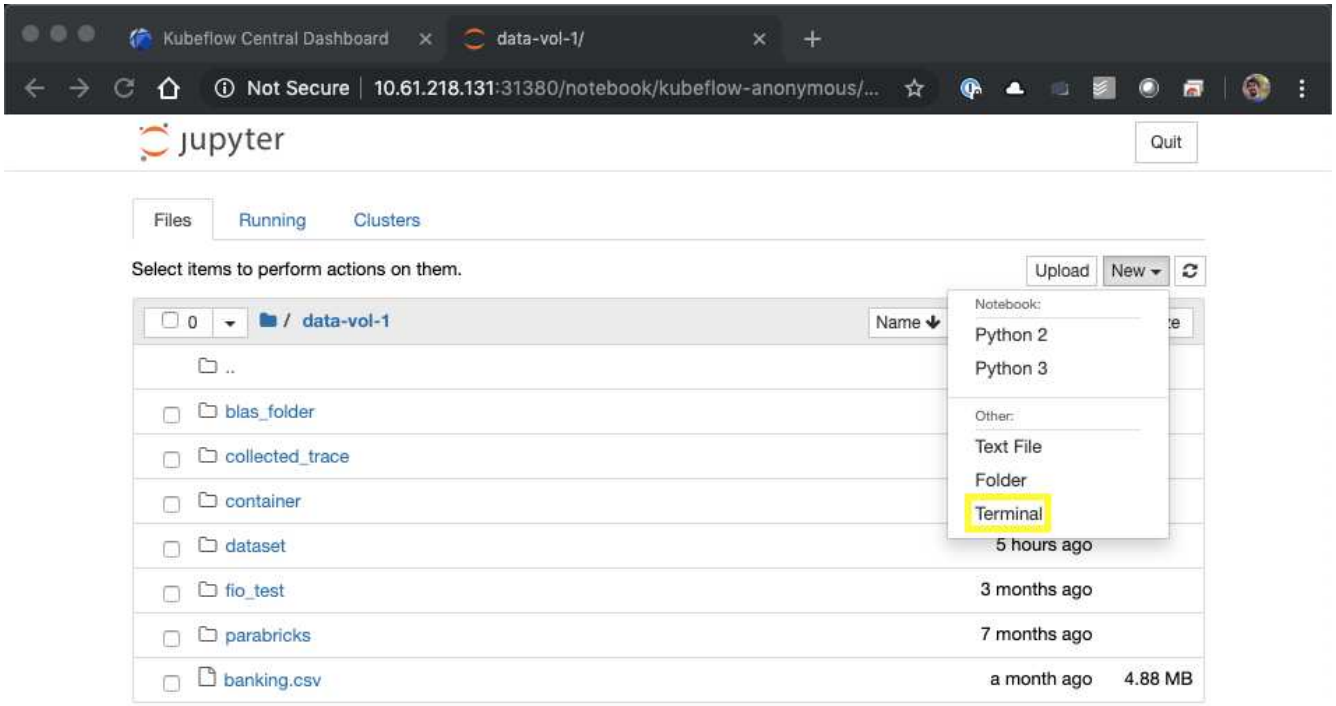
9. [接続] をクリックして、新しいサーバー Web インターフェイスに接続します。
10. 手順 6 で指定したデータセットボリュームがサーバにマウントされていることを確認します。デフォルトでは、このボリュームはデフォルトのワークスペース内にマウントされます。ユーザーの観点から見ると、これはワークスペース内の別のフォルダーにすぎません。データサイエンティストで、インフラのエキスパートではないユーザは、このボリュームを使用するためにストレージの専門知識を持っている必要はありません。

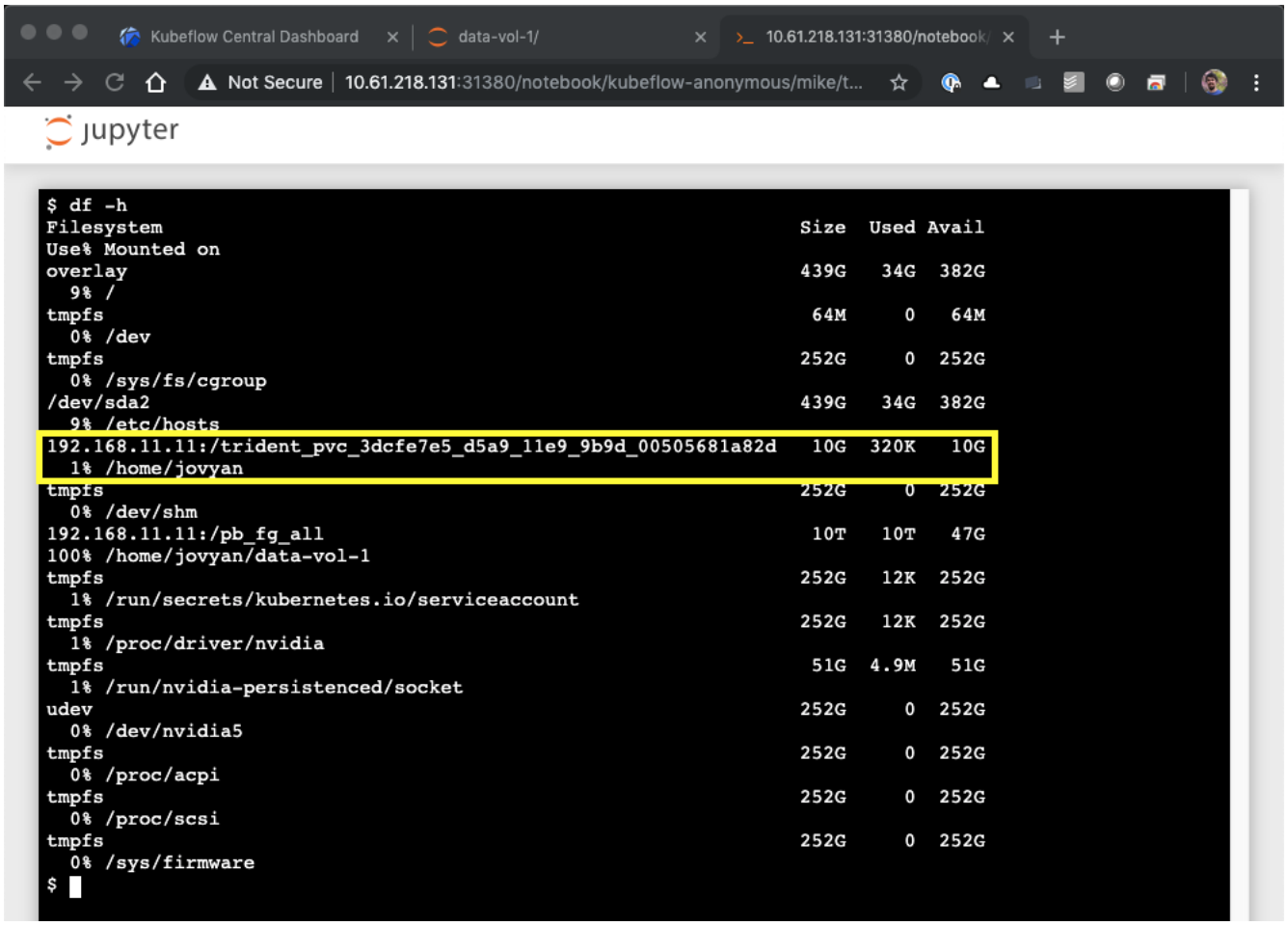




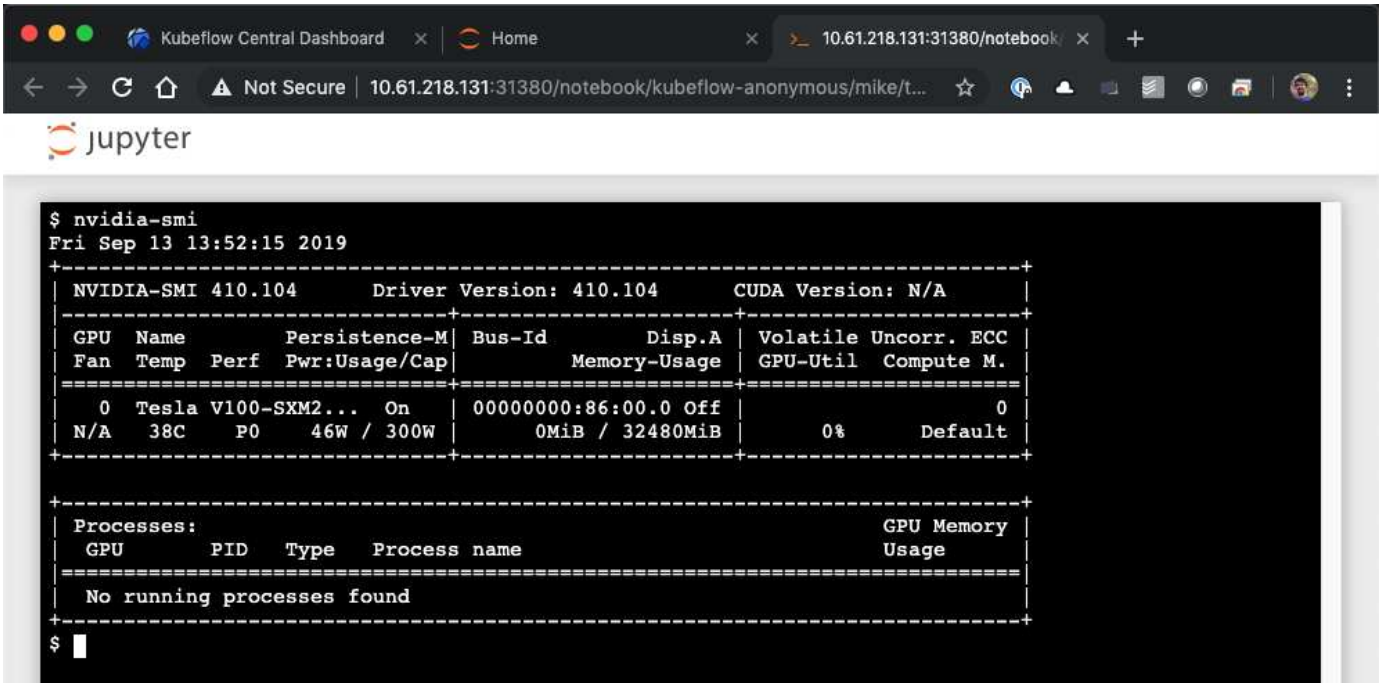
11. ターミナルを開き、手順 5 で新しいボリュームが要求された場合は、「 d f -h 」を実行して、 Trident でプロビジョニングされた新しい永続ボリュームがデフォルトのワークスペースとしてマウントされていることを確認します。

    デフォルトのワークスペースディレクトリは、サーバーの Web インターフェイスに最初にアクセスしたときに表示されるベースディレクトリです。そのため、 Web インターフェイスを使用して作成したアーティファクトは、 Trident でプロビジョニングされた永続ボリュームに保存されます。





12. ターミナルを使用して「 nvidia-smi 」を実行し、ノートブックサーバーに正しい数の GPU が割り当てられていることを確認します。次の例では、手順 7 で要求されたように、 1 つの GPU がノートブックサーバーに割り当てられています。



。 "NetApp Data Science Toolkit for Kubernetes" Kubeflow と組み合わせて使用できます。NetApp Data Science Toolkit と Kubeflow を使用すると、次のようなメリットがあります。

- データサイエンティストは、 Jupyter Notebook 内から、ネットアップの高度なデータ管理操作を直接実行できます。
- Kubeflow Pipelines フレームワークを使用すると、ネットアップの高度なデータ管理処理を自動化されたワークフローに組み込むことができます。

を参照してください "Kubeflow の例" ツールキットと Kubeflow を使用する場合の詳細については、 NetApp Data Science Toolkit GitHub リポジトリのセクションを参照してください。

Apache Airflow の導入

Kubernetes の上で Apache の通気を確保することを推奨します。このセクションでは、Kubernetes クラスタ内に通気を導入するために完了しておく必要のあるタスクについて説明します。

> Kubernetes 以外のプラットフォームに通気を導入することも可能です。Kubernetes 以外のプラットフォームに通気を導入することは、この解決策の範囲外です。

前提条件

このセクションで説明する導入の演習を行う前に、次の作業をすでに実行していることを前提としています。

1. Kubernetes クラスタをすでに使用している。
2. 「 NetApp Trident の導入と構成」のセクションに記載されているように、 Kubernetes クラスタに NetApp Trident をインストールし、設定しておきます。

Helm をインストールします

エアフローは、 Kubernetes の一般的なパッケージマネージャである Helm を使用して導入されます。エアーフローを導入する前に、導入ジャンプホストに Helm をインストールする必要があります。Helm を配置ジャンプホストにインストールするには、に従ってください "インストール手順" Helm の公式ドキュメントを参照してください。

デフォルトの Kubernetes StorageClass を設定します

通気を導入する前に、 Kubernetes クラスタ内にデフォルトのストレージクラスを指定する必要があります。エアフロー導入プロセスでは、デフォルトのストレージクラスを使用して新しい永続ボリュームのプロビジョニングが試行されます。StorageClass がデフォルトの StorageClass として指定されていない場合、導入は失敗します。クラスタ内でデフォルトの StorageClass を指定するには、の手順に従ってください "Kubeflow の導入"。クラスタ内ですでにデフォルトの StorageClass を指定している場合は、この手順を省略できます。

Helm を使用してエアフローを展開します

Helm を使用して Kubernetes クラスタに通気を導入するには、導入ジャンプホストから次のタスクを実行します。

1. Helm を使用してエアフローを導入します。に従ってください "導入手順" アーティファクトハブの公式エアフロー図については、を参照してください。以下のコマンド例は、 Helm を使用したエアーフローの配置を示しています。必要に応じて 'custom-values/yaml ファイルの値を変更 ' 追加 ' または削除します

```
$ cat << EOF > custom-values.yaml
###################################
# Airflow - Common Configs
###################################
airflow:
  ## the airflow executor type to use
  ##
  executor: "CeleryExecutor"
  ## environment variables for the web/scheduler/worker Pods (for
airflow configs)
  ##
  #
###################################
# Airflow - WebUI Configs
###################################
web:
  ## configs for the Service of the web Pods
  ##
  service:
    type: NodePort
###################################
# Airflow - Logs Configs
###################################
logs:
  persistence:
    enabled: true
###################################
# Airflow - DAGs Configs
###################################
dags:
  ## configs for the DAG git repository & sync container
  ##
  gitSync:
    enabled: true
    ## url of the git repository
    ##
    repo: "git@github.com:mboglesby/airflow-dev.git"
    ## the branch/tag/sha1 which we clone
    ##
    branch: master
    revision: HEAD
    ## the name of a pre-created secret containing files for ~/.ssh/
```

```
    ##
    ## NOTE:
    ## - this is ONLY RELEVANT for SSH git repos
    ## - the secret commonly includes files: id_rsa, id_rsa.pub,
known_hosts
    ## - known_hosts is NOT NEEDED if `git.sshKeyscan` is true
    ##
    sshSecret: "airflow-ssh-git-secret"
    ## the name of the private key file in your `git.secret`
    ##
    ## NOTE:
    ## - this is ONLY RELEVANT for PRIVATE SSH git repos
    ##
    sshSecretKey: id_rsa
    ## the git sync interval in seconds
    ##
    syncWait: 60
EOF
$ helm install airflow airflow-stable/airflow -n airflow --version 8.0.8
--values ./custom-values.yaml
...
Congratulations. You have just deployed Apache Airflow!
1. Get the Airflow Service URL by running these commands:
   export NODE_PORT=$(kubectl get --namespace airflow -o
jsonpath="{.spec.ports[0].nodePort}" services airflow-web)
   export NODE_IP=$(kubectl get nodes --namespace airflow -o
jsonpath="{.items[0].status.addresses[0].address}")
   echo http://$NODE_IP:$NODE_PORT/
2. Open Airflow in your web browser
```

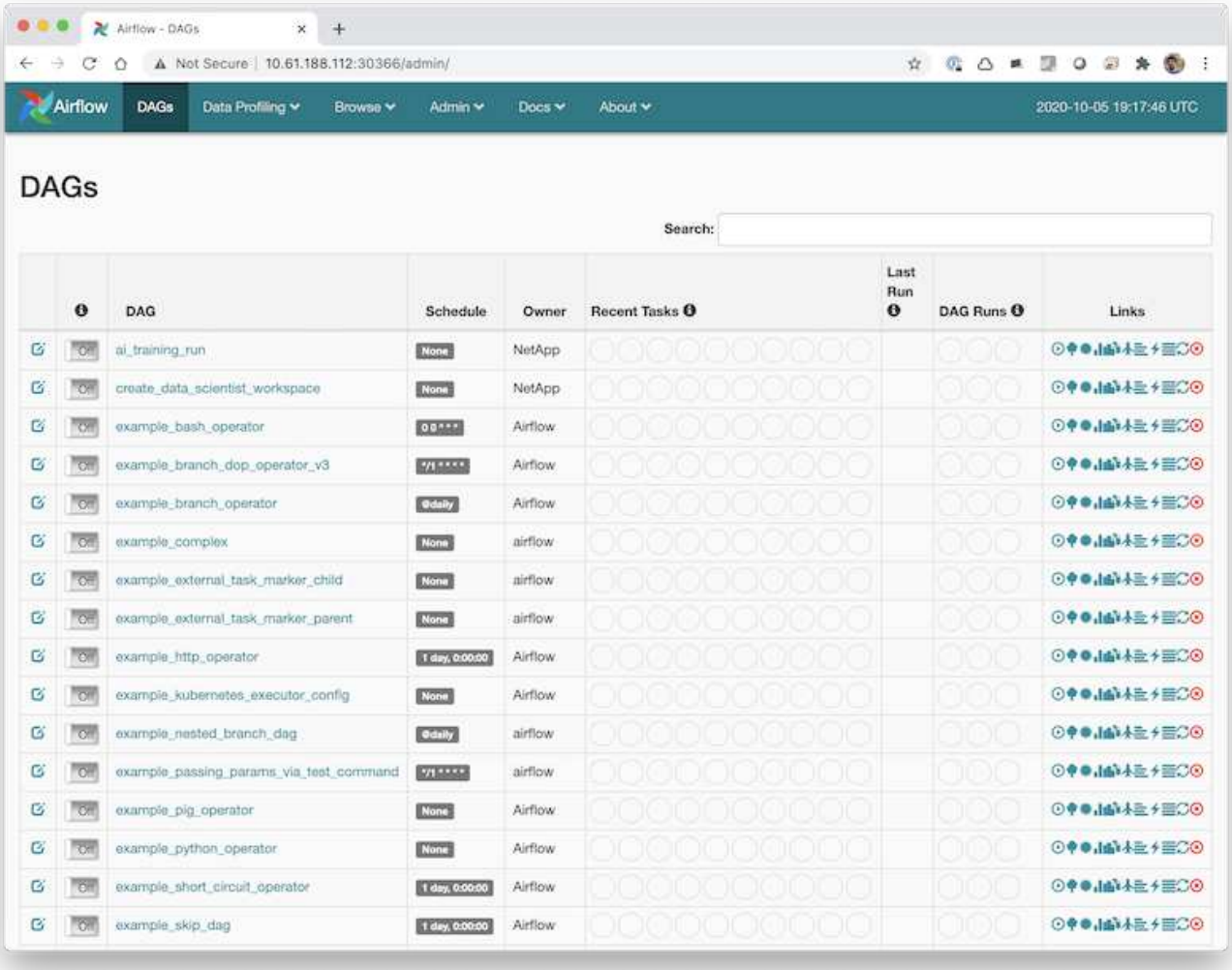
2. すべての通気ポッドが稼働中であることを確認します。すべてのポッドが起動するまでに数分かかる場合があります。

```
$ kubectl -n airflow get pod
NAME                                READY   STATUS    RESTARTS   AGE
airflow-flower-b5656d44f-h8qjk      1/1     Running   0          2h
airflow-postgresql-0                1/1     Running   0          2h
airflow-redis-master-0              1/1     Running   0          2h
airflow-scheduler-9d95fcdf9-clf4b   2/2     Running   2          2h
airflow-web-59c94db9c5-z7rg4        1/1     Running   0          2h
airflow-worker-0                    2/2     Running   2          2h
```

3. 手順 1 の Helm を使用してエアーフローを導入したときにコンソールに出力された指示に従って、エアーフロー Web サービスの URL を取得します。

```
$ export NODE_PORT=$(kubectl get --namespace airflow -o
jsonpath="{.spec.ports[0].nodePort}" services airflow-web)
$ export NODE_IP=$(kubectl get nodes --namespace airflow -o
jsonpath="{.items[0].status.addresses[0].address}")
$ echo http://$NODE_IP:$NODE_PORT/
```

4. 通気 Web サービスにアクセスできることを確認します。



Apache の通気ワークフローの例

。 "NetApp Data Science Toolkit for Kubernetes" エアーフローと組み合わせて使用できます。NetApp Data Science Toolkit とエアフローを組み合わせることで、エアフローによって自動化されたワークフローにネットアップのデータ管理操作を組み込むことができます。

を参照してください "通気の例" ツールキットの通気と使用方法の詳細については、 NetApp Data Science Toolkit GitHub リポジトリのセクションを参照してください。

このセクションでは、 Trident で実行したいさまざまな処理の例を紹介します。

既存のボリュームをインポートします

ネットアップストレージシステム / プラットフォーム上に Kubernetes クラスタ内のコンテナにマウントする必要のある既存のボリュームがあり、クラスタ内の PVC に関連付けられていない場合は、それらのボリュームをインポートする必要があります。これらのボリュームは、 Trident のボリュームインポート機能を使用してインポートできます。

以降のコマンド例では、「 pb_fg_all 」という名前の同じボリュームを、セクションの例で作成した各 Trident バックエンドに対して 1 回ずつ、 2 回インポートしています "ONTAP AI 導入向け Trident バックエンドの例"、手順 1.同じボリュームをこの 2 つの方法でインポートすると、（既存の FlexGroup ボリューム）を複数の LIF にまたがって複数回マウントできます。詳細については、セクションを参照してください "ONTAP AI 導入向け Trident バックエンドの例"、手順 1.PVC の詳細については、を参照してください "Kubernetes の公式ドキュメント"。ボリュームインポート機能の詳細については、を参照してください "Trident のドキュメント"。

「 accessModes 」の値「 ReadOnlyMany 」は、 PVC 仕様ファイルの例で指定されています。「 accessMode 」フィールドの詳細については、を参照してください "Kubernetes の公式ドキュメント"。

> 次の例で指定するバックエンド名 インポートコマンドは、で作成されたバックエンドに対応します の例を参照してください "ONTAP AI 導入向け Trident バックエンドの例"、手順 1.次の例で指定した StorageClass 名 PVC 定義ファイルは、作成された StorageClasses に対応しています を参照してください "ONTAP AI 導入向けの Kubernetes StorageClasses の例"、手順 1.

```
$ cat << EOF > ./pvc-import-pb_fg_all-iface1.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pb-fg-all-iface1
  namespace: default
spec:
  accessModes:
    - ReadOnlyMany
  storageClassName: ontap-ai-flexgroups-retain-iface1
EOF
$ tridentctl import volume ontap-ai-flexgroups-iface1 pb_fg_all -f ./pvc-
import-pb_fg_all-iface1.yaml -n trident
+--------------------------------+--------
+-----------------------------------+----------
+-----------------------------------------------+--------+----------+
|           NAME                   |  SIZE  |        STORAGE CLASS
| PROTOCOL |              BACKEND UUID                          | STATE   |
MANAGED |
+--------------------------------+--------
+-----------------------------------+----------
+-----------------------------------------------+--------+----------+
```

```
| default-pb-fg-all-iface1-7d9f1 | 10 TiB | ontap-ai-flexgroups-retain-
iface1 | file     | b74cbddb-e0b8-40b7-b263-b6da6dec0bdd | online | true
|
+--------------------------------+--------
+-----------------------------------+----------
+-------------------------------------------+--------+---------+
$ cat << EOF > ./pvc-import-pb_fg_all-iface2.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pb-fg-all-iface2
  namespace: default
spec:
  accessModes:
    - ReadOnlyMany
  storageClassName: ontap-ai-flexgroups-retain-iface2
EOF
$ tridentctl import volume ontap-ai-flexgroups-iface2 pb_fg_all -f ./pvc-
import-pb_fg_all-iface2.yaml -n trident
+--------------------------------+--------
+-----------------------------------+----------
+-------------------------------------------+--------+---------+
|          NAME                  |  SIZE  |       STORAGE CLASS
| PROTOCOL |             BACKEND UUID                        | STATE  |
MANAGED |
+--------------------------------+--------
+-----------------------------------+----------
+-------------------------------------------+--------+---------+
| default-pb-fg-all-iface2-85aee | 10 TiB | ontap-ai-flexgroups-retain-
iface2 | file     | 61814d48-c770-436b-9cb4-cf7ee661274d | online | true
|
+--------------------------------+--------
+-----------------------------------+----------
+-------------------------------------------+--------+---------+
$ tridentctl get volume -n trident
+-----------------------------------+---------
+-----------------------------------+----------
+--------------------------------------+--------+---------+
|               NAME               |  SIZE   |           STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+-----------------------------------+---------
+-----------------------------------+----------
+--------------------------------------+--------+---------+
| default-pb-fg-all-iface1-7d9f1   | 10 TiB  | ontap-ai-flexgroups-retain-
iface1 | file     | b74cbddb-e0b8-40b7-b263-b6da6dec0bdd | online | true
|
```

```
| default-pb-fg-all-iface2-85aee   | 10 TiB  | ontap-ai-flexgroups-retain-
iface2 | file     | 61814d48-c770-436b-9cb4-cf7ee661274d | online | true
|
+-----------------------------------+---------
+------------------------------------+----------
+--------------------------------------+--------+---------+
$ kubectl get pvc
NAME                  STATUS   VOLUME                                CAPACITY
ACCESS MODES   STORAGECLASS                        AGE
pb-fg-all-iface1      Bound    default-pb-fg-all-iface1-7d9f1
10995116277760   ROX            ontap-ai-flexgroups-retain-iface1   25h
pb-fg-all-iface2      Bound    default-pb-fg-all-iface2-85aee
10995116277760   ROX            ontap-ai-flexgroups-retain-iface2   25h
```

新しいボリュームをプロビジョニングします

Trident を使用して、ネットアップストレージシステムまたはプラットフォームで新しいボリュームをプロビジョニングできます。次のコマンド例は、新しい FlexVol ボリュームのプロビジョニングを表示します。この例では、セクションの例で作成した StorageClass を使用してボリュームがプロビジョニングされます "ONTAP AI 導入向けの Kubernetes StorageClasses の例"、ステップ 2 。

次の PVC 定義ファイル例では 'accessModes' の値 ReadWriteMany が指定されています「 accessMode 」フィールドの詳細については、を参照してください "Kubernetes の公式ドキュメント"。

```
$ cat << EOF > ./pvc-tensorflow-results.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: tensorflow-results
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-ai-flexvols-retain
EOF
$ kubectl create -f ./pvc-tensorflow-results.yaml
persistentvolumeclaim/tensorflow-results created
$ kubectl get pvc
NAME                                STATUS    VOLUME
CAPACITY         ACCESS MODES   STORAGECLASS                        AGE
pb-fg-all-iface1                    Bound     default-pb-fg-all-iface1-7d9f1
10995116277760   ROX            ontap-ai-flexgroups-retain-iface1   26h
pb-fg-all-iface2                    Bound     default-pb-fg-all-iface2-85aee
10995116277760   ROX            ontap-ai-flexgroups-retain-iface2   26h
tensorflow-results                  Bound     default-tensorflow-results-
2fd60   1073741824       RWX            ontap-ai-flexvols-retain
25h
```

ONTAP AI 導入向けハイパフォーマンスジョブの例

このセクションでは、 ONTAP AI ポッドに Kubernetes を導入する際に実行可能なさまざまなハイパフォーマンスジョブの例を示します。

ONTAP AI 導入向けハイパフォーマンスジョブの例

このセクションでは、 ONTAP AI ポッドに Kubernetes を導入する際に実行可能なさまざまなハイパフォーマンスジョブの例を示します。

シングルノードの AI ワークロードを実行

Kubernetes クラスタでシングルノードの AI ジョブと ML ジョブを実行するには、導入ジャンプホストから次のタスクを実行します。Trident を使用すると、数ペタバイトのデータが含まれる可能性のあるデータボリュームをすばやく簡単に作成し、 Kubernetes のワークロードからアクセスできます。Kubernetes ポッド内からこのようなデータボリュームにアクセスできるようにするには、ポッドの定義で PVC を指定します。このステップは Kubernetes ネイティブの運用であり、ネットアップの専門知識は不要です。

このセクションでは、 Kubernetes クラスタで実行しようとしている特定の AI および ML ワークロードを（ Docker コンテナ形式で）コンテナ化済みであることを前提としています。

1. 次のコマンド例は、 ImageNet データセットを使用する TensorFlow ベンチマークワークロード用の Kubernetes ジョブを作成する方法を示しています。ImageNet データセットの詳細については、を参照してください "ImageNet の Web サイト"。

   このジョブ例では、 8 個の GPU を要求するため、 8 個以上の GPU を搭載した 1 つの GPU ワーカーノードで実行することができます。このジョブ例は、 8 個以上の GPU を搭載したワーカーノードが存在しない、または現在別のワークロードを使用しているクラスタで送信できます。その場合、そのようなワーカーノードが使用可能になるまで、ジョブは保留状態のままになります。

   また、ストレージ帯域幅を最大限にするために、必要なトレーニングデータを含むボリュームが、このジョブで作成されるポッド内に 2 回マウントされます。ポッドには別のボリュームもマウントされています。この 2 つ目のボリュームには、結果と指標を格納します。これらのボリュームは、 PVC の名前を使用してジョブ定義内で参照されます。Kubernetes ジョブの詳細については、を参照してください "Kubernetes の公式ドキュメント"。

   「 M emory 」の値が「 emory 」である「 emptyDir 」ボリュームは、この例のジョブで作成されるポッド内の「 /dev/shm 」にマウントされます。Docker コンテナランタイムによって自動的に作成される「 /dev/shm 」仮想ボリュームのデフォルトサイズが、 TensorFlow のニーズに十分でない場合があります。次の例のように 'emptyDir' ボリュームをマウントすると '/dev/shm' 仮想ボリュームが十分に大きくなります「 emptyDir 」ボリュームの詳細については、を参照してください "Kubernetes の公式ドキュメント"。

   この例のジョブ定義で指定されている単一のコンテナには 'ecurityContext> 特権値 'true' が与えられますこの値は、コンテナにホスト上のルートアクセス権があることを意味します。このアノテーションは、実行中の特定のワークロードにルートアクセスが必要なために使用されます。具体的には、ワークロードで実行されるクリアキャッシュ処理にはルートアクセスが必要です。これが特権 : true の注釈であるかどうかは ' 実行している特定のワークロードの要件によって異なります

```
$ cat << EOF > ./netapp-tensorflow-single-imagenet.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: netapp-tensorflow-single-imagenet
spec:
  backoffLimit: 5
  template:
    spec:
      volumes:
      - name: dshm
        emptyDir:
          medium: Memory
      - name: testdata-iface1
        persistentVolumeClaim:
          claimName: pb-fg-all-iface1
      - name: testdata-iface2
        persistentVolumeClaim:
          claimName: pb-fg-all-iface2
```

```
      - name: results
        persistentVolumeClaim:
          claimName: tensorflow-results
      containers:
      - name: netapp-tensorflow-py2
        image: netapp/tensorflow-py2:19.03.0
        command: ["python", "/netapp/scripts/run.py", "--
dataset_dir=/mnt/mount_0/dataset/imagenet", "--dgx_version=dgx1", "--
num_devices=8"]
        resources:
          limits:
            nvidia.com/gpu: 8
        volumeMounts:
        - mountPath: /dev/shm
          name: dshm
        - mountPath: /mnt/mount_0
          name: testdata-iface1
        - mountPath: /mnt/mount_1
          name: testdata-iface2
        - mountPath: /tmp
          name: results
        securityContext:
          privileged: true
      restartPolicy: Never
EOF
$ kubectl create -f ./netapp-tensorflow-single-imagenet.yaml
job.batch/netapp-tensorflow-single-imagenet created
$ kubectl get jobs
NAME                                       COMPLETIONS   DURATION   AGE
netapp-tensorflow-single-imagenet          0/1           24s        24s
```

2. 手順 1 で作成したジョブが正しく実行されていることを確認します。次のコマンド例では、ジョブ定義で指定したとおりにジョブ用にポッドが 1 つ作成され、このポッドが GPU ワーカーノードの 1 つで現在実行されていることを確認します。

```
$ kubectl get pods -o wide
NAME                                              READY   STATUS
RESTARTS   AGE
IP              NODE            NOMINATED NODE
netapp-tensorflow-single-imagenet-m7x92           1/1     Running     0
3m    10.233.68.61    10.61.218.154   <none>
```

3. 手順 1 で作成したジョブが正常に完了したことを確認します。次のコマンド例は、ジョブが正常に完了したことを確認します。

```
$ kubectl get jobs
NAME                                              COMPLETIONS   DURATION
AGE
netapp-tensorflow-single-imagenet                 1/1           5m42s
10m
$ kubectl get pods
NAME                                                   READY   STATUS
RESTARTS   AGE
netapp-tensorflow-single-imagenet-m7x92                0/1     Completed
0          11m
$ kubectl logs netapp-tensorflow-single-imagenet-m7x92
[netapp-tensorflow-single-imagenet-m7x92:00008] PMIX ERROR: NO-
PERMISSIONS in file gds_dstore.c at line 702
[netapp-tensorflow-single-imagenet-m7x92:00008] PMIX ERROR: NO-
PERMISSIONS in file gds_dstore.c at line 711
Total images/sec = 6530.59125
================ Clean Cache !!! ==================
mpirun -allow-run-as-root -np 1 -H localhost:1 bash -c 'sync; echo 1 >
/proc/sys/vm/drop_caches'
==========================================
mpirun -allow-run-as-root -np 8 -H localhost:8 -bind-to none -map-by
slot -x NCCL_DEBUG=INFO -x LD_LIBRARY_PATH -x PATH python
/netapp/tensorflow/benchmarks_190205/scripts/tf_cnn_benchmarks/tf_cnn_be
nchmarks.py --model=resnet50 --batch_size=256 --device=gpu
--force_gpu_compatible=True --num_intra_threads=1 --num_inter_threads=48
--variable_update=horovod --batch_group_size=20 --num_batches=500
--nodistortions --num_gpus=1 --data_format=NCHW --use_fp16=True
--use_tf_layers=False --data_name=imagenet --use_datasets=True
--data_dir=/mnt/mount_0/dataset/imagenet
--datasets_parallel_interleave_cycle_length=10
--datasets_sloppy_parallel_interleave=False --num_mounts=2
--mount_prefix=/mnt/mount_%d --datasets_prefetch_buffer_size=2000
--datasets_use_prefetch=True --datasets_num_private_threads=4
--horovod_device=gpu >
/tmp/20190814_105450_tensorflow_horovod_rdma_resnet50_gpu_8_256_b500_ima
genet_nodistort_fp16_r10_m2_nockpt.txt 2>&1
```

4. * オプション： * ジョブアーティファクトをクリーンアップします。次のコマンド例は、手順 1 で作成したジョブオブジェクトの削除を示しています。

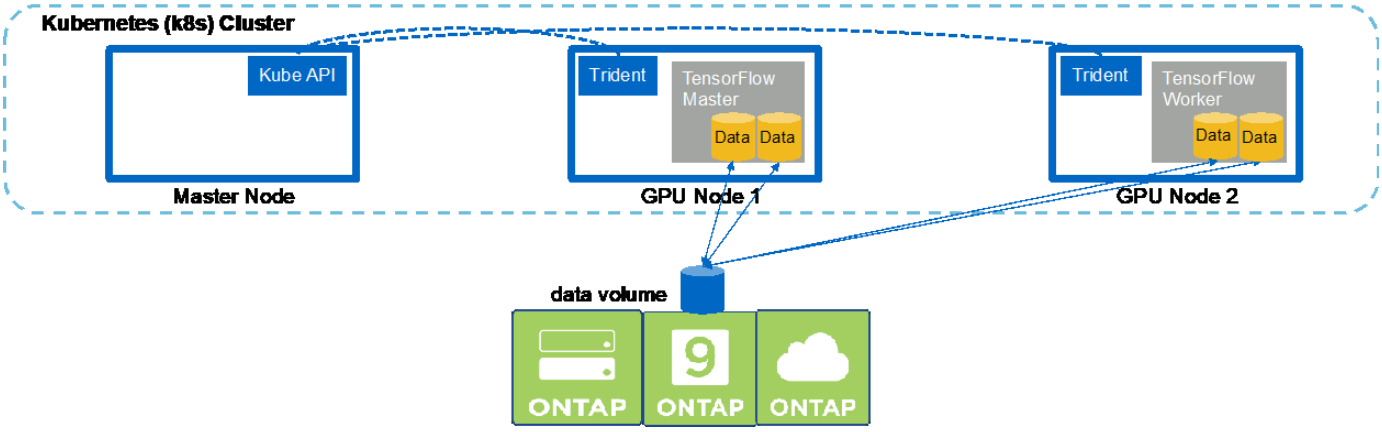
   ジョブオブジェクトを削除すると、関連付けられているポッドは Kubernetes によって自動的に削除されます。

```
$ kubectl get jobs
NAME                                                COMPLETIONS   DURATION
AGE
netapp-tensorflow-single-imagenet                   1/1           5m42s
10m
$ kubectl get pods
NAME                                                       READY   STATUS
RESTARTS   AGE
netapp-tensorflow-single-imagenet-m7x92                    0/1     Completed
0          11m
$ kubectl delete job netapp-tensorflow-single-imagenet
job.batch "netapp-tensorflow-single-imagenet" deleted
$ kubectl get jobs
No resources found.
$ kubectl get pods
No resources found.
```

分散型 **AI** の同期ワークロードを実行します

Kubernetes クラスタでマルチノードの AI と ML の同期ジョブを実行するには、導入ジャンプホストで次のタスクを実行します。このプロセスにより、ネットアップボリュームに格納されているデータを活用し、 1 つのワーカーノードで提供されるものよりも多くの GPU を使用することができます。同期分散 AI ジョブの説明については、次の図を参照してください。

> 同期分散ジョブを使用すると、非同期分散ジョブに比べてパフォーマンスとトレーニングの精度が向上します。同期ジョブと非同期ジョブの長所と短所については、本ドキュメントでは説明していません。



1. 次のコマンド例は、 1 つのワーカーの作成を示しています これは、同じの同期分散実行に関与します 1 つのノードで実行された TensorFlow ベンチマークジョブ を参照してください "シングルノードの AI ワークロードを実行"。この例では、 2 つのワーカーノードでジョブが実行されるため、導入されるワーカーは 1 つだけです。

この例のワーカー導入では、 8 個の GPU を要求し、 8 個以上の GPU を搭載した 1 つの GPU ワーカーノードで実行できます。GPU ワーカーノードが 8 個以上の GPU を搭載している場合、パフォーマンスを最大化するには、この数をワーカーノードが機能する GPU の数と同じにすると便利です。Kubernetes の導入の詳細については、を参照してください "Kubernetes の公式ドキュメント"。

この例では、このコンテナ化された特定のワーカーが自分で完了することはないため、 Kubernetes 環境が作成されます。そのため、 Kubernetes のジョブ構造を使用して導入することは理にかなっていません。従業員が自分で設計または作成した場合、作業構成を使用して従業員を配置することが理にかなっている場合があります。

この配置例の仕様で指定されているポッドには 'hostNetwork' の値が true に設定されていますこの値は、ポッドが、 Kubernetes が各ポッドに通常作成する仮想ネットワーキングスタックではなく、ホストワーカーノードのネットワークスタックを使用することを意味します。このアノテーションは、特定のワークロードが Open MPI 、 NCCL 、 Horovod を使用して同期分散方法でワークロードを実行するために使用されます。そのため、ホストネットワークスタックにアクセスする必要があります。Open MPI 、 NCCL 、および Horovod についての説明は、本ドキュメントの範囲外です。この hostNetwork:true' 注釈が必要かどうかは ' 実行している特定のワークロードの要件によって決まります「 hostNetwork 」フィールドの詳細については、を参照してください "Kubernetes の公式ドキュメント"。

```
$ cat << EOF > ./netapp-tensorflow-multi-imagenet-worker.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: netapp-tensorflow-multi-imagenet-worker
spec:
  replicas: 1
  selector:
    matchLabels:
      app: netapp-tensorflow-multi-imagenet-worker
  template:
    metadata:
      labels:
        app: netapp-tensorflow-multi-imagenet-worker
    spec:
      hostNetwork: true
      volumes:
      - name: dshm
        emptyDir:
          medium: Memory
      - name: testdata-iface1
        persistentVolumeClaim:
          claimName: pb-fg-all-iface1
      - name: testdata-iface2
        persistentVolumeClaim:
          claimName: pb-fg-all-iface2
      - name: results
        persistentVolumeClaim:
          claimName: tensorflow-results
```

```
      containers:
      - name: netapp-tensorflow-py2
        image: netapp/tensorflow-py2:19.03.0
        command: ["bash", "/netapp/scripts/start-slave-multi.sh",
"22122"]
        resources:
          limits:
            nvidia.com/gpu: 8
        volumeMounts:
        - mountPath: /dev/shm
          name: dshm
        - mountPath: /mnt/mount_0
          name: testdata-iface1
        - mountPath: /mnt/mount_1
          name: testdata-iface2
        - mountPath: /tmp
          name: results
        securityContext:
          privileged: true
EOF
$ kubectl create -f ./netapp-tensorflow-multi-imagenet-worker.yaml
deployment.apps/netapp-tensorflow-multi-imagenet-worker created
$ kubectl get deployments
NAME                                      DESIRED   CURRENT   UP-TO-DATE
AVAILABLE   AGE
netapp-tensorflow-multi-imagenet-worker   1         1         1
1           4s
```

2. 手順 1 で作成したワーカー導入が正常に起動したことを確認します。次のコマンド例は、導入定義に示すように、単一のワーカーポッドが導入用に作成されたこと、およびこのポッドが GPU ワーカーノードの 1 つで現在実行されていることを確認します。

```
$ kubectl get pods -o wide
NAME                                                      READY
STATUS    RESTARTS   AGE
IP              NODE            NOMINATED NODE
netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725   1/1
Running   0          60s   10.61.218.154   10.61.218.154   <none>
$ kubectl logs netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725
22122
```

3. マルチノード同期ジョブの実行を開始して参加させ、追跡するマスター用の Kubernetes ジョブを作成します。次のコマンド例では、 1 つのマスターを作成します。このマスターは、セクションの例で 1 つのノード上で実行された、同じ TensorFlow ベンチマークジョブの同期分散実行を追跡し、開始します "シングルノードの AI ワークロードを実行"。

この例では、マスタージョブは 8 個の GPU を要求するため、 8 個以上の GPU を搭載した 1 つの GPU ワーカーノードで実行できます。GPU ワーカーノードが 8 個以上の GPU を搭載している場合、パフォーマンスを最大化するには、この数をワーカーノードが機能する GPU の数と同じにすると便利です。

この例のジョブ定義で指定されているマスターポッドには、手順 1 でワーカーポッドに「 hostNetwork 」の値「 true 」が与えられたのと同様に、「 hostNetwork 」の値が「 true 」に設定されます。この値が必要な理由については、手順 1 を参照してください。

```
$ cat << EOF > ./netapp-tensorflow-multi-imagenet-master.yaml
apiVersion: batch/v1
kind: Job
metadata:
  name: netapp-tensorflow-multi-imagenet-master
spec:
  backoffLimit: 5
  template:
    spec:
      hostNetwork: true
      volumes:
      - name: dshm
        emptyDir:
          medium: Memory
      - name: testdata-iface1
        persistentVolumeClaim:
          claimName: pb-fg-all-iface1
      - name: testdata-iface2
        persistentVolumeClaim:
          claimName: pb-fg-all-iface2
      - name: results
        persistentVolumeClaim:
          claimName: tensorflow-results
      containers:
      - name: netapp-tensorflow-py2
        image: netapp/tensorflow-py2:19.03.0
        command: ["python", "/netapp/scripts/run.py", "--
dataset_dir=/mnt/mount_0/dataset/imagenet", "--port=22122", "--
num_devices=16", "--dgx_version=dgx1", "--
nodes=10.61.218.152,10.61.218.154"]
        resources:
          limits:
            nvidia.com/gpu: 8
        volumeMounts:
        - mountPath: /dev/shm
          name: dshm
        - mountPath: /mnt/mount_0
          name: testdata-iface1
        - mountPath: /mnt/mount_1
```

```
          name: testdata-iface2
        - mountPath: /tmp
          name: results
        securityContext:
          privileged: true
      restartPolicy: Never
EOF
$ kubectl create -f ./netapp-tensorflow-multi-imagenet-master.yaml
job.batch/netapp-tensorflow-multi-imagenet-master created
$ kubectl get jobs
NAME                                        COMPLETIONS   DURATION   AGE
netapp-tensorflow-multi-imagenet-master   0/1           25s        25s
```

4. 手順 3 で作成したマスタージョブが正しく実行されていることを確認します。次のコマンド例では、ジョブ定義に示されているように、ジョブに対して単一のマスターポッドが作成され、このポッドが GPU ワーカーノードの 1 つで現在実行されていることを確認します。また、手順 1 で最初に確認したワーカーポッドがまだ実行中で、マスターポッドとワーカーポッドが別々のノードで実行されていることも確認する必要があります。

```
$ kubectl get pods -o wide
NAME                                                          READY
STATUS    RESTARTS   AGE
IP              NODE            NOMINATED NODE
netapp-tensorflow-multi-imagenet-master-ppwwj                 1/1
Running   0          45s   10.61.218.152   10.61.218.152   <none>
netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725   1/1
Running   0          26m   10.61.218.154   10.61.218.154   <none>
```

5. 手順 3 で作成したマスタージョブが正常に完了したことを確認します。次のコマンド例は、ジョブが正常に完了したことを確認します。

```
$ kubectl get jobs
NAME                                        COMPLETIONS   DURATION   AGE
netapp-tensorflow-multi-imagenet-master   1/1           5m50s      9m18s
$ kubectl get pods
NAME                                                          READY
STATUS      RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj                 0/1
Completed   0          9m38s
netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725   1/1
Running     0          35m
$ kubectl logs netapp-tensorflow-multi-imagenet-master-ppwwj
[10.61.218.152:00008] WARNING: local probe returned unhandled
shell:unknown assuming bash
rm: cannot remove '/lib': Is a directory
```

```
[10.61.218.154:00033] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at
line 702
[10.61.218.154:00033] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at
line 711
[10.61.218.152:00008] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at
line 702
[10.61.218.152:00008] PMIX ERROR: NO-PERMISSIONS in file gds_dstore.c at
line 711
Total images/sec = 12881.33875
================ Clean Cache !!! ==================
mpirun -allow-run-as-root -np 2 -H 10.61.218.152:1,10.61.218.154:1 -mca
pml ob1 -mca btl ^openib -mca btl_tcp_if_include enp1s0f0 -mca
plm_rsh_agent ssh -mca plm_rsh_args "-p 22122" bash -c 'sync; echo 1 >
/proc/sys/vm/drop_caches'
==========================================
mpirun -allow-run-as-root -np 16 -H 10.61.218.152:8,10.61.218.154:8
-bind-to none -map-by slot -x NCCL_DEBUG=INFO -x LD_LIBRARY_PATH -x PATH
-mca pml ob1 -mca btl ^openib -mca btl_tcp_if_include enp1s0f0 -x
NCCL_IB_HCA=mlx5 -x NCCL_NET_GDR_READ=1 -x NCCL_IB_SL=3 -x
NCCL_IB_GID_INDEX=3 -x
NCCL_SOCKET_IFNAME=enp5s0.3091,enp12s0.3092,enp132s0.3093,enp139s0.3094
-x NCCL_IB_CUDA_SUPPORT=1 -mca orte_base_help_aggregate 0 -mca
plm_rsh_agent ssh -mca plm_rsh_args "-p 22122" python
/netapp/tensorflow/benchmarks_190205/scripts/tf_cnn_benchmarks/tf_cnn_be
nchmarks.py --model=resnet50 --batch_size=256 --device=gpu
--force_gpu_compatible=True --num_intra_threads=1 --num_inter_threads=48
--variable_update=horovod --batch_group_size=20 --num_batches=500
--nodistortions --num_gpus=1 --data_format=NCHW --use_fp16=True
--use_tf_layers=False --data_name=imagenet --use_datasets=True
--data_dir=/mnt/mount_0/dataset/imagenet
--datasets_parallel_interleave_cycle_length=10
--datasets_sloppy_parallel_interleave=False --num_mounts=2
--mount_prefix=/mnt/mount_%d --datasets_prefetch_buffer_size=2000 --
datasets_use_prefetch=True --datasets_num_private_threads=4
--horovod_device=gpu >
/tmp/20190814_161609_tensorflow_horovod_rdma_resnet50_gpu_16_256_b500_im
agenet_nodistort_fp16_r10_m2_nockpt.txt 2>&1
```

6. 不要になったワーカー配置を削除します。次のコマンド例は、手順 1 で作成したワーカー配置オブジェクトの削除を示しています。

   ワーカー導入オブジェクトを削除すると、関連付けられているワーカーポッドは Kubernetes によって自動的に削除されます。

```
$ kubectl get deployments
NAME                                        DESIRED   CURRENT   UP-TO-DATE
AVAILABLE   AGE
netapp-tensorflow-multi-imagenet-worker   1         1         1
1           43m
$ kubectl get pods
NAME                                                        READY
STATUS      RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj              0/1
Completed   0          17m
netapp-tensorflow-multi-imagenet-worker-654fc7f486-v6725   1/1
Running     0          43m
$ kubectl delete deployment netapp-tensorflow-multi-imagenet-worker
deployment.extensions "netapp-tensorflow-multi-imagenet-worker" deleted
$ kubectl get deployments
No resources found.
$ kubectl get pods
NAME                                              READY   STATUS
RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj   0/1     Completed   0
18m
```

7. * オプション： * マスタージョブアーティファクトをクリーンアップします。次のコマンド例は、手順 3 で作成したマスタージョブオブジェクトの削除を示しています。

   マスタージョブオブジェクトを削除すると、関連付けられているマスターポッドは Kubernetes によって自動的に削除されます。

```
$ kubectl get jobs
NAME                                        COMPLETIONS   DURATION   AGE
netapp-tensorflow-multi-imagenet-master   1/1           5m50s      19m
$ kubectl get pods
NAME                                              READY   STATUS
RESTARTS   AGE
netapp-tensorflow-multi-imagenet-master-ppwwj   0/1     Completed   0
19m
$ kubectl delete job netapp-tensorflow-multi-imagenet-master
job.batch "netapp-tensorflow-multi-imagenet-master" deleted
$ kubectl get jobs
No resources found.
$ kubectl get pods
No resources found.
```

パフォーマンステスト

この解決策の作成の一環として、パフォーマンスを簡単に比較しました。Kubernetes を使用して、 NetApp AI の標準ベンチマークジョブをいくつか実行しました。また、ベンチマークの結果を、単純な Docker run コマンドを使用して実行した実行結果と比較しました。パフォーマンスに顕著な違いは見られませんでした。このため、 Kubernetes を使用してコンテナ化された AI トレーニングジョブをオーケストレーションしても、パフォーマンスに悪影響が及ばないことがわかりました。パフォーマンス比較の結果については、次の表を参照してください。

ベンチマーク	データセット	**Docker Run** （イメージ / 秒）	**Kubernetes** （画像 / 秒）
シングルノード TensorFlow	統合データ	6,667.2475	6,661.93125
シングルノード TensorFlow	ImageNet	6,570.2025	6,530.59125
2 ノードの同期分散 TensorFlow	統合データ	13,213.70625	13,218.288125
2 ノードの同期分散 TensorFlow	ImageNet	12,941.69125	12,881.33875

まとめ

あらゆる規模の企業や組織が、あらゆる業界で、人工知能（ AI ）、機械学習（ ML ）、ディープラーニング（ DL ）を活用して、現実世界の問題を解決し、革新的な製品やサービスを提供し、競争が激化する市場で優位に立つことが求められています。AI 、 ML 、 DL の利用が増えるにつれ、ワークロードの拡張性やデータの可用性など、多くの課題に直面しています。これらの課題には、 NetApp AI コントロールプレーン解決策を使用して対処できます。

この解決策を使用すると、データネームスペースのクローンを迅速に作成できます。さらに、トレーサビリティとバージョン管理のためにデータやモデルベースラインをほぼ瞬時に作成する AI 、 ML 、 DL のトレーニングワークフローを定義して実装できます。この解決策を使用すると、すべてのモデルトレーニングを、モデルがトレーニングされ、検証されたデータセットに戻すことができます。最後に、この解決策を使用すると、Jupyter Notebook ワークスペースをすばやくプロビジョニングし、大規模なデータセットにアクセスできます。

この解決策はデータサイエンティストとデータエンジニアを対象としているため、ネットアップまたは ONTAP に関する専門知識は最小限で済みます。この解決策を使用すると、シンプルで使い慣れたツールやインターフェイスを使用してデータ管理機能を実行できます。さらに、この解決策では、完全なオープンソースおよび無償のコンポーネントを使用しています。したがって、ネットアップストレージがすでにある環境では、この解決策を実装できます。この解決策でドライブをテストしたいが、ネットアップストレージはまだお持ちでない場合は、を参照してください "cloud.netapp.com"また、クラウドベースの NetApp Storage 解決策を使用すれば、いつでも稼働を開始できます。

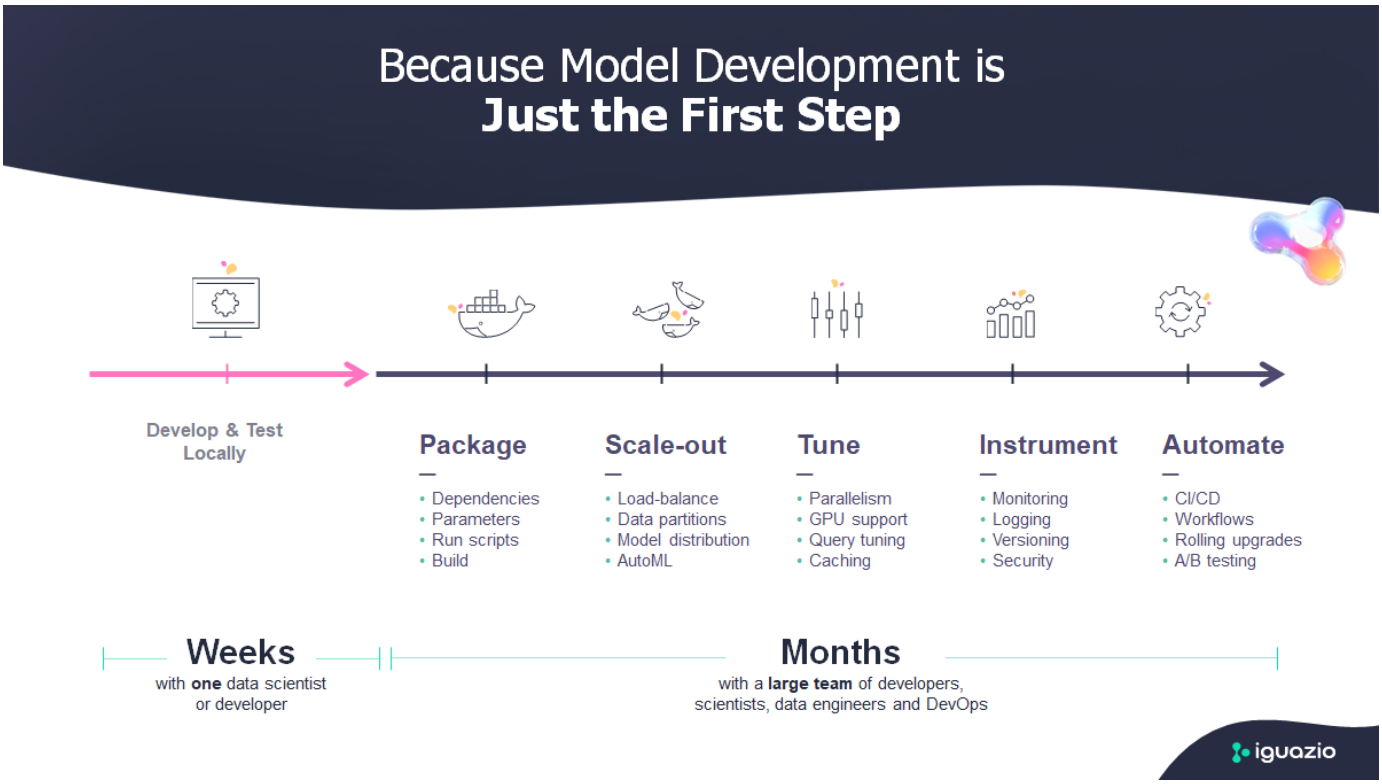
Iguazio による MLRun パイプライン

TR-4834 ：『 NetApp and Iguazio for MLRun Pipeline 』

Rick Huang 氏、 David Arnette 氏、 NetApp Marcelo Litovsky 氏、 Iguazio 氏

本ドキュメントでは、 NetApp ONTAP AI 、 NetApp AI コントロールプレーン、 NetApp Cloud Volume ソフトウェア、 Iguazio データサイエンスプラットフォームを使用した MLRun パイプラインの詳細について説明します。サーバレス機能の Nuclio 、 Kubernetes Persistent Volume 、 NetApp Cloud Volume 、 NetApp Snapshot コピー、 Grafana ダッシュボード、 および Iguazio プラットフォーム上のその他のサービスにより、ネットワーク障害検出のシミュレーション用のエンドツーエンドのデータパイプラインを構築できます。イグアスとネットアップのテクノロジを統合し、オンプレミスだけでなくクラウドでも、迅速なモデル導入、データレプリケーション、本番環境の監視を実現しました。

データサイエンティストの仕事は、機械学習（ ML ）モデルと人工知能（ AI ）モデルのトレーニングと調整に集中する必要があります。しかし、 Google の調査によると、データサイエンティストは、 AI / ML ワークフローでのモデル開発を示す次の図に示すように、モデルをエンタープライズアプリケーションで使用し、大規模に実行する方法を検討する時間の約 80% を費やしています。



エンドツーエンドの AI / ML プロジェクトを管理するには、エンタープライズコンポーネントについてより広範な理解が必要です。DevOps ではこのような種類のコンポーネントの定義、統合、導入が引き継がれていますが、機械学習の運用では、 AI / ML プロジェクトを含む同様のフローがターゲットとなります。エンドツーエンドの AI / ML パイプラインが企業内でどのように影響するかを知るには、次の必要なコンポーネントのリストを参照してください。

- ストレージ

- ネットワーキング
- データベース
- ファイルシステム
- コンテナ
- 継続的統合 / 継続的導入（ CI / CD ）パイプライン
- 開発統合開発環境（ IDE ）
- セキュリティ
- データアクセスポリシー
- ハードウェア
- クラウド
- 仮想化
- データサイエンスのツールセットとライブラリ

本書では、ネットアップと Iguazio のパートナーシップによってエンドツーエンドの AI / ML パイプラインの開発が大幅に簡易化されたことを紹介します。この簡易化により、 AI / ML アプリケーションの市場投入までの時間が短縮されます。

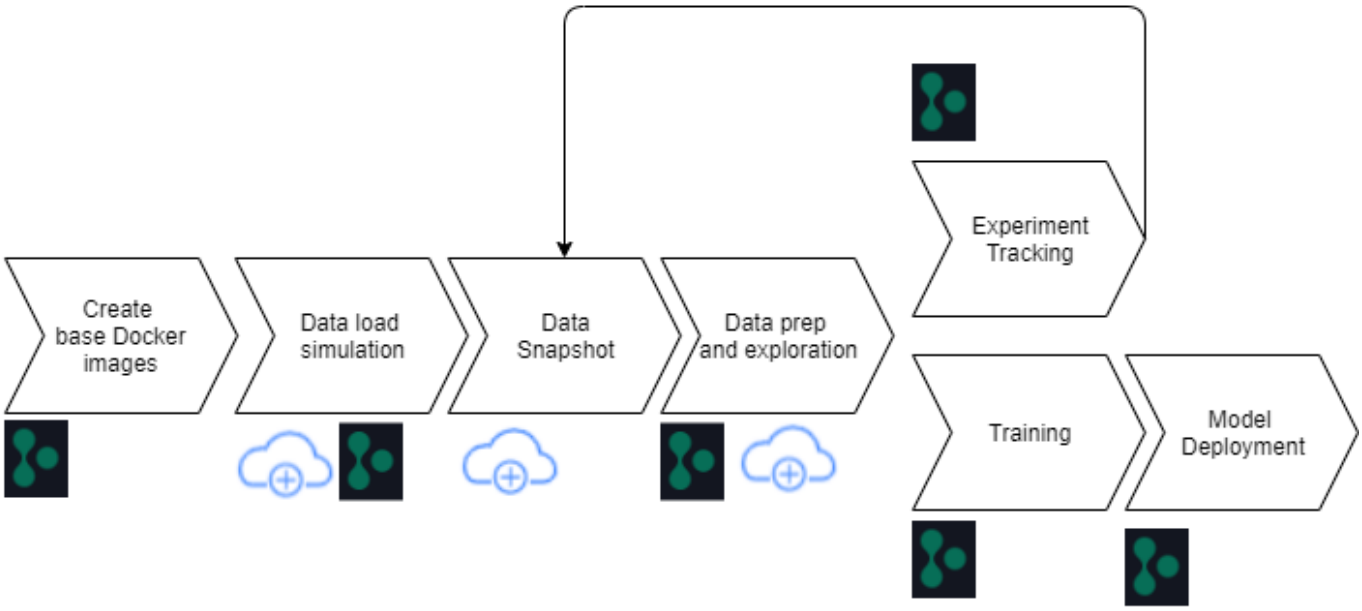
対象読者

データサイエンスの世界は、情報技術とビジネスのさまざまな分野に影響をもたらしています。

- データサイエンティストは、選択したツールとライブラリを柔軟に使用できる必要があります。
- データエンジニアは、データの流れと配置場所を把握する必要があります。
- DevOps エンジニアは、新しい AI / ML アプリケーションを CI / CD パイプラインに統合するためのツールを必要としています。
- ビジネスユーザは、 AI / ML アプリケーションにアクセスしたいと考えています。ネットアップと Iguazio がどのようにしてプラットフォームのビジネスに価値をもたらすかを説明します。

解決策の概要

この解決策は、 AI / ML アプリケーションのライフサイクルに従います。まず、データサイエンティストの仕事から始めて、データの前処理やモデルのトレーニングと導入に必要なさまざまな手順を定義します。成果物の追跡、実行の実験、 Kubeflow への導入が可能な、完全なパイプラインの構築に必要な作業をフォローしています。このサイクルを完了するには、パイプラインと NetApp Cloud Volume を統合し、次の図に示すようにデータのバージョン管理を可能にします。



iguazio

Cloud Volumes ONTAP

テクノロジの概要

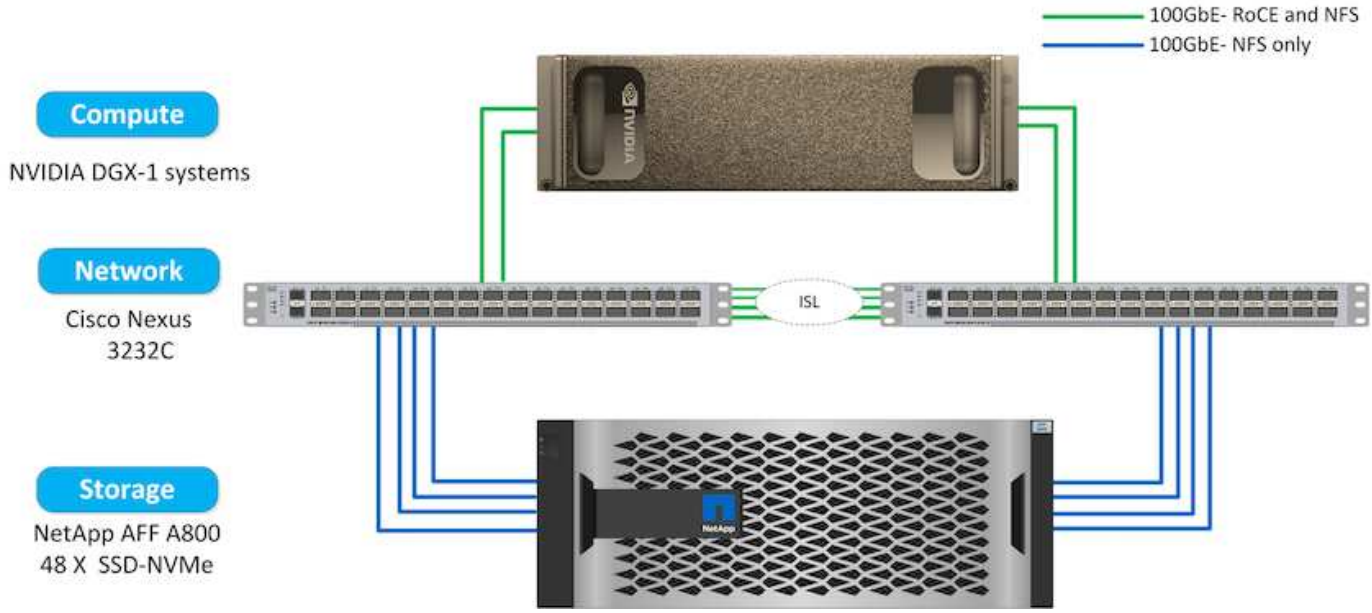
ネットアップの概要

ネットアップは、ハイブリッドクラウド環境におけるデータ管理のオーソリティです。ネットアップは、クラウド環境とオンプレミス環境全体でアプリケーションとデータの管理を簡易化し、デジタル変革を加速する、幅広いハイブリッドクラウドデータサービスを提供しています。グローバル企業がデータのポテンシャルを最大限に引き出し、お客様とのコンタクトの強化、イノベーションの促進、業務の最適化を図れるよう、パートナー様とともに取り組んでいます。

NetApp ONTAP AI

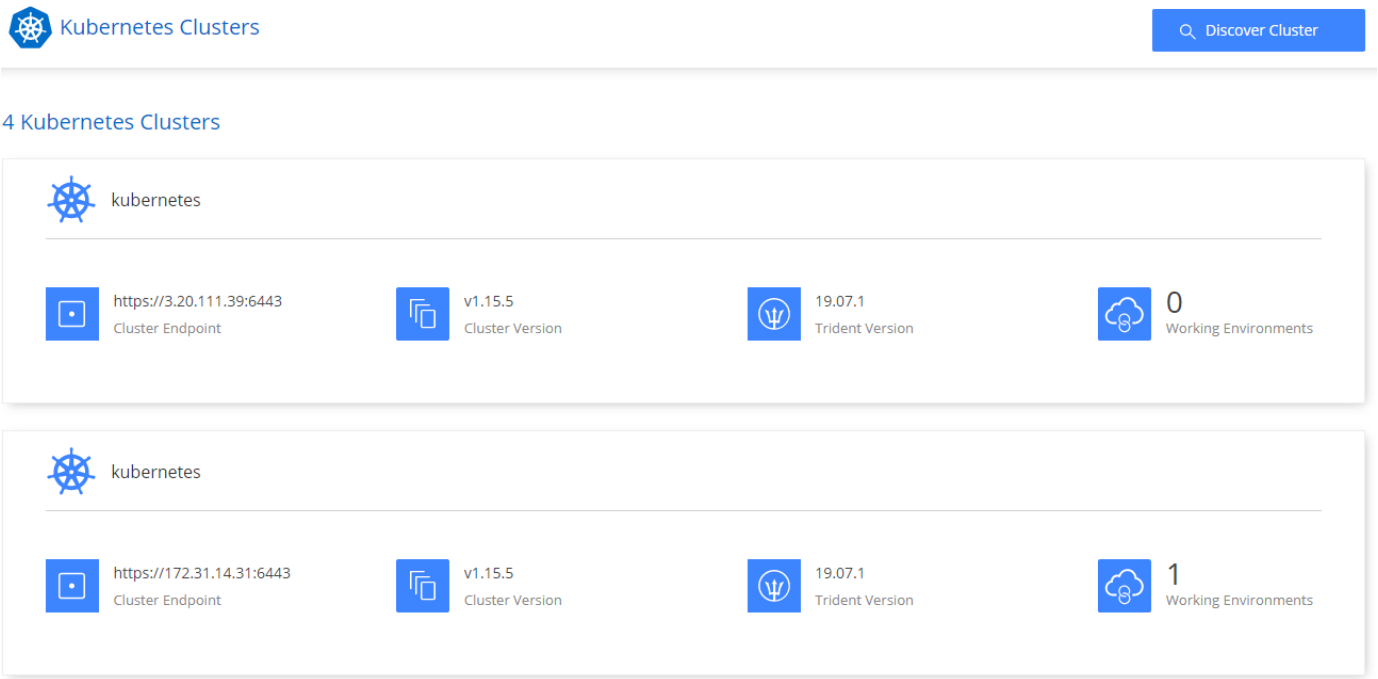
NVIDIA DGX システムとネットアップのクラウド対応オールフラッシュストレージを基盤とする NetApp ONTAP AI は、データの信頼性を高め、エッジからコア、クラウドにわたるデータファブリックで分析、トレーニング、推論を高速化します。IT 組織には、次のようなメリットをもたらすアーキテクチャが提供されます。

- 設計の複雑さを解消
- コンピューティングとストレージを個別に拡張できます
- 小規模構成から始めて、シームレスに拡張できます
- さまざまなパフォーマンスとコストの観点から、幅広いストレージオプションを提供 NetApp ONTAP AI は、 NVIDIA DGX-1 、ペタフロップス規模の AI システム、 NVIDIA Mellanox ハイパフォーマンスイーサネットスイッチを統合したコンバージドインフラスタックを提供し、 AI ワークロードの統合、導入の簡易化、 ROI の向上を実現します。このテクニカルレポートでは、 ONTAP AI を DGX-1 と NetApp AFF A800 ストレージシステムの 1 つに活用しました。次の図は、この検証で使用した DGX-1 システムを使用した ONTAP AI のトポロジを示しています。



NetApp AI コントロールプレーン

ネットアップの AI コントロールプレーンは、卓越した拡張性、合理的な導入、ノンストップのデータ可用性を備えた解決策で、 AI と ML を最大限に活用できます。AI コントロールプレーン解決策は、 Kubernetes と Kubeflow をネットアップのデータファブリックと統合します。クラウドネイティブ環境向けの業界標準のコンテナオーケストレーションプラットフォームである Kubernetes は、ワークロードの拡張性とモビリティを実現します。Kubeflow はオープンソースの機械学習プラットフォームで、管理と導入を簡易化し、開発者がより多くのデータサイエンスをより短時間で行えるようにします。ネットアップのデータファブリックは、エッジからコア、クラウドまで、パイプライン全体でデータに確実にアクセスできるよう、データの可用性とモビリティを妥協することなく提供します。このテクニカルレポートでは、 MLRun パイプラインで NetApp AI コントロールプレーンを使用しています。次の図は、 Kubernetes クラスタ管理ページを示しています。各クラスタに異なるエンドポイントを割り当てることができます。NFS Persistent Volume を Kubernetes クラスタに接続し、次の図に示すように、クラスタに接続された永続的ボリュームが表示されます "NetApp Trident" 永続的ストレージのサポートとデータ管理機能を提供します。



Persistent Volumes for Kubernetes

Connected with Kubernetes Cluster

Cloud Volumes ONTAP is connected to 1 Kubernetes cluster. View Cluster

You can connect another Kubernetes cluster to this Cloud Volumes ONTAP system. If the Kubernetes cluster is in a different network than Cloud Volumes ONTAP, specify a custom export policy to provide access to clients.

Kubernetes Cluster

Select Kubernetes Cluster

kubernetes

Custom Export Policy *(Optional)*

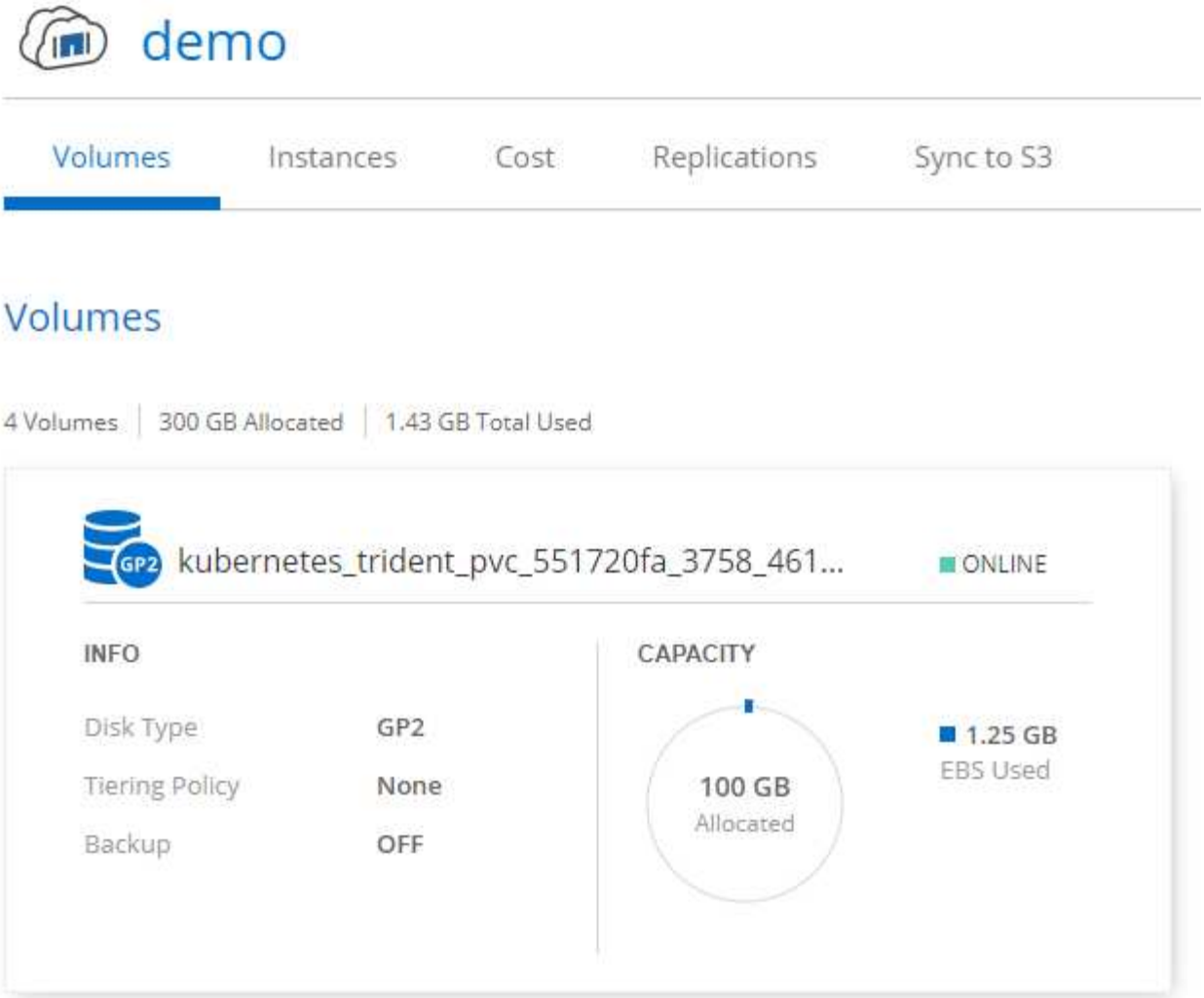
Custom Export Policy

172.31.0.0/16

☑ Set as default storage class

◉ NFS ○ iSCSI

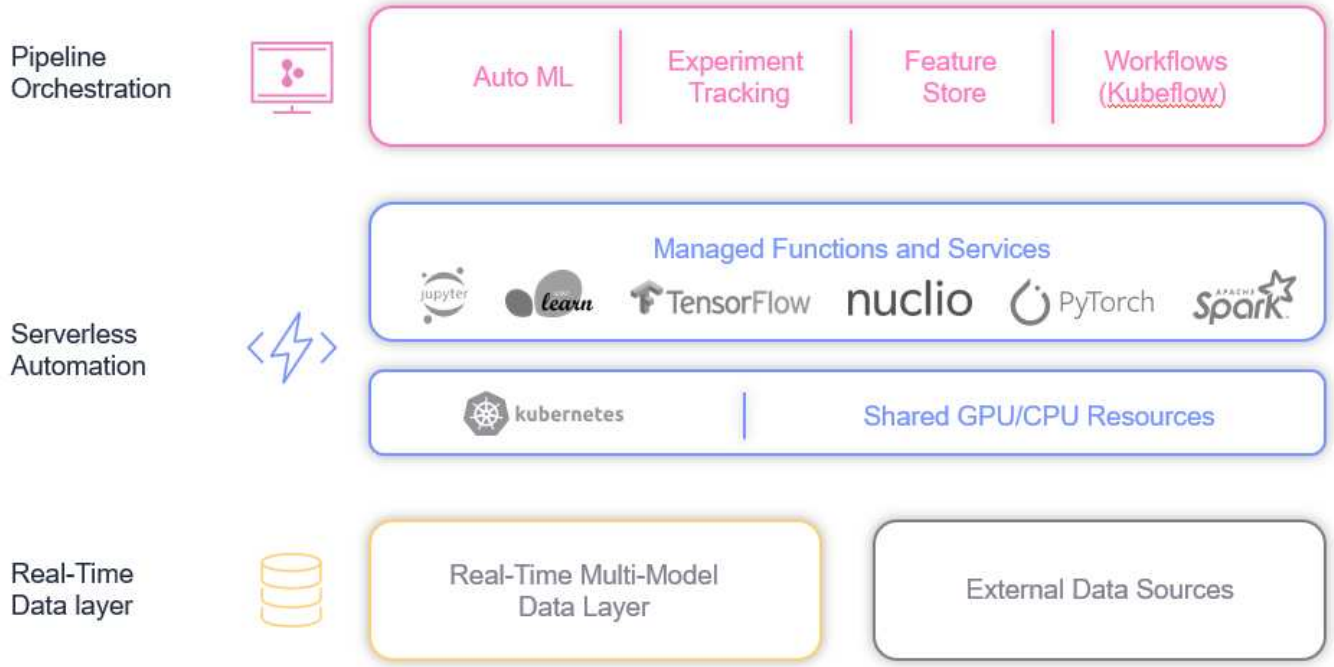
Connect | Cancel



Iguazio の概要

Iguazio Data Science Platform は、開発の簡素化、パフォーマンスの向上、コラボレーションの促進、運用上の課題への対処を可能にする、完全に統合された安全なデータサイエンスプラットフォームサービス（ PaaS ）です。このプラットフォームには以下のコンポーネントが組み込まれており、 Iguazio データサイエンスプラットフォームを次の図に示します。

- Jupyter Notebook 、統合分析エンジン、 Python パッケージを含むデータサイエンスワークベンチ
- 実験追跡機能と自動化されたパイプライン機能を使用したモデル管理
- 拡張性に優れた Kubernetes クラスタでデータサービスと ML サービスを管理
- サーバレス関数のリアルタイムフレームワークである Nuclio
- SQL 、 NoSQL 、時系列データベース、ファイル（シンプルなオブジェクト）、ストリーミングをサポートする、きわめて高速でセキュアなデータレイヤです
- ネットアップ、 Amazon S3 、 HDFS 、 SQL データベース、ストリーミングプロトコルやメッセージングプロトコルなどのサードパーティ製データソースとの統合
- Grafana に基づくリアルタイムダッシュボード



ソフトウェアとハードウェアの要件

ネットワーク構成：

クラウドにセットアップするためのネットワーク構成の要件は次のとおりです。

- Iguazio クラスタと NetApp Cloud Volume は、同じ仮想プライベートクラウドに存在する必要があります。
- クラウドマネージャは、 Iguazio アプリノードのポート 443 にアクセスできる必要があります。
- 本テクニカルレポートでは Amazon Web Services を使用しました。ただし、任意のクラウドプロバイダに解決策を導入することもできます。 ONTAP AI で NVIDIA DGX-1 を使用したオンプレミステストの場合は、便宜のために Iguazio ホスト DNS サービスを使用しました。

クライアントは、動的に作成される DNS ドメインにアクセスできる必要があります。お客様は必要に応じて独自の DNS を使用できます。

ハードウェア要件

イグアスは、ご使用のクラスタにオンプレミスでインストールできます。ネットアップでは、 NVIDIA DGX-1 システムで NetApp ONTAP AI の解決策を検証しました。次の表に、この解決策のテストに使用したハードウェアを示します。

ハードウェア	数量
DGX-1 システム	1.
NetApp AFF A800 システム	ハイアベイラビリティ（ HA ）ペア × 1 、コントローラ × 2 、 NVMe SSD × 48 （ 3.8TB 以上）
Cisco Nexus 3232C ネットワークスイッチ	2.

次の表に、オンプレミステストに必要なソフトウェアコンポーネントを示します。

ソフトウェア	バージョンまたはその他の情報
NetApp ONTAP データ管理ソフトウェア	9.7
Cisco NX-OS スイッチのファームウェア	7.0 （3） I6 （1）
NVIDIA DGX OS	4.4 - Ubuntu 18.04 LTS
Docker コンテナプラットフォーム	19.03.5
コンテナバージョン	20.01-tF1 - py2
機械学習フレームワーク	TensorFlow 1.15.0
イグアテオ	バージョン 2.8 以降
ESX サーバ	6.5

この解決策は、 Iguazio バージョン 2.5 および NetApp Cloud Volumes ONTAP for AWS で完全にテストされています。Iguazio クラスタとネットアップのソフトウェアは、どちらも AWS で実行されています。

ソフトウェア	**[** バージョン **]** または **[** タイプ **]**
イグアテオ	バージョン 2.8 以降
アプリケーションノード	m5.mc
データノード	I3.と は、および

ネットワークデバイスの障害予測ユースケースの概要

この使用例は、アジアの通信空間における Iguazio の顧客をベースにしています。100 万社の企業顧客と年間 125 万件のネットワーク停止イベントに対応しているため、ネットワーク障害による顧客への影響を防止するために、事前に対処する必要がありました。この解決策には、次のような利点があります。

- ネットワーク障害の予測分析
- チケット発行システムとの統合
- このような Iguazio の実装により、ネットワーク障害を予防するための予防的措置を講じ、障害の 60% を予防しました。

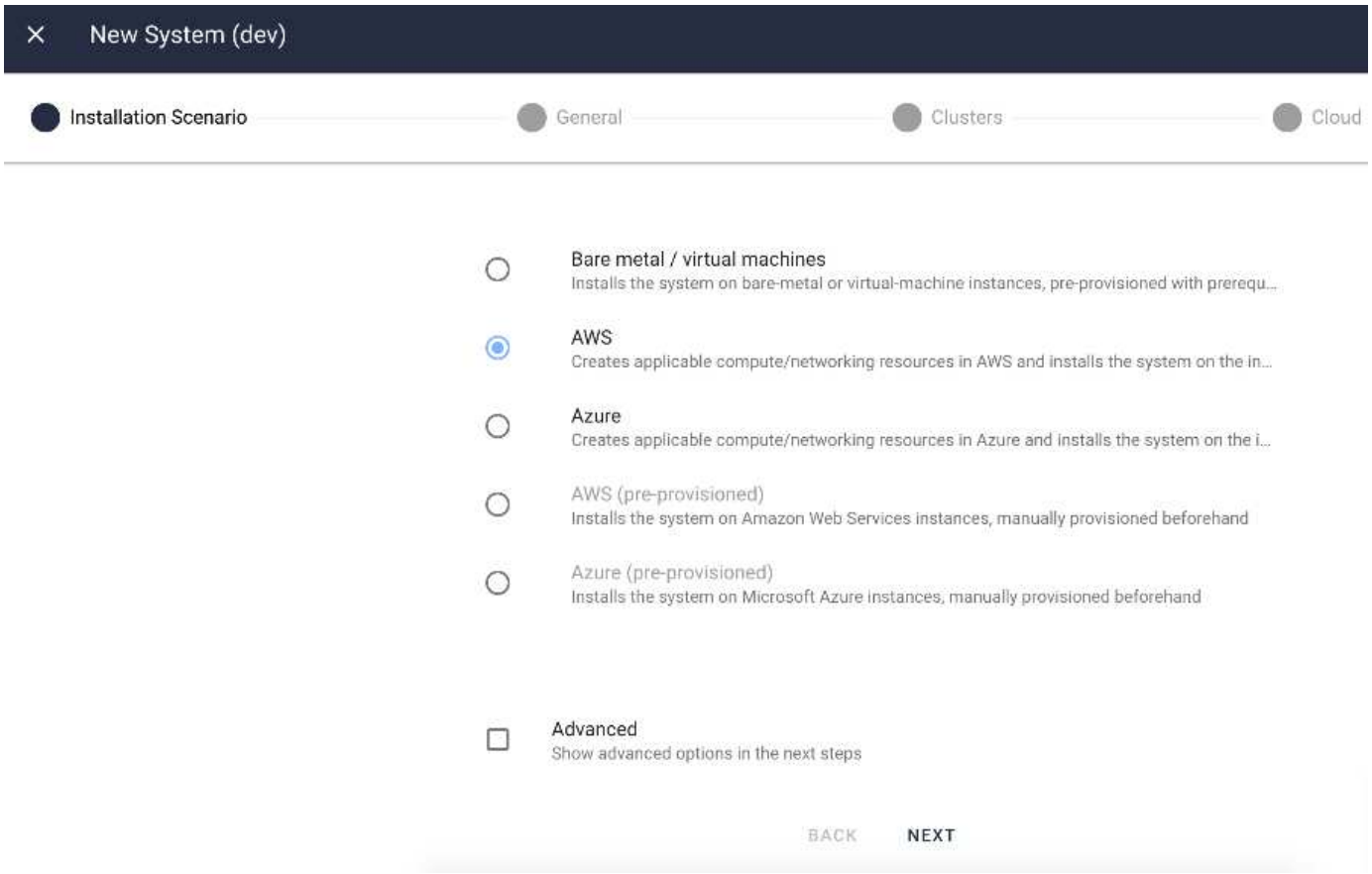
セットアップの概要

Iguazio は、オンプレミスまたはクラウドプロバイダにインストールできます。

Iguazio の取り付け

プロビジョニングはサービスとして実行でき、 Iguazio またはお客様による管理が可能です。どちらの場合も、 Iguazio はクラスタの導入と管理に使用する導入アプリケーション（ Provazio ）を提供します。

オンプレミスでのインストールについては、を参照してください "NVA-1121." コンピューティング、ネットワーク、ストレージのセットアップに使用できます。イグアスのオンプレミス導入は、顧客に追加料金なしで提供されます。を参照してください "このページです" DNS サーバおよび SMTP サーバの設定Provazio のインストールページは次のとおりです。



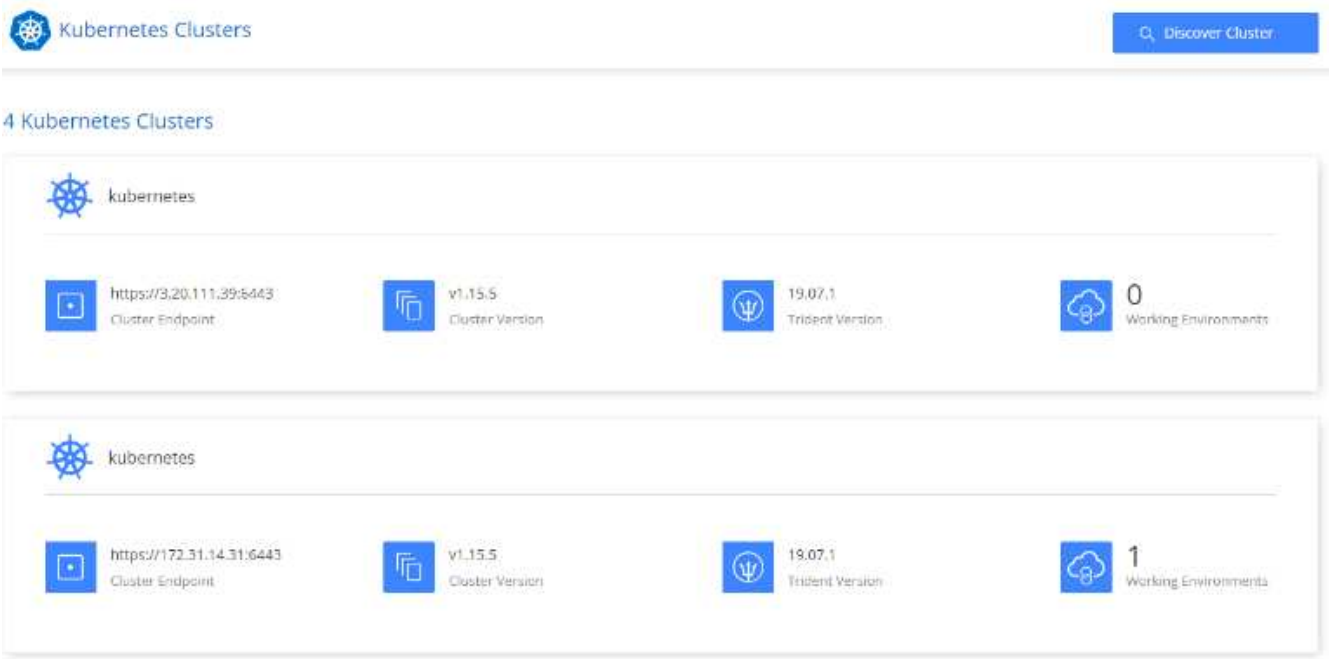
Kubernetes クラスタを設定しています

このセクションは、クラウドとオンプレミスのそれぞれの導入について、 2 つの部分に分かれています。

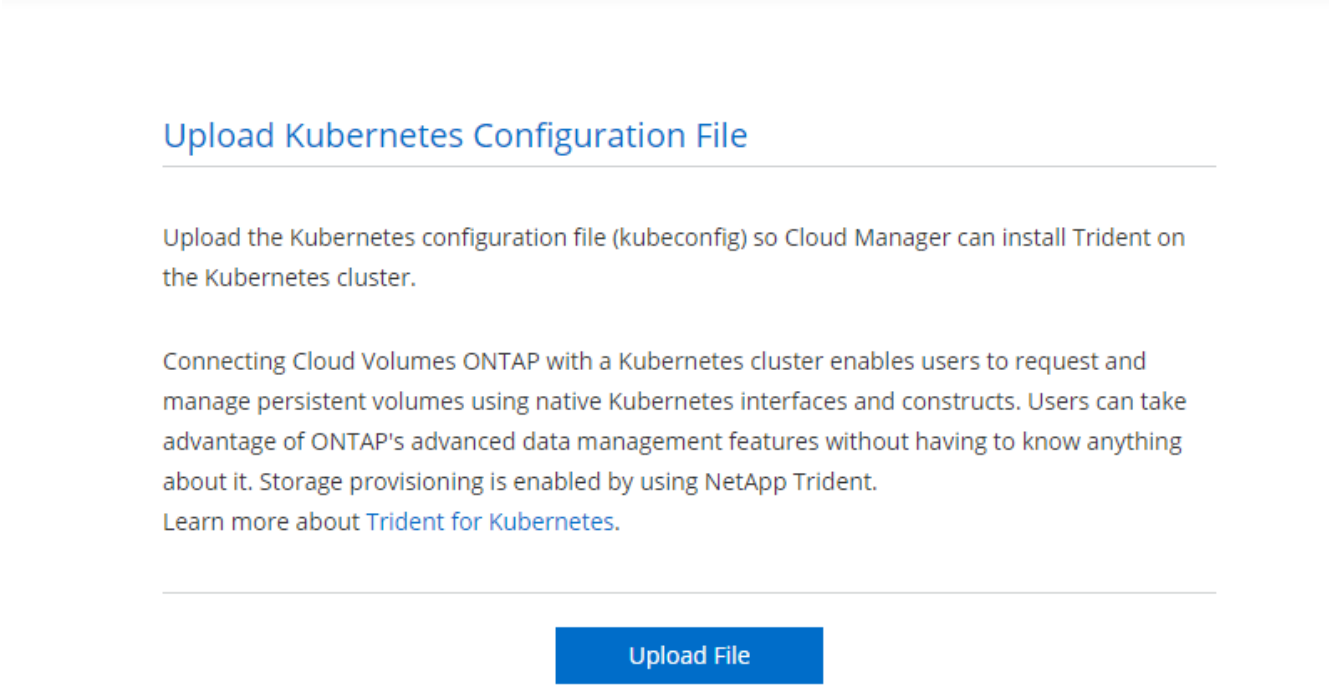
Cloud Deployment Kubernetes Configuration を参照してください

NetApp Cloud Manager を使用して、イグアス Kubernetes クラスタへの接続を定義できます。Trident では、ボリュームを使用できるようにするために、クラスタ内の複数のリソースにアクセスする必要があります。

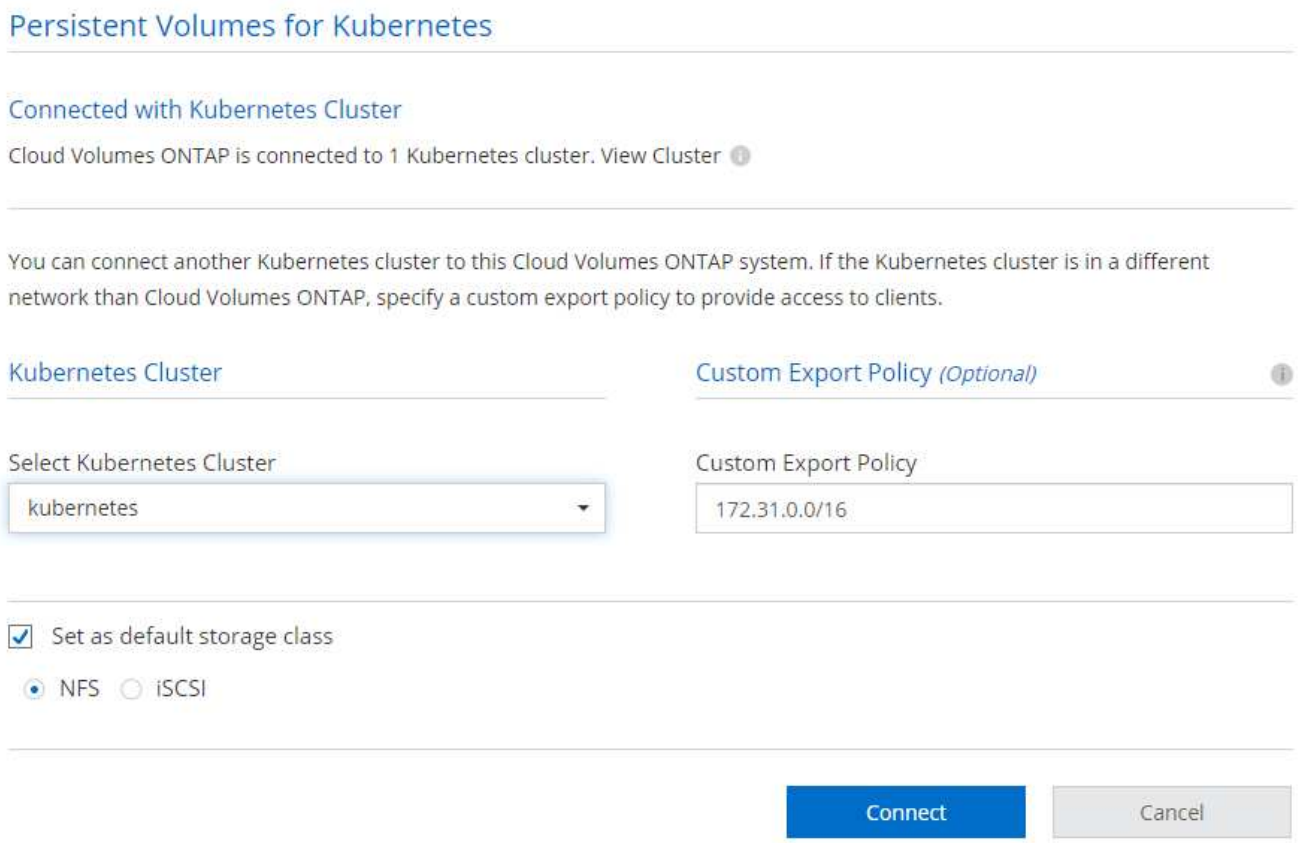
1. アクセスを有効にするには、 1 つの Iguazio ノードから Kubernetes 構成ファイルを取得します。ファイルは、「 /home/Iguazio/.kube/config. 」の下にあります このファイルをデスクトップにダウンロードします。
2. 設定するクラスタの検出に進みます。



3. Kubernetes 構成ファイルをアップロードします。次の図を参照してください。



4. Trident を導入し、ボリュームをクラスタに関連付けます。Iguazio クラスタへの持続ボリュームの定義と割り当てについては、次の図を参照してください。このプロセスにより、 Iguazio の Kubernetes クラスタに永続ボリューム（ PV ）が作成されます。使用する前に、永続的ボリューム要求（ PVC ）を定義する必要があります。



オンプレミス導入の **Kubernetes** 構成

NetApp Trident のオンプレミスインストールについては、を参照してください "TR-4798" を参照してください。Kubernetes クラスタを設定して NetApp Trident をインストールしたら、 Trident を Iguazio クラスタに接続して、データやモデルの Snapshot コピーの作成などのネットアップのデータ管理機能を利用できます。

永続的ボリューム要求を定義

1. 次の YAML をファイルに保存して、 Basic タイプの PVC を作成します。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: basic
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
  storageClassName: netapp-file
```

2. Iguazio Kubernetes クラスタに YAML ファイルを適用します。

```
Kubectl -n default-tenant apply -f <your yaml file>
```

NetApp ボリュームを **Jupyter Notebook** に接続します

Iguazio は、データサイエンティストが AI / ML アプリケーションの開発と導入のための完全なエンドツーエンドスタックを提供するための、複数のマネージドサービスを提供します。これらのコンポーネントの詳細については、を参照してください "Iguazio アプリケーションサービスおよびツールの概要"。

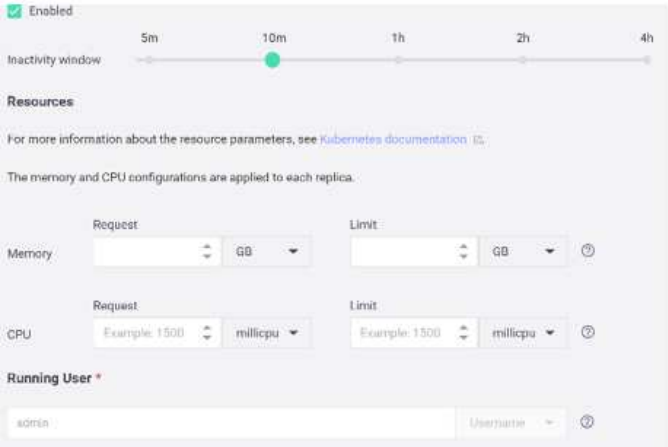
マネージドサービスの 1 つに Jupyter Notebook があります。開発者はそれぞれ、開発に必要なリソースを備えたノートブックコンテナを独自に導入します。NetApp Cloud Volume へのアクセスを許可するには、コンテナにボリュームを割り当て、リソースの割り当て、ユーザの実行、および永続ボリュームに関する環境変数の設定を次の図に示します。

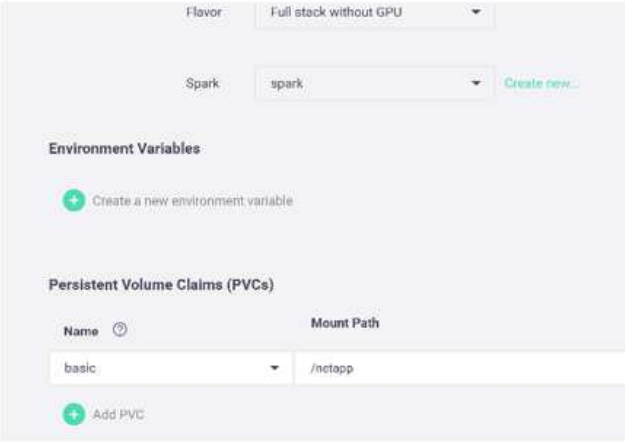
オンプレミス構成の場合は、を参照してください "TR-4798" Trident のセットアップでは、データの Snapshot コピーの作成やバージョン管理のためのモデルなど、 NetApp ONTAP のデータ管理機能を有効にできます。Trident バックエンド構成ファイルに次の行を追加すると、 Snapshot ディレクトリが表示されます。

```
{
    …
    "defaults": {
        "snapshotDir": "true"
    }
}
```

Trident バックエンド構成ファイルを JSON 形式で作成し、次のコマンドを実行する必要があります "Trident コマンド" 参照するには：

```
tridentctl create backend -f <backend-file>
```





アプリケーションの展開

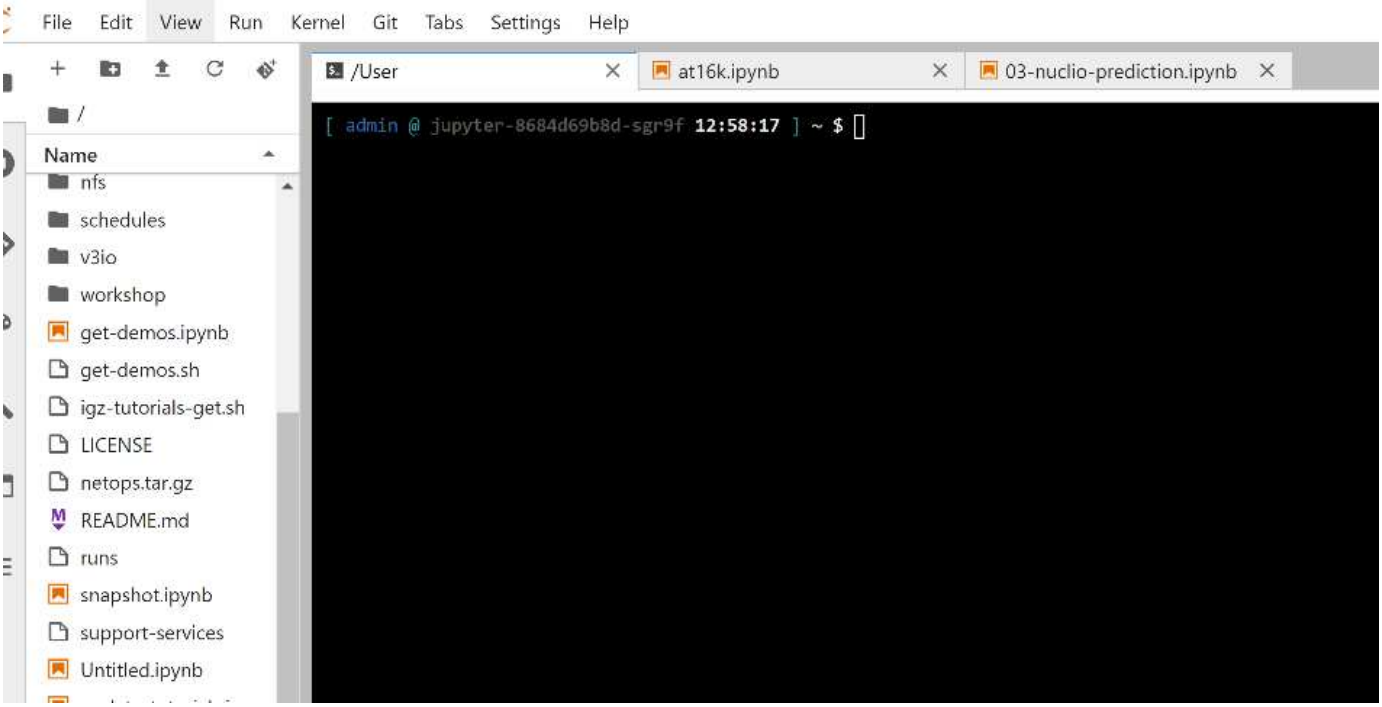
次のセクションでは、アプリケーションのインストールと展開の方法について説明します。

GitHub からコードを取得します

これで、 Iguazio クラスタおよび開発者環境で NetApp Cloud Volume または NetApp Trident ボリュームを使用できるようになりました。アプリケーションの確認を開始できます。

ユーザは独自のワークスペース（ディレクトリ）を持ちます。すべてのノートブックで ' ユーザー・ディレクトリへのパスは '/User' ですIguazio プラットフォームは、ディレクトリを管理します。上記の手順に従った場合、 NetApp Cloud ボリュームは「 /NetApp 」ディレクトリにあります。

Jupyter 端子を使用して GitHub からコードを取得します。



Jupyter ターミナルのプロンプトで、プロジェクトのクローンを作成します。

```
cd /User
git clone .
```

Jupyter ワークスペースのファイルツリーには、「 NetOps - NetApp 」フォルダが表示されます。

作業環境を構成します

「 Notebook ```s_env-example.ipynb を 'et_env.ipynb` としてコピーします。「 et_env.ipynb 」を開き、編集します。このノートブックでは、資格情報、ファイルの場所、および実行ドライバの変数を設定します。

上記の手順を実行すると、次の手順だけが変更されます。

1. この値は、 Iguazio サービスダッシュボード「 dOcker_registry 」から取得します

例：「 ocker-registry.default-tenant.app.clusterq.iguaziodev.com:80` 」

2. 「 admin 」を Iguazio のユーザ名に変更します。

   'IGZ_container_path='/users/admin'

   ONTAP システムの接続の詳細を次に示します。Trident のインストール時に生成されたボリューム名も指定します。オンプレミスの ONTAP クラスタの場合、次の設定が適用されます。

```
ontapClusterMgmtHostname = '0.0.0.0'
ontapClusterAdminUsername = 'USER'
ontapClusterAdminPassword = 'PASSWORD'
sourceVolumeName = 'SOURCE VOLUME'
```

   Cloud Volumes ONTAP の設定は次のとおりです。

```
MANAGER=ontapClusterMgmtHostname
svm='svm'
email='email'
password=ontapClusterAdminPassword
weid="weid"
volume=sourceVolumeName
```

ベースとなる **Docker** イメージを作成

ML パイプラインの構築に必要なものはすべて、 Iguazio プラットフォームに含まれています。開発者は、パイプラインの実行に必要な Docker イメージの仕様を定義し、 Jupyter Notebook からイメージの作成を実行できます。ノートブック 'create-images .ipynb' を開き、すべてのセルを実行します。

このノートブックでは、パイプラインで使用する 2 つのイメージが作成されます。

- 「 iguazio/NetApp. 」を参照してください ML タスクの処理に使用されます。

Create image for training pipeline

```
[4]: fn.build_config(image=docker_registry+'/iguazio/netapp', commands=['pip install \
     v3io_frames fsspec>=0.3.3 PyYAML==5.1.2 pyarrow==0.15.1 pandas==0.25.3 matplotlib seaborn yellowb
     fn.deploy()
```

- 「 NetApp/pipeline. 」。NetApp Snapshot コピーを処理するユーティリティが含まれています。

Create image for Ontap utilitites

```
[9]: fn.build_config(image=docker_registry + '/netapp/pipeline:latest',commands=['apt -y update','pip install v3io_frames netapp_ontap'
     fn.deploy()
```

Jupyter ノートブックを個別に確認します

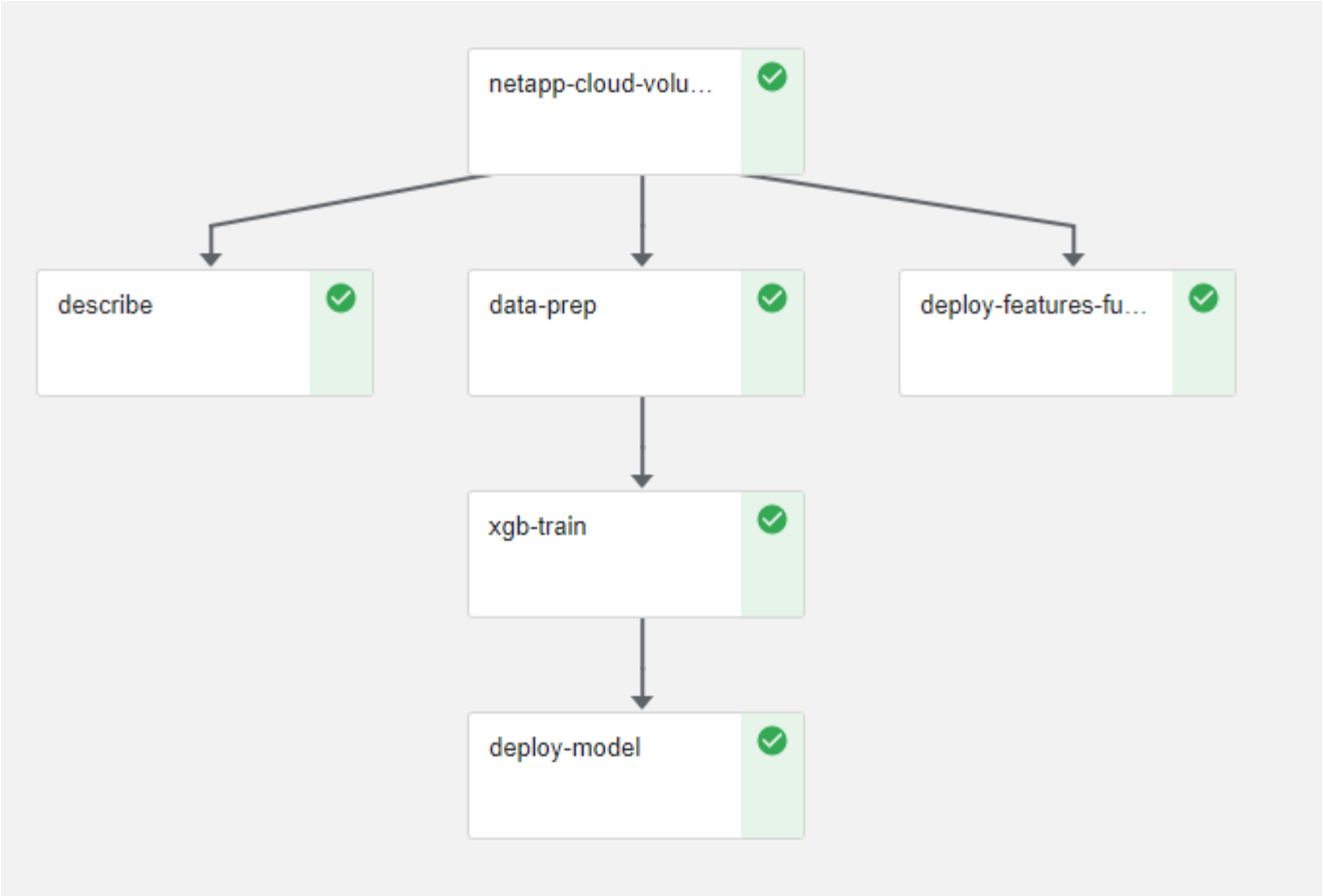
次の表に、このタスクの構築に使用したライブラリとフレームワークを示します。これらのコンポーネントは

すべて、 Iguazio のロールベースアクセスおよびセキュリティ制御と完全に統合されています。

ライブラリ / フレームワーク	説明
MLRun （ MLRun ）	Iguazio によって管理され、 ML / AI パイプラインのアセンブリ、実行、および監視を可能にします。
Nuclio	Iguazio と統合されたサーバーレス機能フレームワーク。Iguazio が管理するオープンソースプロジェクトとしても利用できます。
クビフロー	パイプラインを導入するための Kubernetes ベースのフレームワーク。これは、イグアスが寄与するオープンソースのプロジェクトでもあります。Iguazio と統合されているため、他のインフラストラクチャとのセキュリティおよび統合が強化されています。
Docker です	Iguazio プラットフォームでは、 Docker レジストリがサービスとして実行されます。レジストリに接続するように変更することもできます。
NetApp Cloud Volume の略	AWS で Cloud Volume を実行すると、大量のデータにアクセスでき、トレーニングに使用するデータセットのバージョンに Snapshot コピーを作成することもできます。
Trident	Trident は、ネットアップが管理するオープンソースプロジェクトです。Kubernetes でのストレージリソースやコンピューティングリソースとの統合を簡易化します。

複数のノートブックを使用して ML パイプラインを構築しました。各ノートブックを個別にテストしてから、パイプラインにまとめてテストすることができます。このデモアプリケーションの導入フローに従って、各ノートブックについて個別に説明します。

望ましい結果は、データの Snapshot コピーに基づいてモデルをトレーニングし、推論のためにモデルを導入するパイプラインです。完成した MLRun パイプラインのブロック図を次の図に示します。



データ生成機能を導入します

このセクションでは、ネットワークデバイスデータの生成に Nuclio サーバーレス関数を使用する方法について説明します。この使用例は、パイプラインを展開し、イグアスのサービスを使用してネットワークデバイスの障害を監視および予測する Iguazio クライアントに適しています。

ネットワークデバイスからのデータをシミュレートしました。Jupyter ノートブック「 d ata-generator.ipynb 」を実行すると、 10 分ごとに実行されるサーバーレス関数が作成され、新しいデータが保存された寄木細工のファイルが生成されます。この機能を配備するには、このノートブックのすべてのセルを実行します。を参照してください "Nuclio の Web サイト" このノートブックの構成部品を確認します。

関数の生成時に、次のコメントを持つセルは無視されます。ノートブック内のすべてのセルは、機能の一部であると見なされます。Nuclio モジュールをインポートして '%nuclio magic を有効にします

```
# nuclio: ignore
import nuclio
```

関数の仕様では、関数が実行される環境、関数がどのようにトリガされるか、および関数が消費するリソースを定義しました。

```
spec = nuclio.ConfigSpec(config={"spec.triggers.inference.kind":"cron",

"spec.triggers.inference.attributes.interval" :"10m",
                                "spec.readinessTimeoutSeconds" : 60,
                                "spec.minReplicas" : 1},……
```

「 init_context 」関数は、関数の初期化時に Nuclio フレームワークによって呼び出されます。

```
def init_context(context):
    ….
```

関数内にないコードは、関数が初期化されるときに呼び出されます。この関数を呼び出すと、ハンドラ関数が実行されます。ハンドラの名前を変更し、関数仕様で指定できます。

```
def handler(context, event):
            …
```

この機能は、導入前にノートブックからテストできます。

```
%%time
# nuclio: ignore
init_context(context)
event = nuclio.Event(body='')
output = handler(context, event)
output
```

この機能は、ノートブックから導入することも、 CI / CD パイプラインから導入することもできます（このコードを使用）。

```
addr = nuclio.deploy_file(name='generator',project='netops',spec=spec,
tag='v1.1')
```

ノートブックをパイプライン化します

これらのノートブックは、このセットアップで個別に実行することを意図したものではありません。これは、各ノートブックを確認するためのものです。ネットアップは、このような案件をパイプラインの一部として呼び出しました。個別に実行するには、 MLRun のドキュメントを参照して、これらを Kubernetes ジョブとして実行します。

snap_CV.ipynb

このノートブックでは、パイプラインの最初にあるクラウドボリュームの Snapshot コピーを処理します。ボ

リュームの名前をパイプラインコンテキストに渡します。このノートブックは、スナップショットコピーを処理するシェルスクリプトを呼び出します。パイプラインでの実行中、実行コンテキストには、実行に必要なすべてのファイルを見つけるのに役立つ変数が含まれています。このコードを記述する際、開発者は、このコードを実行するコンテナ内のファイルの場所を気にする必要はありません。後で説明したように、このアプリケーションはすべての依存関係とともに配置され、実行コンテキストを提供するパイプラインパラメータの定義です。

```
command = os.path.join(context.get_param('APP_DIR'),"snap_cv.sh")
```

作成された Snapshot コピーの場所は、 MLRun コンテキストに配置され、パイプラインの各ステップで使用されます。

```
context.log_result('snapVolumeDetails',snap_path)
```

次の 3 つのノートブックは並行して実行されます。

データの前処理 **ipynb**

モデルのトレーニングを有効にするには、生の指標を機能に変換する必要があります。このノートでは、Snapshot ディレクトリから生の指標を読み取り、モデルトレーニングの機能をネットアップボリュームに書き込みます。

パイプラインのコンテキストで実行する場合、「 Data ATA_DIR 」という入力には Snapshot コピーの場所が含まれます。

```
metrics_table = os.path.join(str(mlruncontext.get_input('DATA_DIR',
os.getenv('DATA_DIR','/netpp'))),
                             mlruncontext.get_param('metrics_table',
os.getenv('metrics_table','netops_metrics_parquet')))
```

.ipynb を説明する

受信メトリックを視覚化するために、 Kubeflow UI と MLRun UI で使用できるプロットとグラフを提供するパイプラインステップを導入します。各実行には、この表示ツールの独自のバージョンがあります。

```
ax.set_title("features correlation")
plt.savefig(os.path.join(base_path, "plots/corr.png"))
context.log_artifact(PlotArtifact("correlation",  body=plt.gcf()),
local_path="plots/corr.html")
```

deploy-feature-function.ipynb

ネットアップでは、異常を検出している指標を継続的に監視してこのノートブックは、受信メトリックの予測を実行するために必要な機能を生成するサーバーレス機能を作成します。このノートブックは関数の作成を呼び出します。ファンクションコードはノートブック「 ata-prep . ipynb 」にあります。この目的のために、パ

イプラインのステップとして同じノートブックを使用していることに注意してください。

train.ipynb

フィーチャーを作成した後、モデルトレーニングを開始します。このステップの出力は、推論に使用するモデルです。また、統計を収集して各実行を追跡します（実験）。

たとえば、次のコマンドは、その測定条件のコンテキストに精度スコアを入力します。この値は Kubeflow および MLRun で確認できます。

```
context.log_result(‘accuracy’,score)
```

deploy-inion-function.ipynb を展開します

パイプラインの最後のステップは、継続的な推論のためのサーバーレス機能としてモデルを導入することです。このノートブックでは、「 nuclio-increation-function.ipynb 」で定義されたサーバーレス関数の作成を呼び出します。

パイプラインのレビューと構築

パイプラインですべてのノートブックを実行するという組み合わせにより ' テストを継続的に実行して ' モデルの精度を新しいメトリックと比較して再評価することができますまず 'pipeline.ipynb' ノートブックを開きますネットアップと Iguazio が ML パイプラインの導入をどのように簡易化しているかを詳しく説明します。

MLRun を使用して、パイプラインの各ステップにコンテキストを提供し、リソースの割り当てを処理します。MLRun API サービスは、 Iguazio プラットフォームで動作し、 Kubernetes リソースとのやり取りのポイントです。各開発者はリソースを直接要求できません。 API は要求を処理し、アクセス制御を有効にします。

```
# MLRun API connection definition
mlconf.dbpath = 'http://mlrun-api:8080'
```

パイプラインは、 NetApp Cloud Volume やオンプレミスのボリュームと連携できます。このデモでは Cloud Volume を使用するように設計しましたが、オンプレミスで実行できるオプションをコードに示しています。

```
# Initialize the NetApp snap fucntion once for all functions in a notebook
if [ NETAPP_CLOUD_VOLUME ]:
    snapfn =
code_to_function('snap',project='NetApp',kind='job',filename="snap_cv.ipyn
b").apply(mount_v3io())
    snap_params = {
    "metrics_table" : metrics_table,
    "NETAPP_MOUNT_PATH" : NETAPP_MOUNT_PATH,
    'MANAGER' : MANAGER,
    'svm' : svm,
    'email': email,
    'password': password ,
    'weid': weid,
    'volume': volume,
    "APP_DIR" : APP_DIR
       }
else:
    snapfn =
code_to_function('snap',project='NetApp',kind='job',filename="snapshot.ipy
nb").apply(mount_v3io())
….
snapfn.spec.image = docker_registry + '/netapp/pipeline:latest'
snapfn.spec.volume_mounts =
[snapfn.spec.volume_mounts[0],netapp_volume_mounts]
      snapfn.spec.volumes = [ snapfn.spec.volumes[0],netapp_volumes]
```

Jupyter ノートブックを Kubeflow ステップにするために必要な最初のアクションは、コードを関数に変換することです。関数には、ノートブックを実行するために必要なすべての仕様が含まれています。ノートブックを下にスクロールすると、パイプラインのすべてのステップに対応する関数が定義されていることがわかります。

ノートブックの一部	説明
<code_to _function> （ MLRun モジュールの一部）	関数の名前：プロジェクト名。すべてのプロジェクトアーティファクトの編成に使用されます。これは MLRun UI に表示されます。種類：この場合は Kubernetes ジョブ。これには、 Dask 、 MPI 、 spark8s などがあります。詳細については、 MLRun のマニュアルを参照してください。ファイル。ノートブックの名前。これは Git （ HTTP ）の場所にすることもできます。
イメージ（ Image ）	この手順で使用する Docker イメージの名前。先ほど 'create-image.ipynb ノートブックを作成しました
volume_mounts と volumes	実行時に NetApp Cloud Volume をマウントするための詳細情報。

また、ステップのパラメーターも定義します。

```
params={   "FEATURES_TABLE":FEATURES_TABLE,
           "SAVE_TO" : SAVE_TO,
           "metrics_table" : metrics_table,
           'FROM_TSDB': 0,
           'PREDICTIONS_TABLE': PREDICTIONS_TABLE,
           'TRAIN_ON_LAST': '1d',
           'TRAIN_SIZE':0.7,
           'NUMBER_OF_SHARDS' : 4,
           'MODEL_FILENAME' : 'netops.v3.model.pickle',
           'APP_DIR' : APP_DIR,
           'FUNCTION_NAME' : 'netops-inference',
           'PROJECT_NAME' : 'netops',
           'NETAPP_SIM' : NETAPP_SIM,
           'NETAPP_MOUNT_PATH': NETAPP_MOUNT_PATH,
           'NETAPP_PVC_CLAIM' : NETAPP_PVC_CLAIM,
           'IGZ_CONTAINER_PATH' : IGZ_CONTAINER_PATH,
           'IGZ_MOUNT_PATH' : IGZ_MOUNT_PATH
            }
```

すべてのステップの関数定義が完了したら、パイプラインを構築できます。この定義には 'kfp' モジュールを使用しますMLRun を使用することと、独自に構築することの違いは、コーディングの簡素化と短縮です。

定義した関数は、 MLRun の「 As _ step 」関数を使用してステップコンポーネントになります。

スナップショットステップの定義

Snapshot 機能を開始し、 v3io をソースとしてマウントします。

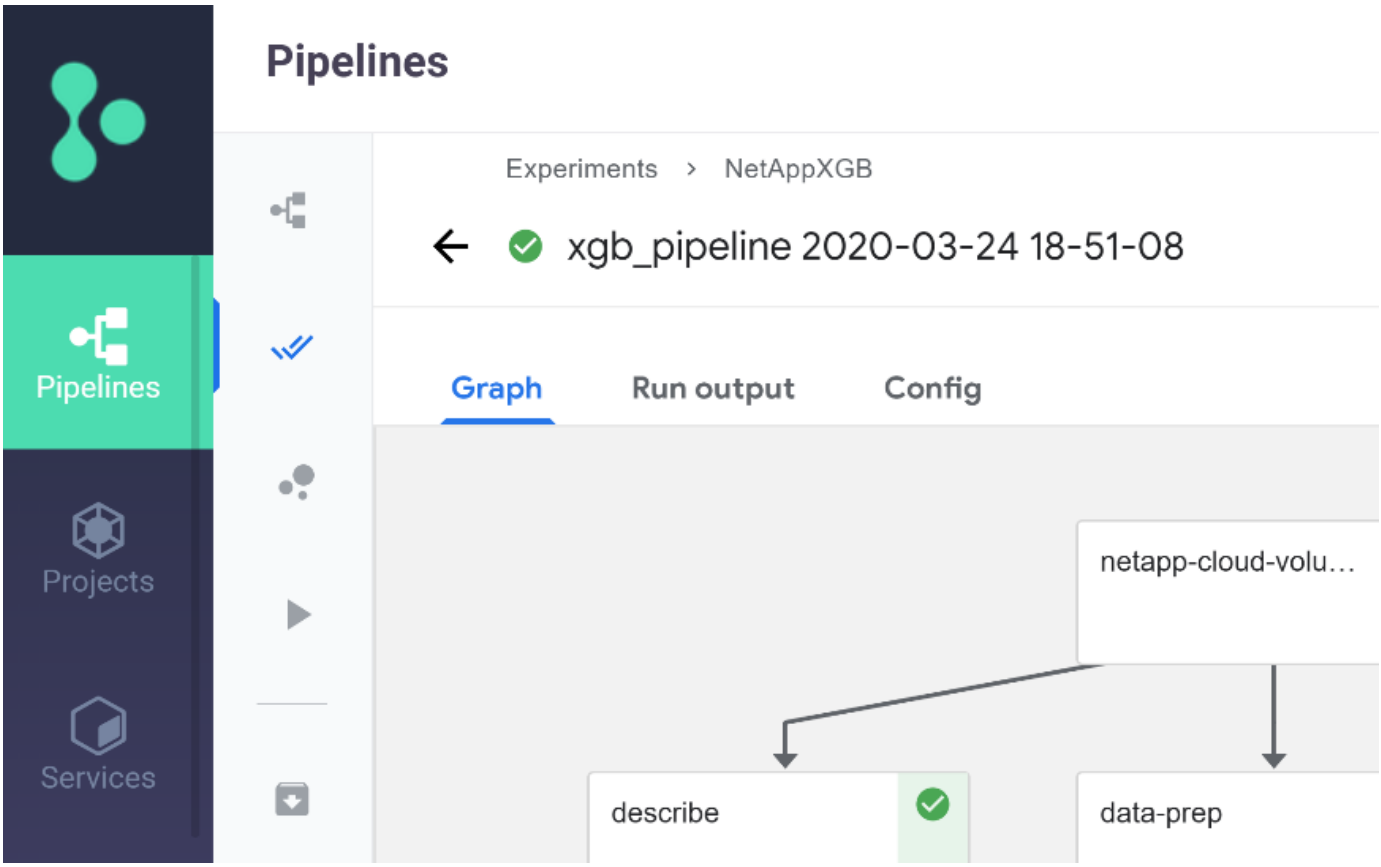
```
snap = snapfn.as_step(NewTask(handler='handler',params=snap_params),
name='NetApp_Cloud_Volume_Snapshot',outputs=['snapVolumeDetails','training
_parquet_file']).apply(mount_v3io())
```

パラメータ	詳細
新しいタスクです	newtask は、実行される関数の定義です。
（ MLRun モジュール）	ハンドラ。呼び出す Python 関数の名前。ノートブックでは name ハンドラーを使用しましたが、必須ではありません。パラメータ実行に渡されたパラメータ。このコードでは、 context.get_param （「パラメータ」）を使用して値を取得します。
ステップとして（ _STEP. ）	名前Kubeflow パイプラインステップの名前。出力：これらは、完了時にステップが辞書に追加する値です。SNAP_CV.ipynb ノートブックを参照してください。mount_v3io()これにより、パイプラインを実行しているユーザーの /User をマウントするステップが構成されます。

```
prep = data_prep.as_step(name='data-prep',
handler='handler',params=params,
                          inputs = {'DATA_DIR':
snap.outputs['snapVolumeDetails']} ,

out_path=artifacts_path).apply(mount_v3io()).after(snap)
```

パラメータ	詳細
入力	前の手順の出力に渡すことができます。この場合、snap.outputs['napVolumeDetails'] は、スナップステップで作成した Snapshot コピーの名前です。
out_path	MLRun モジュール LOG_Artifacts を使用して生成するアーティファクトを配置する場所。

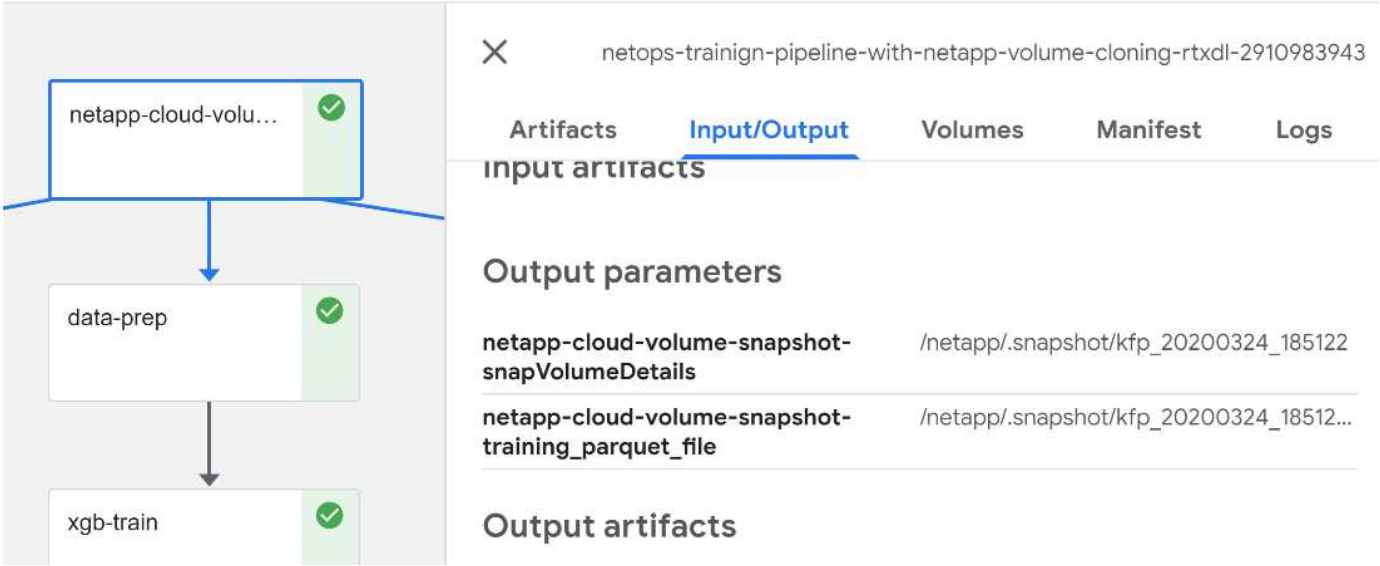
上から下に 'pipeline.ipynb' を実行できます次に、 Iguazio ダッシュボードの Pipelines タブに移動して、Iguazio ダッシュボードの Pipelines タブに示すように、進捗状況を監視できます。



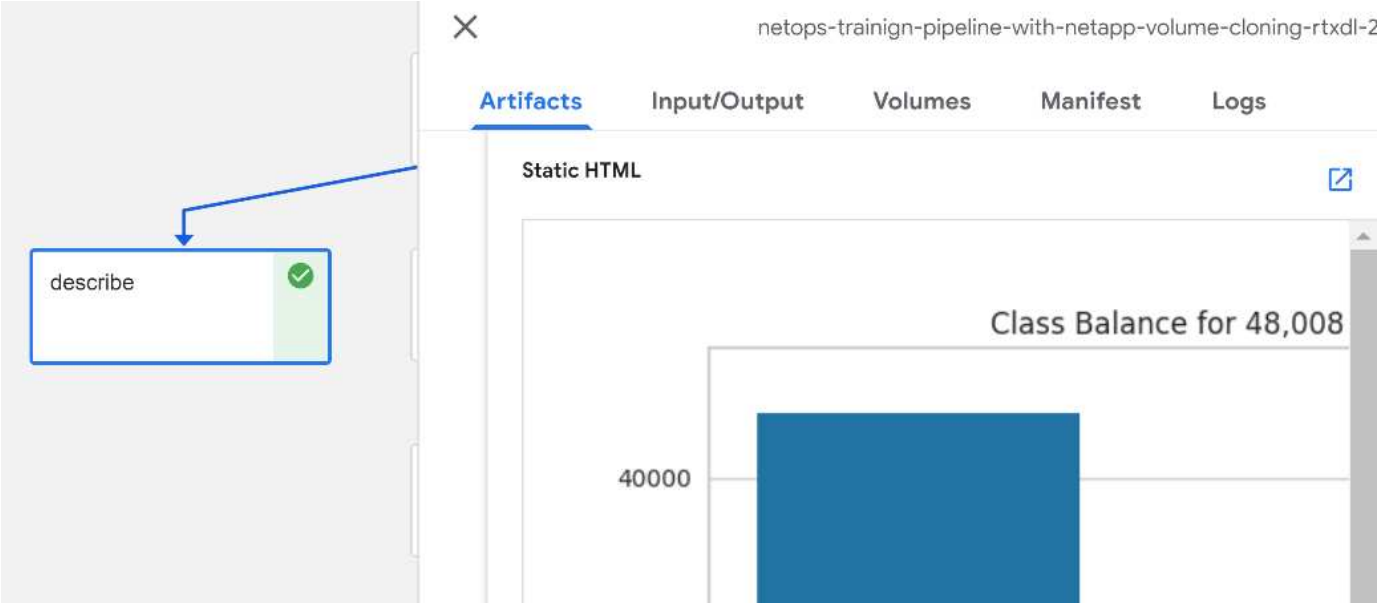
トレーニングステップの精度はすべての実行で記録されているため、トレーニングの正確性の記録に示されているように、各テストの精度の記録があります。

Run name	Status	Duration	Pipeline Version	Recurring ...	Start time	accuracy
xgb_pipeline 2020-03-24 18-51-...	✓	0:08:43	[View pipeline]	-	3/24/2020, 2:51:09 PM	0.985
xgb_pipeline 2020-03-19 13-31-...	✓	0:08:14	[View pipeline]	-	3/19/2020, 9:31:19 AM	0.980
xgb_pipeline 2020-03-18 12-56-...	✓	0:08:11	[View pipeline]	-	3/18/2020, 8:56:08 AM	0.990
xgb_pipeline 2020-03-17 19-49-...	✓	0:08:03	[View pipeline]	-	3/17/2020, 3:49:31 PM	0.985
xgb_pipeline 2020-03-17 18-34-...	✓	0:05:54	[View pipeline]	-	3/17/2020, 2:34:56 PM	0.980
xgb_pipeline 2020-03-17 17-34-...	✓	0:04:48	[View pipeline]	-	3/17/2020, 1:34:16 PM	0.982
xgb_pipeline 2020-03-17 17-01-...	✓	0:05:25	[View pipeline]	-	3/17/2020, 1:01:58 PM	0.987
xgb_pipeline 2020-03-16 16-47-...	✓	0:06:08	[View pipeline]	-	3/16/2020, 12:47:19 ...	0.983
xgb_pipeline 2020-03-16 13-57-...	✓	0:05:18	[View pipeline]	-	3/16/2020, 9:57:03 AM	0.980

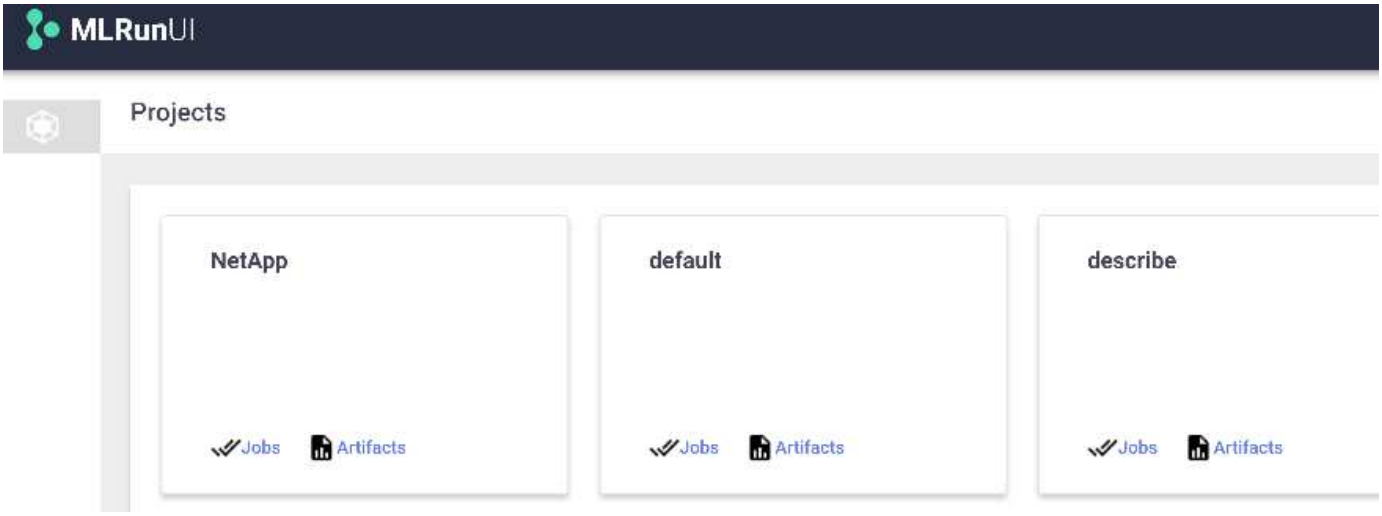
Snapshot ステップを選択すると、この実験を実行するために使用された Snapshot コピーの名前が表示されます。



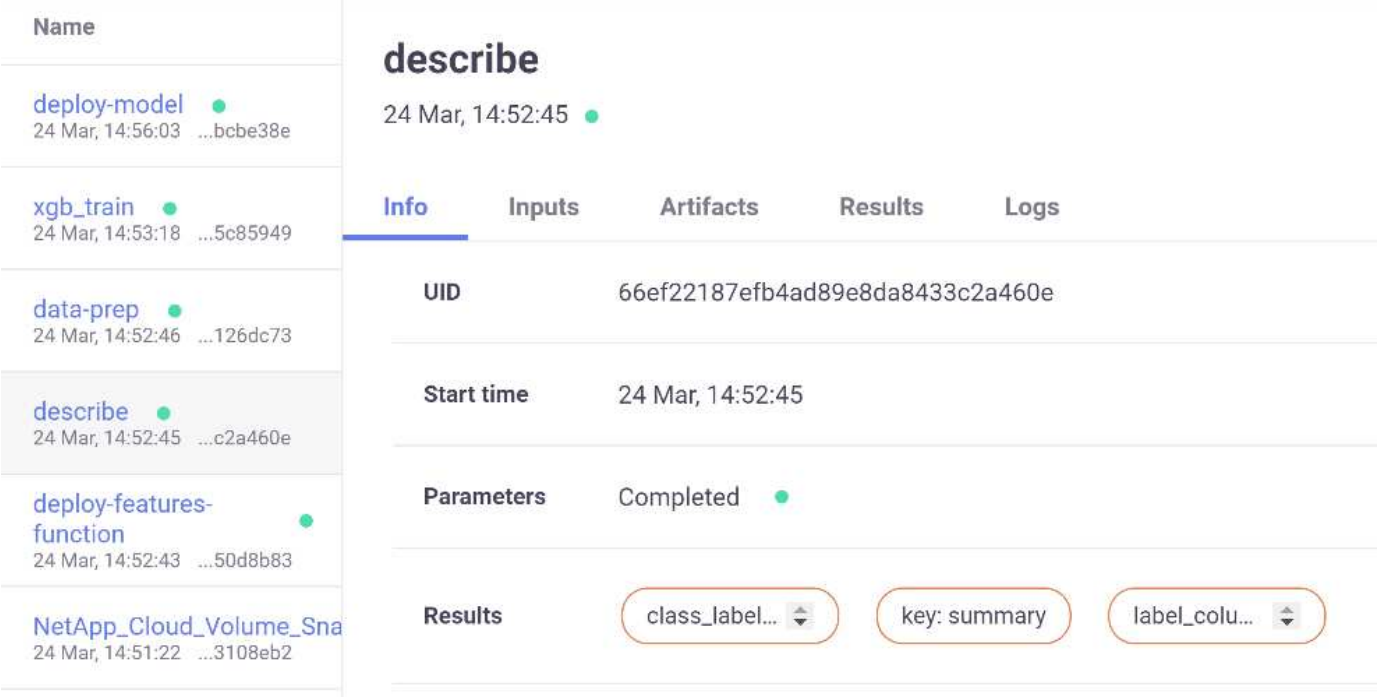
ここで説明する手順には、使用した指標を確認するための視覚的なアーティファクトがあります。を展開すると、次の図のように全プロットを表示できます。



MLRun API データベースは、プロジェクトごとに編成された各ランの入力、出力、およびアーティファクトも追跡します。各ランの入力、出力、およびアーティファクトの例を次の図に示します。



各ジョブについて、追加の詳細情報が保存されます。



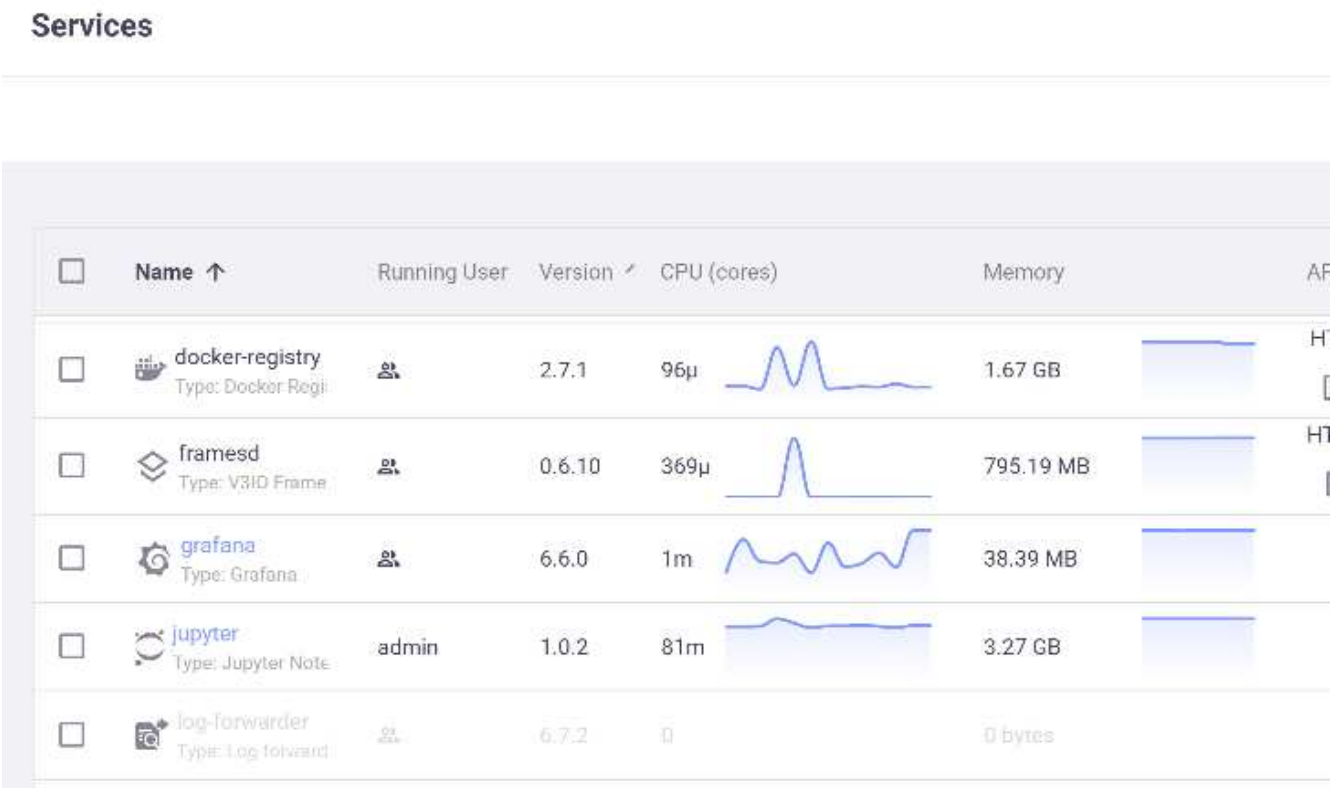
MLRun の詳細については、このドキュメントで説明している内容を参照してください。ステップと関数の定義を含むアルアーティファクトは、 API データベースに保存したり、バージョン管理したり、個別に呼び出すことも、完全なプロジェクトとして呼び出すこともできます。プロジェクトを保存して Git にプッシュし、後で使用することもできます。詳細については、を参照してください "MLRun GitHub サイト"。

Grafana ダッシュボードを導入します

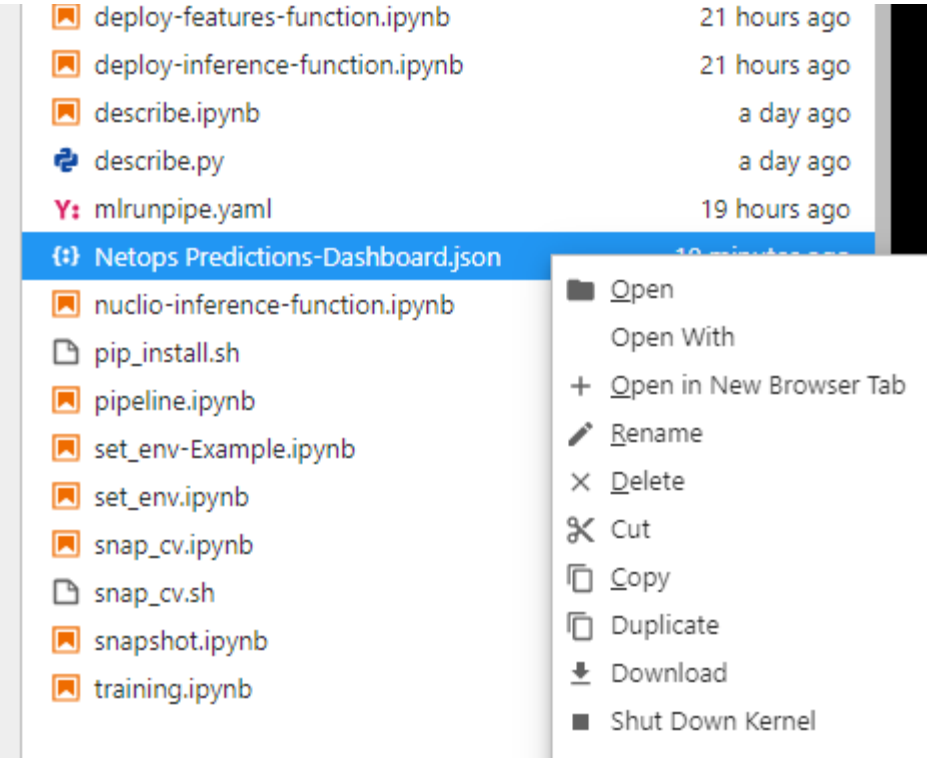
すべてのデータを導入したら、新しいデータに対して推論を実行します。このモデルは、ネットワークデバイス機器の障害を予測します。予測の結果は、 Iguazio 時系列テーブルに格納されます。Iguazio のセキュリティおよびデータアクセスポリシーと統合されたプラットフォームで、 Grafana を使用して結果を表示できます。

ダッシュボードを導入するには、指定した JSON ファイルをクラスタ内の Grafana インターフェイスにインポートします。

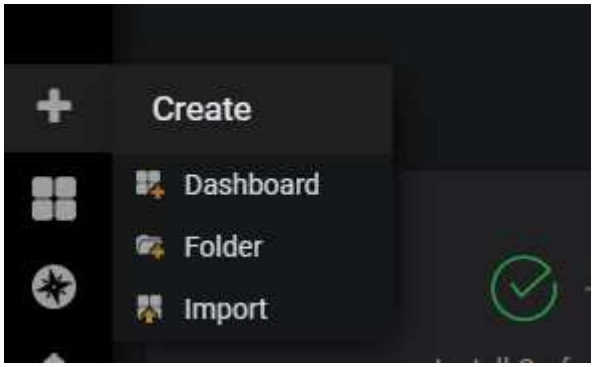
1. Grafana サービスが実行されていることを確認するには、 Services の下を参照してください。



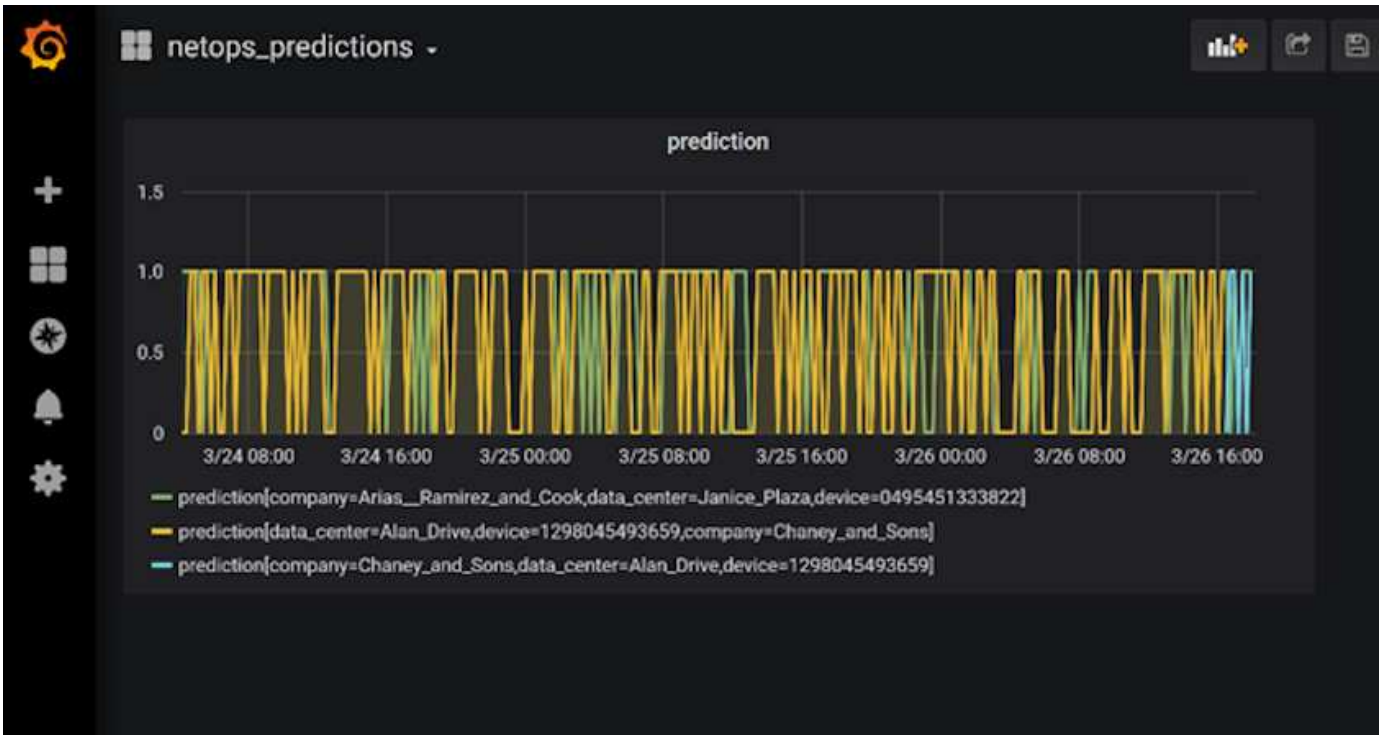
2. インスタンスが存在しない場合は、 [サービス] セクションからインスタンスを展開します。
   a. [新しいサービス] をクリックします。
   b. リストから Grafana を選択します。
   c. デフォルトを受け入れます。
   d. 次のステップをクリックします。
   e. ユーザ ID を入力します。
   f. [サービスの保存] をクリックします
   g. 上部の [Apply Changes] をクリックします。
3. ダッシュボードを展開するには、 Jupyter インターフェイスから「 NetopsPredictions - Dashboard.json 」ファイルをダウンロードします。



4. Services セクションで Grafana を開き、ダッシュボードをインポートします。



5. Upload `*.json ' File をクリックして、以前にダウンロードしたファイル (`NetopsPredictions - Dashboard.json ') を選択します。アップロードが完了すると、ダッシュボードが表示されます。



デプロイクリーンアップ機能

大量のデータを生成する場合は、すべてをクリーンで整理することが重要です。これを行うには 'cleanup.ipynb' ノートブックを使用してクリーンアップ機能を配備します

利点

ネットアップと Iguazio は、 Kubeflow 、 Apache Spark 、 TensorFlow などの必須フレームワークや、 Docker や Kubernetes などのオーケストレーションツールを構築することで、 AI アプリケーションや ML アプリケーションの導入を迅速化し、簡易化します。ネットアップと Iguazio は、エンドツーエンドのデータパイプラインを統合することで、多くの高度なコンピューティングワークロードに固有のレイテンシと複雑さを軽減し、開発と運用のギャップを効果的に解消します。データサイエンティストは、大規模なデータセットに対してクエリを実行し、トレーニングフェーズ中にデータやアルゴリズムのモデルを権限のあるユーザと安全に共有できます。コンテナ化されたモデルを本番環境で使用できるようになったら、開発環境から運用環境に簡単に移行できます。

まとめ

独自の AI / ML パイプラインを構築する場合、アーキテクチャ内のコンポーネントの統合、管理、セキュリティ、およびアクセス性の設定は困難な作業です。開発者に環境へのアクセスと管理を許可することには、もう 1 つの課題があります。

ネットアップと Iguazio を組み合わせることで、これらのテクノロジをマネージドサービスとして統合し、テクノロジの採用を促進し、新しい AI / ML アプリケーションの市場投入期間を短縮できます。

TR-4915：『Data movement with E-Series and BeeGFS for AI and analytics workflows』

ネットアップ、Cody HarrymanとRyan Rodineです

TR-4915では、任意のデータリポジトリから、NetApp EシリーズSANストレージをサポートするBeeGFSファイルシステムにデータを移動する方法について説明します。人工知能（AI）や機械学習（ML）のアプリケーションの場合、お客様は、数ペタバイトを超える大規模なデータセットを、モデル開発のためにBeeGFSクラスタに移動する必要があることがあります。このドキュメントでは、NetApp XCPおよびNetApp BlueXPのコピーと同期ツールを使用してこれを実現する方法について説明します。

"TR-4915：『Data movement with E-Series and BeeGFS for AI and analytics workflows』"

ユースケース

AIと機密性の高い推論を担当-ネットアップのAIとProtopia Image Transformation

TR-4928：『Responsible AI and Confidential Inferencing - NetApp AI with Protopia Image and Data Transformation』

Sathish Thyagarajan氏、Michael Oglesby氏、NetApp Byung Hoon Ahn氏、Jennifer Cwagenberg氏、Protopia氏

画像の撮影や画像処理の出現とのコミュニケーションには、視覚的な解釈が不可欠です。デジタル画像処理における人工知能（AI）は、がんやその他の疾患識別のための医療分野などの新たなビジネスチャンスをもたらします。また、地球空間での視覚分析による環境ハザード調査、パターン認識、犯罪と闘うためのビデオ処理などにも活用できます。しかし、この機会には特別な責任も伴います。

AIを導入する意思決定が増えるほど、データのプライバシー、セキュリティ、法律、倫理、規制に関する問題に関連するリスクを受け入れることができます。責任あるAIは、大企業のAIに欠かせない信頼とガバナンスを企業や政府機関が構築できるようにするための実践を可能にします。本ドキュメントでは、ネットアップが3つの異なるシナリオで検証したAI推論解決策 について説明します。この検証では、Protopiaデータ難読化ソフトウェアとネットアップのデータ管理テクノロジを併用して、機密データをプライベート化し、リスクと倫理的な懸念を軽減します。

毎日何百万もの画像が生成され、消費者とビジネスエンティティの両方がさまざまなデジタルデバイスを使用します。その結果、データとコンピューティングのワークロードが急増し、企業はクラウドコンピューティングプラットフォームを利用して、拡張性と効率性を高めることになります。一方、画像データに含まれる機密情報に関するプライバシーの懸念は、パブリッククラウドへの転送によって生じます。画像処理AIシステムの導入における主な障壁は、セキュリティとプライバシーの欠如です。

また、もあります "イレイジャーコーディングの権利" GDPRでは、組織がすべての個人データを消去するように要求する権利が個人に与えられます。また、もあります "プライバシー法"は、公正な情報慣行のコードを確立します。写真などのデジタル画像は、GDPRに基づく個人データを構成し、データの収集、処理、消去の方法を規定します。これを怠るとGDPRに準拠できず、組織に深刻な損害を与える可能性があるコンプライアンス違反に対する多額の罰金が科せられる可能性があります。プライバシーに関する原則は、機械学習（ML）モデルとディープラーニング（DL）モデルの予測において公平性を確保し、プライバシーや規制へのコンプライアンス違反に伴うリスクを軽減する、責任あるAIを実装するためのバックボーンとなっています。

このドキュメントでは、プライバシーの保護と責任あるAI解決策 の導入に関連する画像難読化機能を使用した、または使用しない、3つの異なるシナリオにおける検証済み設計解決策 について説明します。

- シナリオ**1.** Jupyterノートブック内でのオンデマンド推論。

- シナリオ**2.** Kubernetesでのバッチ推論。
- シナリオ**3.** NVIDIA * Triton推論サーバ。

この解決策 では、フェース検出データセットとベンチマーク（FDDB）を使用します。これは、フェース検出の問題を調査するために設計されたフェース領域のデータセットで、PyTorch機械学習フレームと組み合わせて、フェースボックスを実装するためのフレームワークです。このデータセットには、さまざまな解像度の2845枚の画像セットに5171個の面の注釈が含まれています。さらに、このテクニカルレポートでは、この解決策 を適用できる状況において、ネットアップのお客様やフィールドエンジニアから収集した解決策 の領域と関連するユースケースを紹介します。

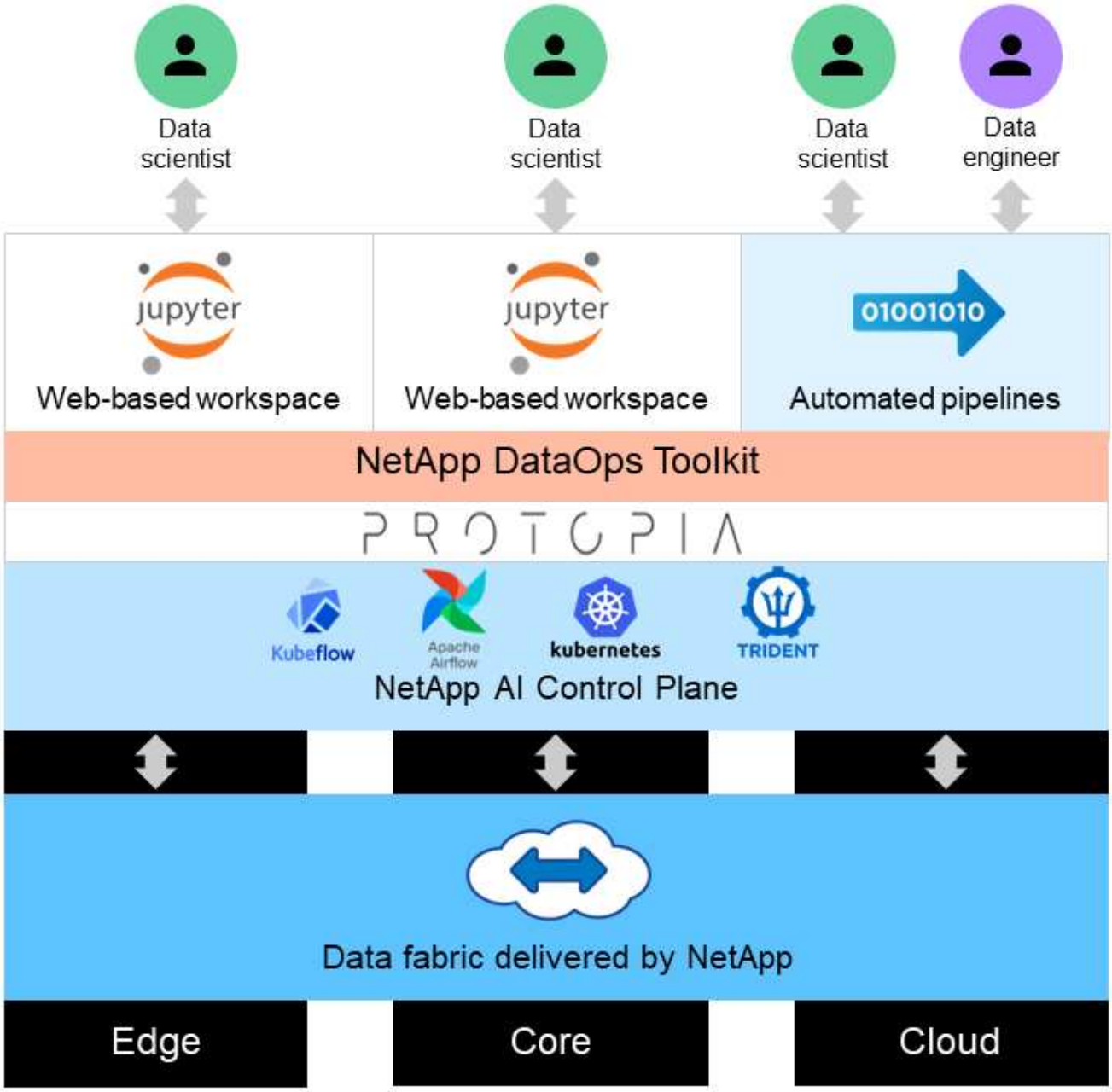
対象読者

このテクニカルレポートは、次のような方を対象としています。

- 公共スペースでの顔画像処理に関するデータ保護やプライバシーの問題に対処するために、責任あるAIの設計と導入を希望するビジネスリーダーおよびエンタープライズアーキテクト。
- データサイエンティスト、データエンジニア、AI /機械学習（ML）研究者、AI / MLシステム開発者は、プライバシーの保護と維持を目指します。
- GDPR、CCPA、国防総省（DoD）や政府機関のプライバシー法などの規制基準に準拠するAI / MLモデルおよびアプリケーション向けのデータ難読化ソリューションを設計するエンタープライズアーキテクト。
- データサイエンティストとAIエンジニアは、機密情報を保護するディープラーニング（DL）モデルとAI / ML / DL推論モデルを効率的に導入する方法を探しています。
- エッジ推論モデルの導入と管理を担当するエッジデバイスマネージャとエッジサーバ管理者。

解決策アーキテクチャ

この解決策 は、GPUの処理能力と従来のCPUを併用することで、大規模なデータセットにおけるAIワークロードのリアルタイムおよびバッチ推論を処理するように設計されています。この検証では、MLのプライバシー保護推論と、責任のあるAI導入を求めている組織に必要な最適なデータ管理が実証されています。この解決策 は、エッジコンピューティングとクラウドコンピューティングを連携させる単一ノードまたはマルチノードのKubernetesプラットフォームに適したアーキテクチャを提供します。このプラットフォームは、オンプレミスのコアソフトウェアであるNetApp ONTAP AI、NetApp DataOpsツールキット、およびJupyter LabとCLIインターフェイスを使用したProtopia難読化ソフトウェアと連携します。次の図は、ネットアップのDataOpsツールキットとProtopiaを使用したデータファブリックの論理アーキテクチャの概要を示しています。



Protopia難読化ソフトウェアは、NetApp DataOpsツールキットの上でシームレスに実行され、ストレージサーバから離れる前にデータを変換します。

解決策エリア

デジタル画像処理には多くの利点があり、多くの組織が視覚表現に関連するデータを最大限に活用できるようになっています。このネットアップとProtopia解決策 は、ML / DLのライフサイクル全体にわたってAI / MLデータを保護し、民営化するための、独自のAI推論設計を提供しています。お客様は機密データの所有権を維持し、プライバシーに関する懸念を解消して、拡張性と効率性を高めるためにパブリッククラウドまたはハイブリッドクラウドの導入モデルを使用できます。また、エッジでAI推論を導入することもできます。

環境インテリジェンス

業界は、環境上の危険がある領域で地理空間分析を活用する方法が数多くあります。政府や公共事業部は、公衆衛生や天候に関する実用的な洞察を得ることができ、パンデミックや野生火災などの自然災害時の国民へのアドバイスを向上させることができます。たとえば、空港や病院などの公共スペースで新型コロナウイルス感染症の患者を特定する際に、該当する個人のプライバシーを侵害することなく、必要な安全対策を講じるために各当局および近隣の公衆に警告を出すことができます。

エッジデバイスのウェアラブル

軍事演習や戦場では、エッジでのAI推論をウェアラブルデバイスとして使用することで、兵士の健康状態を追跡し、運転者の行動を監視し、軍車に接近する際の安全および関連するリスクについて当局に警告しながら、兵士のプライバシーを守り、保護することができます。軍事の未来は、Internet of Battlefield Things（IOBT）とInternet of Military Things（IANMT）でハイテク化を進めています。この製品は、兵士が敵を特定し、高速エッジコンピューティングを使用して戦闘でより良いパフォーマンスを発揮するのに役立ちます。ドローンやウェアラブル・ギアなどのエッジデバイスから収集した視覚的なデータを保護し、保護することは、ハッカーや敵をベイに維持するために不可欠です。

非燃焼型避難作業

非戦闘員避難行動（Neos）は、米国市民および国民、DoD民間人、および指定された人（HN）および第3国国民（TCN）の避難を支援するために、国防総省によって実施されます。所定の管理制御では、主に手動による退避対象者スクリーニングプロセスが使用されます。ただし、高度に自動化されたAI / MLツールとAI / MLビデオ難読化テクノロジを組み合わせることで、避難先の識別、退避対象の追跡、脅威スクリーニングの精度、セキュリティ、速度を向上させることができます。

ヘルスケアおよび生物医学研究

画像処理は、CT（Computed Tomography）またはMRI（Magnetic Resonance Imaging）から取得した3D画像から外科的な計画を立てる際の病理を診断するために使用されます。HIPAAのプライバシールールは、すべての個人情報および写真などのデジタル画像に関して、組織がデータを収集、処理、消去する方法を規定しています。HIPAA Safe Harbor規制の下でデータが共有可能となるようにするには、フルフェイスの写真画像と、それに相当する画像を削除する必要があります。構造CT/MR画像から個人の顔の特徴を隠すために使用される匿名化アルゴリズムや頭蓋骨除去アルゴリズムなどの自動化技術は、生物医学研究機関のデータ共有プロセスの重要な要素となっています。

AI / ML分析のクラウド移行

エンタープライズのお客様は従来、AI / MLモデルのトレーニングを受け、オンプレミスで導入してきました。スケールメリットと効率性の理由から、AI / ML機能をパブリッククラウド、ハイブリッドクラウド、マルチクラウド環境に移行するお客様が増えています。ただし、他のインフラにどのようなデータを公開できるかによって制限されます。ネットアップのソリューションは、に求められるあらゆるサイバーセキュリティの脅威に対処します "データ保護" また、セキュリティ評価を行い、Protopiaのデータ変換と組み合わせることで、画像処理AI / MLワークロードをクラウドに移行する際のリスクを最小限に抑えることができます。

他の業界のエッジコンピューティングとAI推論のその他のユースケースについては、を参照してください "TR-4886『エッジでのAI推論』" NetApp AIブログ、 "インテリジェンスとプライバシー"。

テクノロジの概要

このセクションでは、この解決策 を完了するために必要なさまざまな技術コンポーネントの概要を説明します。

プロトピア

Protopia AIは、今日の市場における機密性の高い推論のための、目立たないソフトウェア型解決策 を提供します。Protopia解決策 は、機密情報の漏洩を最小限に抑えることで、推論サービスに対する比類のない保護機能を提供します。AIには、データレコード内の情報のみが提供されます。この情報は、実際にはタスクを実行するために不可欠であり、それ以上何も必要ありません。ほとんどの推論タスクでは、すべてのデータレコードに存在するすべての情報が使用されるわけではありません。画像、音声、ビデオ、構造化された表形式データのどれをAIが消費しているかにかかわらず、Protopiaは推論サービスが必要としているものだけを提供します。特許取得済みのコアテクノロジーは、数学的にキュレーションされたノイズを使用してデータを変革し、特定のMLサービスでは必要のない情報を蓄積します。この解決策 はデータをマスクするのではなく、キュレーションされたランダムなノイズを使用してデータ表現を変更します。

Protopia解決策 は、モデルの機能に関して、入力フィーチャースペースの関連情報を保持する勾配ベースの摂動最大化方法としてリプレゼンテーションを変更する問題を計算します。このディスカバリプロセスは、MLモデルのトレーニング終了時に微調整パスとして実行されます。このパスによって確率分布のセットが自動的に生成されると、オーバーヘッドが低いデータ変換によって、これらの分布から生成されたノイズサンプルがデータに適用され、これが難読化されてから推論モデルに渡されます。

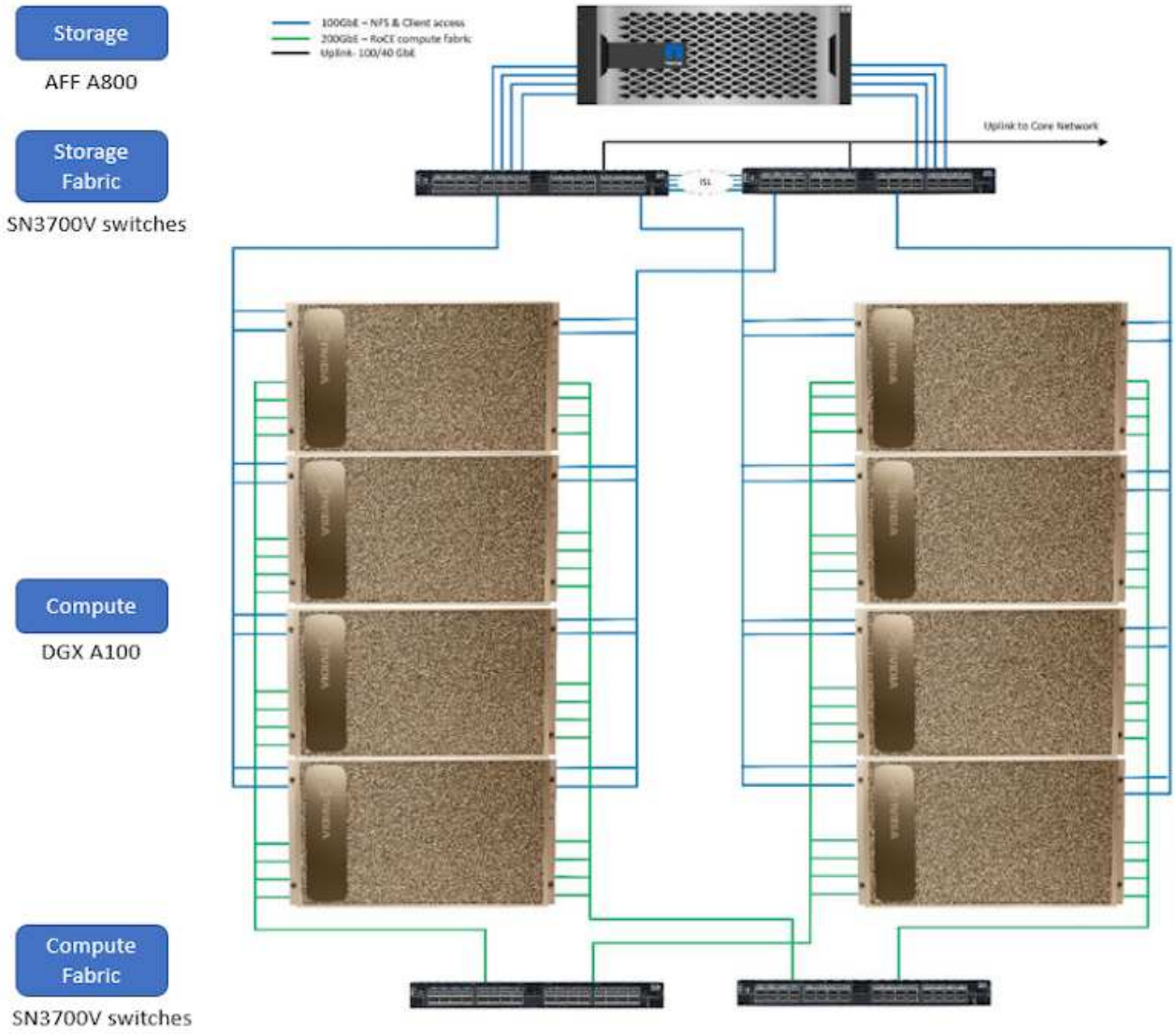
NetApp ONTAP AI

DGX A100システムとネットアップのクラウド対応ストレージシステムを基盤とするNetApp ONTAP AIリファレンスアーキテクチャは、ネットアップとNVIDIAによって開発、検証されました。IT 組織には、次のようなメリットをもたらすアーキテクチャが提供されます。

- 設計の複雑さを解消
- コンピューティングとストレージを個別に拡張できます
- 小規模構成から始めて、シームレスに拡張できます
- は、さまざまなパフォーマンスとに対応するさまざまなストレージオプションを提供します コストポイント

ONTAP AIは、DGX A100システムとNetApp AFF A800ストレージシステムを最先端のネットワークと緊密に統合します。ONTAP AIは、設計の複雑さと推測に頼らず、AI導入を簡易化します。小規模構成から始めて、システムを停止することなく拡張でき、エッジからコア、クラウドまで、データをインテリジェントに管理できます。

次の図は、DGX A100システムを使用したONTAP AIソリューションファミリーのいくつかのバリエーションを示しています。AFF A800システムのパフォーマンスは、最大8台のDGX A100システムで検証されます。ONTAP クラスタにストレージコントローラペアを追加することで、アーキテクチャを複数のラックに拡張して、パフォーマンスがリニアに向上したDGX A100システムとペタバイト規模のストレージ容量をサポートできます。このアプローチにより、使用されるDLモデルのサイズと必要なパフォーマンス指標に基づいて、コンピューティングとストレージの比率を個別に変更できる柔軟性が得られます。



追加情報 About ONTAP AIについては、を参照してください "NVA-1153：NVIDIA DGX A100システムとMellanox Spectrumイーサネットスイッチを搭載したNetApp ONTAP AI"

NetApp ONTAP

ネットアップが提供する最新世代のストレージ管理ソフトウェアONTAP 9.11を使用すれば、インフラを刷新し、クラウド対応データセンターに移行できます。ONTAP は、業界をリードするデータ管理機能を活用して、データの格納場所に関係なく、単一のツールセットでデータの管理と保護を実現します。エッジ、コア、クラウドなど、必要な場所に自由にデータを移動することもできます。ONTAP 9.11には、データ管理を簡易化し、重要なデータを高速化、保護し、ハイブリッドクラウドアーキテクチャ全体で次世代インフラ機能を実現するためのさまざまな機能が搭載されています。

NetApp DataOps ツールキット

NetApp DataOpsツールキットはPythonライブラリで、開発者、データサイエンティスト、DevOpsエンジニア、データエンジニアは、新しいデータボリュームやJupyterLabワークスペースのほぼ瞬時のプロビジョニング、データボリュームやJupyterLabワークスペースのほぼ瞬時のクローニングなど、さまざまなデータ管理タスクを簡単に実行できます。 トレーサビリティやベースライン化のために、データボリュームまたはJupyterLabワークスペースのスナップショットをほぼ瞬時に作成できます。このPythonライブラリは、任意のPythonプログラムまたはJupyterノートブックに読み込むことができるコマンドラインユーティリティまた

は関数のライブラリとして機能します。

NVIDIA Triton 推論サーバ

NVIDIA Triton Inference Serverは、オープンソースの推論サービスソフトウェアです。モデルの導入と実行を標準化し、高速で拡張性に優れたAIを本番環境に提供できます。Triton Inference Serverは、GPUベースまたはCPUベースのインフラ上のあらゆるフレームワークからトレーニング済みAIモデルを導入、実行、拡張できるため、AI推論が合理化されます。Triton Inference Serverは、TensorFlow、NVIDIA TensorRT、PyTorch、MXNetなどの主要なフレームワークをすべてサポートします。 OpenVNOなど。TritonはKubernetesと統合し、オーケストレーションと拡張を実現します。主要なパブリッククラウドのAIプラットフォームとKubernetesプラットフォームで使用できます。また、多くのMLOpsソフトウェアソリューションと統合されています。

PyTorch

"PyTorch" はオープンソースのMLフレームワークです。GPUとCPUを使用するディープラーニング用に最適化されたテンソルライブラリです。PyTorchパッケージには多次元テンソル用のデータ構造が含まれており、他の有用なユーティリティ間でテンソルを効率的にシリアル化するための多くのユーティリティを提供します。また、コンピューティング機能を備えたNVIDIA GPUでテンソル計算を実行できるCUDA対応製品もあります。この検証では、OpenCV-Python (CV2)ライブラリを使用してモデルを検証しながら、Pythonで最も直感的なコンピュータビジョンの概念を活用しています。

データ管理を簡易化

データ管理は、AIアプリケーションの運用やAI / MLデータセットのトレーニングに適切なリソースを使用できるように、エンタープライズIT運用とデータサイエンティストにとって非常に重要です。以下に記載するネットアップテクノロジに関する追加情報 は、この検証の対象外ですが、導入環境によっては関連性がある場合もあります。

ONTAP データ管理ソフトウェアには、運用を合理化および簡易化し、総運用コストを削減するための次の機能が含まれています。

- インラインデータコンパクション、強化された重複排除：データコンパクションはストレージブロック内の無駄なスペースを削減し、重複排除は実効容量を大幅に増やします。この環境データはローカルに格納され、データはクラウドに階層化されます。
- 最小、最大、アダプティブのQuality of Service（AQoS）。きめ細かいサービス品質（QoS）管理機能により、高度に共有された環境で重要なアプリケーションのパフォーマンスレベルを維持できます。
- NetApp FabricPool の略。Amazon Web Services（AWS）、Azure、NetApp StorageGRID ストレージ解決策 など、パブリッククラウドとプライベートクラウドのストレージオプションへコールドデータを自動的に階層化します。FabricPool の詳細については、を参照してください "TR-4598：『FabricPool best bests』"。

データの高速化と保護

ONTAP は、卓越したパフォーマンスとデータ保護を実現し、以下の方法でこれらの機能を拡張します。

- パフォーマンスとレイテンシの低下：ONTAP は、可能なかぎり最小のレイテンシで最高のスループットを提供します。
- データ保護ONTAP には、組み込みのデータ保護機能が用意されており、すべてのプラットフォームを共通の管理機能で管理できます。
- NetApp Volume Encryption（NVE）：ONTAP は、オンボードと外部キー管理の両方をサポートし、ボリ

ュームレベルでのネイティブな暗号化を実現します。

- マルチテナンシーおよび多要素認証ONTAP を使用すると、最高レベルのセキュリティでインフラリソースを共有できます。

将来のニーズにも対応できるインフラ

ONTAP は、次の機能を備えており、要件が厳しく、絶えず変化するビジネスニーズに対応できます。

- シームレスな拡張とノンストップオペレーションONTAP を使用すると、既存のコントローラとスケールアウトクラスタに無停止で容量を追加できます。NVMe や 32Gb FC などの最新テクノロジへのアップグレードも、コストのかかるデータ移行やシステム停止を行わずに実行できます。
- クラウドへの接続：ONTAP は、すべてのパブリッククラウドでSoftware-Defined Storage（ONTAP Select ）とクラウドネイティブインスタンス（NetApp Cloud Volumes Service ）のオプションを選択できる、マルチクラウドに対応した最もクラウド対応のストレージ管理ソフトウェアです。
- 新しいアプリケーションとの統合：ONTAP は、既存のエンタープライズアプリケーションをサポートするインフラを使用して、自律走行車、スマートシティ、インダストリー4.0などの次世代プラットフォームやアプリケーション向けにエンタープライズクラスのデータサービスを提供します。

ネットアップアストラコントロール

ネットアップの Astra 製品ファミリーは、オンプレミスとパブリッククラウドの Kubernetes アプリケーション向けに、ネットアップのストレージテクノロジとデータ管理テクノロジを基盤とするストレージサービスとアプリケーション対応データ管理サービスを提供します。Kubernetesアプリケーションのバックアップ、データの別のクラスタへの移行、作業用アプリケーションのクローンの瞬時作成を簡単に実行できます。パブリッククラウドで実行されているKubernetesアプリケーションを管理する必要がある場合は、のドキュメントを参照してください "Astra 制御サービス"。Astra Control Service は、 Google Kubernetes Engine （ GKE ）および Azure Kubernetes Service （ AKS ）で Kubernetes クラスタのアプリケーション対応データ管理を提供する、ネットアップが管理するサービスです。

ネットアップアストラト **Trident**

アストラ "Trident" ネットアップは、 Docker と Kubernetes 向けのオープンソースの動的ストレージオーケストレーションツールであり、永続的ストレージの作成、管理、使用を簡易化します。Kubernetesネイティブアプリケーションである Tridentは、Kubernetesクラスタ内で直接実行されます。Trident を使用すると、 DL コンテナイメージをネットアップストレージにシームレスに導入し、エンタープライズクラスの AI コンテナ環境を実現できます。Kubernetesユーザ（ML開発者、データサイエンティストなど）は、オーケストレーションとクローニングを作成、管理、自動化し、ネットアップテクノロジを基盤とする高度なデータ管理機能を活用できます。

NetApp BlueXPのコピーと同期

"BlueXPのコピーと同期" 迅速かつセキュアなデータ同期を実現するネットアップのサービスです。オンプレミスのNFSまたはSMBファイル共有間でファイルを転送する必要があるかどうかにかかわらず、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、 IBM Cloud Object StorageのBlueXP Copy and Syncなら、必要な場所に迅速かつセキュアにファイルを移動できます。転送されたデータは、ソースとターゲットの両方で完全に使用できます。BlueXPのCopyとSynccは、事前定義されたスケジュールに基づいて継続的にデータを同期し、差分のみを移動するため、データレプリケーションにかかる時間とコストを最小限に抑えることができます。BlueXPのCopy and Syncは、セットアップと使用が非常に簡単なソフトウェアサービス（SaaS）ツールです。BlueXPのCopyとSyncによってトリガーされるデータ転送は、データブローカーによって実行されます。BlueXPのCopy and Syncデータブローカーは、AWS、Azure、Google Cloud Platform、オンプレミスに導入できます。

NetApp BlueXPの分類

強力なAIアルゴリズム、 "NetApp BlueXPの分類" データ資産全体の管理とデータガバナンスを自動化します。コスト削減を容易に特定し、コンプライアンスやプライバシーに関する懸念を特定し、最適化の機会を見つけることができます。BlueXPの分類ダッシュボードでは、重複データを特定して冗長性の排除、個人データ、非個人データ、機密データのマッピング、機密データや異常のアラートの有効化を行うための分析情報を提供します。

テストと検証の計画

この解決策 設計では、次の3つのシナリオが検証されました。

- Kubernetes向けNetApp DataOpsツールキットを使用してオーケストレーションされたJupyterLabワークスペース内で、Protopia難読化を使用するかどうかに関係なく、推論タスクを実行します。
- Kubernetesでの一括推論ジョブ。Protopia難読化を使用するかどうかにかかわらず、NetApp DataOps Toolkit for Kubernetesを使用してオーケストレーションされたデータボリュームを使用します。
- Kubernetes向けNetApp DataOpsツールキットを使用してオーケストレーションされた、NVIDIA Triton Inference Serverインスタンスを使用した推論タスク。Triton推論APIを呼び出す前に、Protopia難読化を画像に適用して、ネットワーク経由で送信されるデータを難読化する必要がある一般的な要件をシミュレートしました。このワークフローは、信頼できるゾーン内でデータが収集され、推論のためにその信頼できるゾーンの外部に渡される必要があるユースケースに該当します。Protopia難読化を使用しないと'機密データが信頼ゾーンから離れることなく'このタイプのワークフローを実装できません

設定をテストします

次の表に、解決策 設計検証環境の概要を示します。

コンポーネント	バージョン
Kubernetes	1.1.6
NetApp Astra Trident CSIドライバ	22.01.0
Kubernetes向けNetApp DataOpsツールキット	2.3.0
NVIDIA Triton 推論サーバ	21.11-py3.

手順をテストします

このセクションでは、検証を完了するために必要なタスクについて説明します。

前提条件

このセクションで説明するタスクを実行するには、次のツールをインストールして設定したLinuxまたはmacOSホストにアクセスできる必要があります。

- Kubectl（既存のKubernetesクラスタへのアクセスを設定）
  - インストールと設定の手順については、を参照してください "こちらをご覧ください"。
- Kubernetes向けNetApp DataOpsツールキット
  - インストール手順が記載されています "こちらをご覧ください"。

シナリオ**1–JupyterLab**におけるオンデマンド推論

1. AI / ML推論ワークロード用のKubernetesネームスペースを作成します。

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

2. NetApp DataOpsツールキットを使用して、推論を実行するデータを格納する永続的ボリュームをプロビジョニングします。

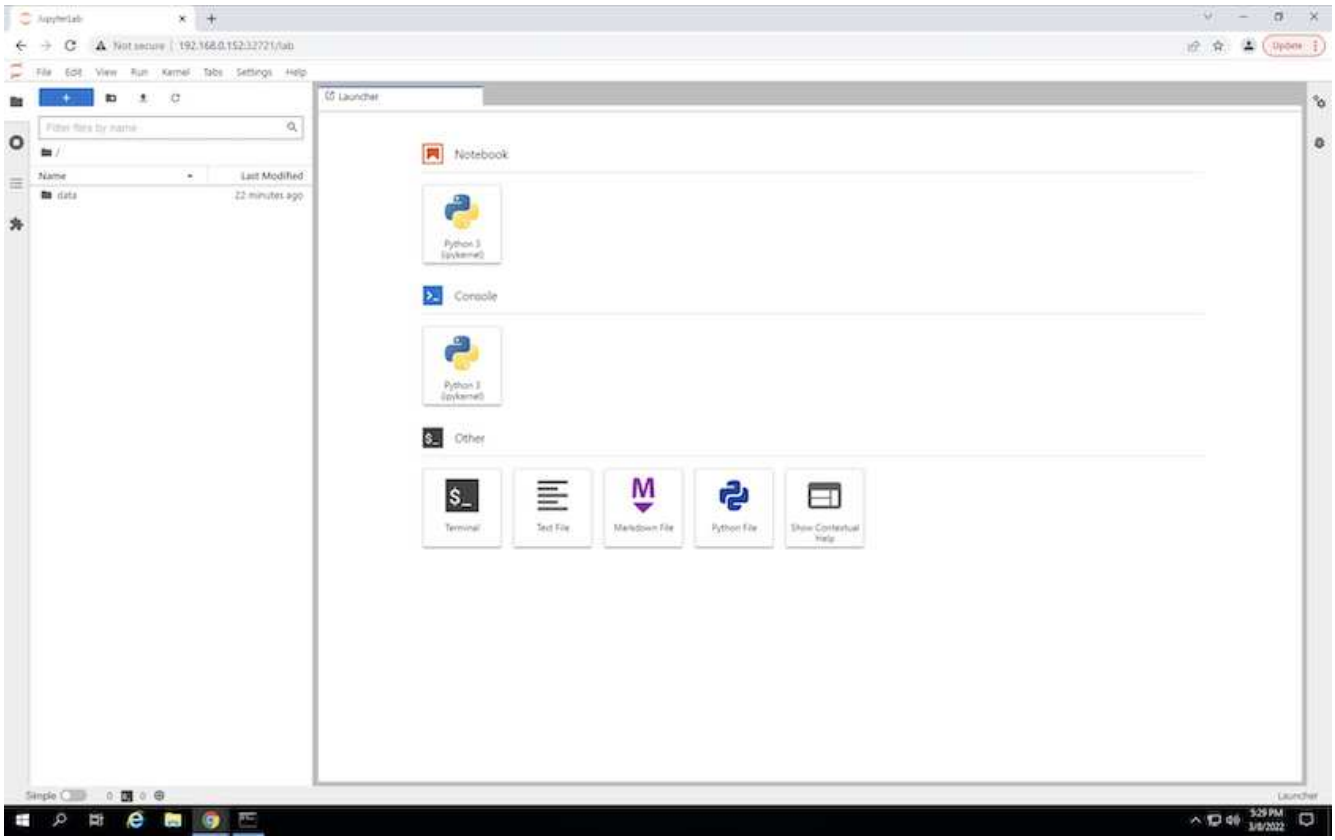
```
$ netapp_dataops_k8s_cli.py create volume --namespace=inference --pvc
-name=inference-data --size=50Gi
Creating PersistentVolumeClaim (PVC) 'inference-data' in namespace
'inference'.
PersistentVolumeClaim (PVC) 'inference-data' created. Waiting for
Kubernetes to bind volume to PVC.
Volume successfully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

3. NetApp DataOpsツールキットを使用して、JupyterLabの新しいワークスペースを作成します。前の手順で作成した永続ボリュームを'--mount-pvcオプションを使用してマウントします必要に応じて'--nvidia -GPU’オプションを使用して’NVIDIA GPUをワークスペースに割り当てます

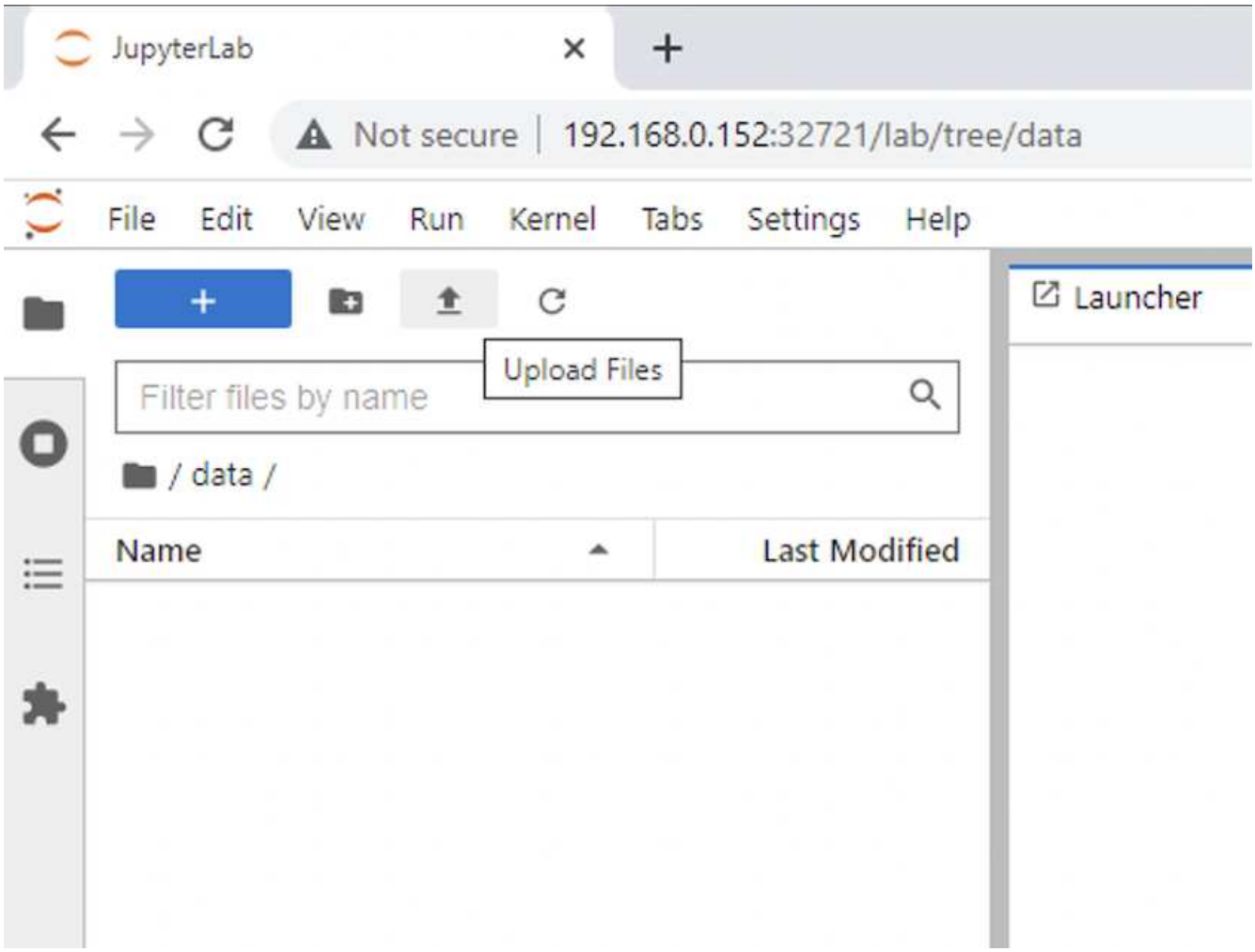
   次の例では’永続ボリュームinerial-data’が'/home/jovyan/data’のJupyterLabワークスペースコンテナにマウントされています公式のProject Jupyterコンテナイメージを使用する場合、「/home/jovyan」はJupyterLab Webインターフェイス内の最上位ディレクトリとして表示されます。

```
$ 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
```

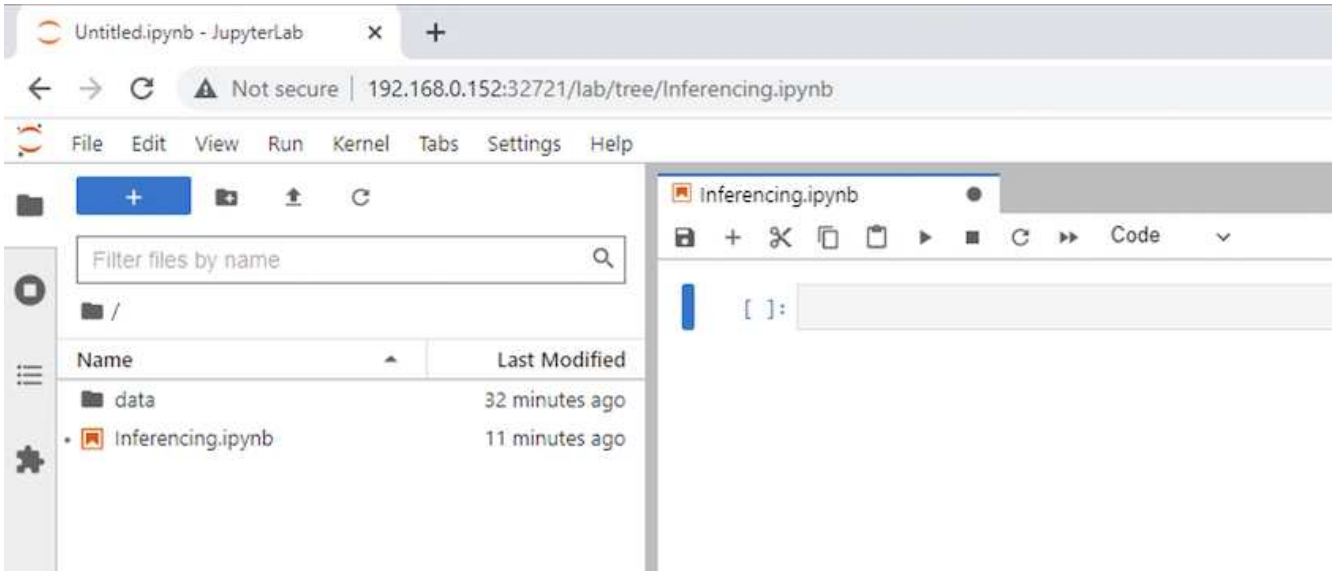
4. 「create jupyterlab」コマンドの出力で指定したURLを使用して、JupyterLabワークスペースにアクセスします。データディレクトリは、ワークスペースにマウントされた永続ボリュームを表します。



5. 「data」ディレクトリを開き、推論を実行するファイルをアップロードします。ファイルがデータディレクトリにアップロードされると、ワークスペースにマウントされた永続ボリュームに自動的に保存されます。ファイルをアップロードするには、次の図に示すように、[ファイルのアップロード]アイコンをクリックします。



6. トップレベルのディレクトリに戻り、新しいノートブックを作成します。



7. ノートブックに推論コードを追加します。次の例は、イメージ検出のユースケースの推論コードを示しています。

Launcher | image-demo-pytorch.ipynb

Code | Python 3 (ipykernel)

STEP 3-1: Clean (Without obfuscation) detection

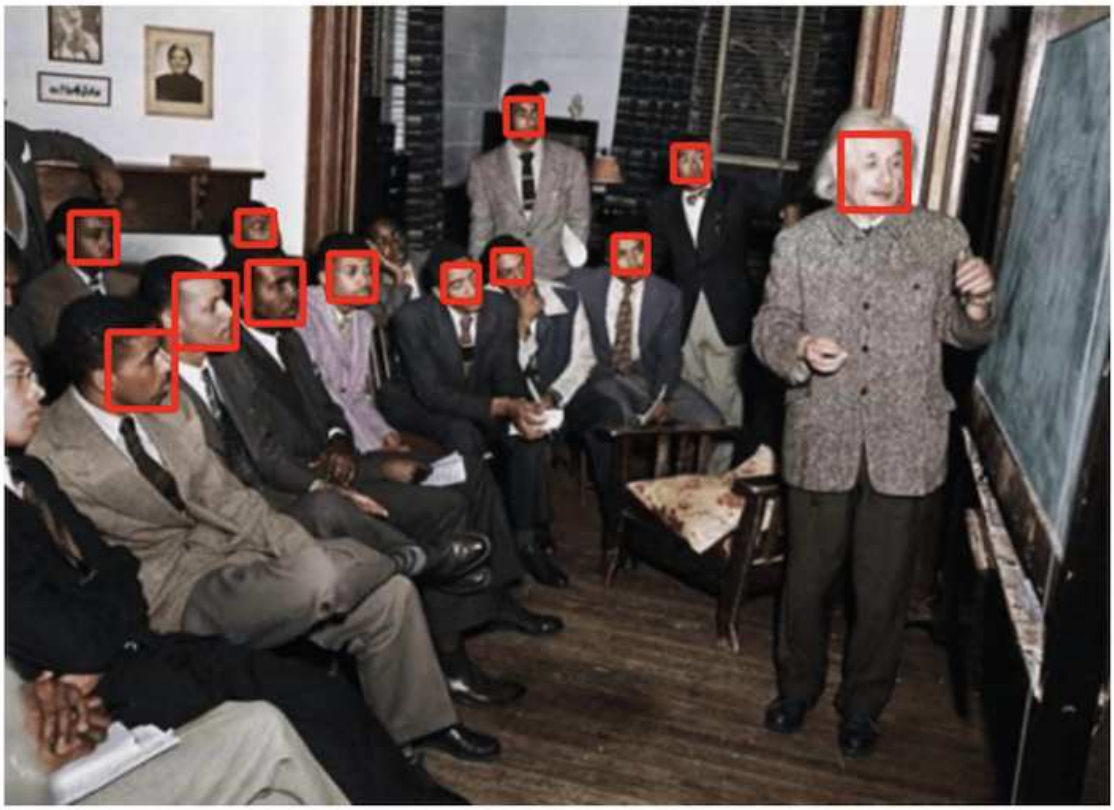
```
[9]: # get current frame
frame = input_image

# 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. 推測コードにProtopia難読化を追加します。Protopiaは、お客様と直接協力してユースケースに固有のドキュメントを提供しますが、本書では取り上げません。次の例は、Protopia難読化を追加した場合のイメージ検出の推論コードを示しています。

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)
```

シナリオ**2–Kubernetes**でのバッチ推論

1. AI / ML推論ワークロード用のKubernetesネームスペースを作成します。

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

2. NetApp DataOpsツールキットを使用して、推論を実行するデータを格納する永続的ボリュームをプロビジョニングします。

```
$ 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. 新しい永続ボリュームに、推論を実行するデータを入力します。

   PVCにデータをロードする方法はいくつかあります。データがNetApp StorageGRID やAmazon S3などのS3互換オブジェクトストレージプラットフォームに現在格納されている場合は、を使用できます"NetApp DataOpsツールキットS3 Data Moverの機能"。また、JupyterLabワークスペースを作成し、JupyterLab Webインターフェイスを使用してファイルをアップロードする方法も簡単です。手順3〜5を参照してくださいシナリオ1–JupyterLabにおけるオンデマンド推論」

4. バッチ推論タスク用のKubernetesジョブを作成します。次の例は、イメージ検出のユースケースに対するバッチ推論ジョブを示しています。このジョブは、一連のイメージ内の各イメージに対して推論を実行し、stdoutに推論の精度の指標を書き込みます。

```
$ 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. 推論ジョブが正常に完了したことを確認します。

```
$ 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. 推測ジョブにProtopia難読化を追加します。Protopiaの難読化を追加する手順は、このテクニカルレポートでは説明していませんが、Protopiaから直接追加できます。次の例は、アルファ値0.8を使用してProtopia難読化を行った場合のフェース検出のバッチ推論ジョブを示しています。このジョブは、一連のイメージ内の各イメージに対して推論を実行する前にProtopia難読化を適用し、stdoutに推論の精度指標を書き込みます。

   このステップは、アルファ値0.05、0.1、0.2、0.4、0.6について繰り返しました。 0.8、0.9、および0.95。結果はに表示されます "「推論の精度比較」"

```
$ 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. 推論ジョブが正常に完了したことを確認します。

```
$ 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
```

シナリオ3–NVIDIA Triton Inference Server

1. AI / ML推論ワークロード用のKubernetesネームスペースを作成します。

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

2. NetApp DataOpsツールキットを使用して、NVIDIA Triton Inference Serverのモデルリポジトリとして使用する永続的ボリュームをプロビジョニングします。

```
$ 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. の新しい永続ボリュームにモデルを保存します "の形式で入力し" これはNVIDIA Triton Inference Serverによって認識されます。

   PVCにデータをロードする方法はいくつかあります。簡単な方法としては、「」の手順3～5で説明しているように、JupyterLabワークスペースを作成し、JupyterLab Webインターフェイスを使用してファイルをアップロードする方法がありますシナリオ1–JupyterLabにおけるオンデマンド推論。」

4. NetApp DataOpsツールキットを使用して、新しいNVIDIA Triton Inference Serverインスタンスを導入します。

```
$ 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. 推論タスクを実行するには、TritonクライアントSDKを使用します。次のPythonコードの抜粋では、Triton PythonクライアントSDKを使用して、フェース検出のユースケースに対する推論タスクを実行しています。この例では、推論のためにTriton APIを呼び出し、イメージを渡します。次に、Triton Inference Serverが要求を受信し、モデルを呼び出して、API結果の一部として推論出力を返します。

```
# 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. 推測コードにProtopia難読化を追加します。Protopia難読化を追加する手順はProtopiaから直接確認できますが、この手順については本テクニカルレポートでは説明していません。次の例は、前述の手順5と同じPythonコードを示していますが、Protopia難読化が追加されています。

Triton APIに渡される前に、Protopia難読化が画像に適用されることに注意してください。このため、難読化されていない画像はローカルマシンから離れることはありません。難読化されたイメージだけがネットワークを通過します。このワークフローは、信頼できるゾーン内でデータが収集され、推論のためにその信頼できるゾーンの外部に渡す必要があるユースケースに該当します。Protopiaの難読化がなければ、機密データが信頼できるゾーンから離れることなく、このタイプのワークフローを実装することはできません。

```
# 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)
```

推論の精度の比較

この検証では、rawイメージのセットを使用して、イメージ検出のユースケースに対して推論を実行しました。次に、同じイメージセットに対して、推論の前にProtopia難読化が追加された同じ推論タスクを実行しました。このタスクでは、Protopia難読化コンポーネントに異なる値のalphaを使用しています。Protopia難読化のコンテキストでは、アルファ値は適用される難読化の量を表し、アルファ値が大きいほど難読化のレベルが高くなります。次に、これらの異なる実行間の推論の精度を比較しました。

以下の2つの表に、ネットアップのユースケースとその概要を示します。

Protopiaは、お客様と直接協力して、特定のユースケースに適したアルファ値を決定します。

コンポーネント	詳細
モデル	フェースボックス(PyTorch)-
データセット	FDDBデータセット

Protopia難読化	アルファ	精度
いいえ	該当なし	0.9337148153739079
はい。	0.05	0.9028766627325002
はい。	0.1	0.9024301009661478
はい。	0.2	0.9081836283186224
はい。	0.4	0.9073066107482036
はい。	0.6	0.8847816568680239
はい。	0.8	0.8841195749171925
はい。	0.9	0.8455427675252052
はい。	0.95	0.8455427675252052

難読化の速度

この検証では、Protopia難読化を1920 x 1080ピクセル画像に5回適用し、難読化手順が完了するまでに毎回かかった時間を測定しました。

単一のNVIDIA V100 GPUで動作するPyTorchを使用して難読化を適用し、実行間のGPUキャッシュをクリアしました。難読化手順では、5回の実行でそれぞれ5.47ミリ秒、5.27ミリ秒、4.54ミリ秒、5.24ミリ秒、および4.84ミリ秒が実行されました。平均速度は5.072msでした。

まとめ

データは、保管中、転送中、コンピューティング中の3つの状態に存在します。AI推論サービスの重要な部分は、プロセス全体における脅威からのデータの保護である必要があります。推論の実行中にデータを保護することが重要です。これは、外部のお客様と推論サービスを提供するビジネスの両方の個人情報がこのプロセスによって公開される可能性があるためです。Protopia AIは、今日の市場における機密性の高いAI推論を行うための、悪意のないソフトウェアのみの解決策 です。Protopiaを使用すると、AIには、AI / MLタスクを実行するために必要なデータレコードの変換された情報のみが渡されます。これ以外にも、データレコードには何も入力されません。この確率的変化はマスキングの形式ではなく、キュレーションされたノイズを使用してデータの表現を数学的に変更することに基づいています。

ONTAP 機能を備えたネットアップのストレージシステムは、ローカルSSDストレージと同等以上のパフォーマンスを提供します。また、NetApp DataOpsツールキットと組み合わせることで、データサイエンティスト、データエンジニア、AI / ML開発者、ビジネスまたはエンタープライズITの意思決定者に次のようなメリットをもたらします。

- AI システム、分析などの重要なビジネスシステム間でデータを容易に共有できます。このようなデータ共有により、インフラのオーバーヘッドを削減し、パフォーマンスを向上させ、企業全体のデータ管理を合理化できます。
- 個別に拡張可能なコンピューティングとストレージにより、コストを最小限に抑え、リソース使用率を向上させます。
- 統合された Snapshot コピーとクローンを使用して開発と導入のワークフローを合理化し、ユーザのワークスペースを瞬時にスペース効率よく利用できるほか、バージョン管理機能も統合され、導入も自動化されています。
- ディザスタリカバリ、ビジネス継続性、規制要件に対応するエンタープライズクラスのデータ保護とデータガバナンス。
- データ管理操作の簡単な呼び出し。データサイエンティストのワークスペースのSnapshotコピーを迅速に作成し、Jupyterノートブック内のNetApp DataOpsツールキットからバックアップとトレーサビリティを実現します。

ネットアップとProtopia解決策 は、エンタープライズクラスのAI推論環境に最適な、柔軟性に優れたスケールアウトアーキテクチャを提供します。データ保護を実現し、オンプレミス環境とハイブリッドクラウド環境の両方で、AI推論の機密要件を満たすことができる機密情報のプライバシーを提供します。

追加情報と謝辞の参照先

このドキュメントに記載されている情報の詳細については、以下のドキュメントや Web サイトを参照してください。

- NetApp ONTAP データ管理ソフトウェア—ONTAP 情報ライブラリ

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

- コンテナ向け NetApp 永続的ストレージ— NetApp Trident

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

- NetApp DataOps ツールキット

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

- コンテナ向けNetApp Persistent Storage—NetApp Astra Trident

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

- Protopia AI—Confidential Inference（機密情報推論）

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

- NetApp BlueXPのコピーと同期

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

- NVIDIA Triton 推論サーバ

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

- NVIDIA Triton Inference Serverのドキュメント

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

- PyTorchのフェイスボックス

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

謝辞

- ネットアップ、プリンシパルプロダクトマネージャー、Mark Cates氏
- ネットアップ、テクニカルマーケティングエンジニア、Sufian Ahmad氏
- 最高技術責任者兼Protopia AI教授、Hadi Esmaeilzadeh氏

ネットアップの **AI** による地合い分析

TR-4910 ：『 **NetApp AI** と顧客コミュニケーションを組み合わせた感情分析』

Sathish Thyagarajan 、 Rick Huang 氏、および SFL Scientific 、 Diego Sosa-coba 、 David Arnette 氏

このテクニカルレポートでは、転送学習と会話型 AI を使用して、ネットアップのデータ管理テクノロジと NVIDIA ソフトウェアフレームワークを使用して、エンタープライズレベルのグローバルサポートセンターで感情分析を行うための設計ガイダンスを提供します。この解決策は、チャットログ、 E メール、およびその他のテキストまたは音声通信を表す録音された音声ファイルやテキストファイルから顧客の洞察を得たいと考えているあらゆる業界に適用されます。ネットアップはエンドツーエンドのパイプラインを実装して、ネットアップのクラウド対応オールフラッシュストレージを使用した GPU アクセラレーションコンピューティングクラスタで、自動音声認識、リアルタイムの感情分析、ディープラーニングの自然言語処理モデル再トレーニング機能をデモンストレーションしました。大規模で最先端の言語モデルのトレーニングと最適化により、世界規模のサポートセンターで推論を迅速に実行できるようになり、優れたカスタマーエクスペリエンスと目標を達成し、長期的な従業員パフォーマンス評価を実施できます。

感情分析は、正、負、または中性感情がテキストから抽出される Natural Language Processing （ NLP ）内の研究分野です。会話型 AI システムは、より多くの人がコミュニケーションを行うようになったため、ほぼグローバルレベルの統合にまで成長しました。感情分析には、サポートセンターの従業員のパフォーマンスを発信者との会話で決定し、適切な自動チャットボット応答を提供し、四半期ごとの収益呼における企業の代表者と対象者間のやり取りに基づいて会社の株価を予測するなど、さまざまなユースケースがあります。さらに、感情分析を使用して、ブランドが提供する製品、サービス、サポートに関するお客様の見解を判断できます。

このエンドツーエンドの解決策は、 NLP モデルを使用して、サポートセンター分析フレームワークを可能にする高度なセンチメント分析を実行します。音声録音は文書化されたテキストに処理され、会話の各文から感情が抽出されます。結果はダッシュボードに集約され、会話の感情を分析するために、従来とリアルタイムの両方で巧妙に細工することができます。この解決策は、データモダリティと出力ニーズが似ている他のソリューションに汎用化できます。適切なデータを使用することで、他のユースケースにも対応できます。たとえば、企業収益の問い合わせを、同じエンドツーエンドパイプラインを使用して、センチメントについて分析することができます。また、パイプラインの柔軟性が高いため、トピックモデリングや Named Entity Recognition （ NER ）などの他の形式の NLP 解析も可能です。

これらの AI 実装は、 NVIDIA Rivea 、 NVIDIA TAO Toolkit 、 NetApp DataOps ツールキットが連携して実現しました。NVIDIA のツールを使用すると、あらかじめ組み込まれたモデルとパイプラインを使用して、ハイパフォーマンスな AI ソリューションを迅速に導入できます。NetApp DataOps ツールキットにより、さまざまなデータ管理タスクが簡易化され、開発期間が短縮されます。

お客様にもたらされる価値

企業は、感情分析のためのテキスト、音声、ビデオの会話について、従業員評価および顧客対応ツールから価値を得ています。マネージャーは、ダッシュボードに表示される情報を活用して、会話の両側に基づいて従業員と顧客満足度を評価できます。

さらに、 NetApp DataOps ツールキットは、お客様のインフラストラクチャ内でのデータのバージョン管理と割り当てを管理します。その結果、ダッシュボードに表示される分析情報が頻繁に更新されるため、データストレージのコストを抑えることができません。

ユースケース

これらのサポートセンターで処理されるコールの数が原因で、手動で実行した場合はコールパフォーマンスの評価にかなりの時間がかかる可能性があります。BAG of Words カウンティングなどの従来のメソッドは、いくつかの自動化を実現できますが、これらのメソッドは、ダイナミック言語の微妙な側面や意味をキャプチャしません。AI モデリング手法を使用すると、このように詳細な分析を自動化された方法で実行できます。さらに、 NVIDIA 、 AWS 、 Google などが公開している最新のトレーニング済みモデリングツールを使用することで、複雑なモデルを含むエンドツーエンドのパイプラインを容易に構築し、カスタマイズできるようになりました。

サポートセンターの感情分析のためのエンドツーエンドのパイプラインは、従業員が発信者と会話するときに、音声ファイルをリアルタイムで取り込みます。次に、これらのオーディオファイルは音声テキストコンポーネントで使用するために処理され、テキスト形式に変換されます。会話中の各文は、感情（肯定的、否定的、または中立的）を示すラベルを受け取ります。

感情分析は、コールパフォーマンスを評価するための会話の重要な側面を提供することができます。これらの感情は、従業員と発信者間のやり取りにさらに深いレベルを追加します。AI を活用した感情ダッシュボードは、マネージャーが会話内の感情をリアルタイムに追跡し、従業員の過去の問い合わせを過去に分析します。

この問題を解決するエンドツーエンドの AI パイプラインを迅速に構築するための強力な方法が用意されています。この場合、 NVIDIA Riva ライブラリを使用して、音声変換と感情分析の 2 つの直列タスクを実行できます。1 つ目は教師あり学習信号処理アルゴリズムで、 2 つ目は教師あり学習 NLP 分類アルゴリズムです。NVIDIA TAO Toolkit を使用すれば、ビジネス関連のデータを使用して、関連するあらゆるユースケースに合わせてアルゴリズムを微調整できます。その結果、コストとリソースの数分の 1 に過ぎず、より正確で強力なソリューションを構築できます。お客様はを組み込むことができます "NVIDIA Maxine の 2 つのポートが" サポートセンター設計における GPU アクセラレーションビデオ会議アプリケーションのフレームワーク。

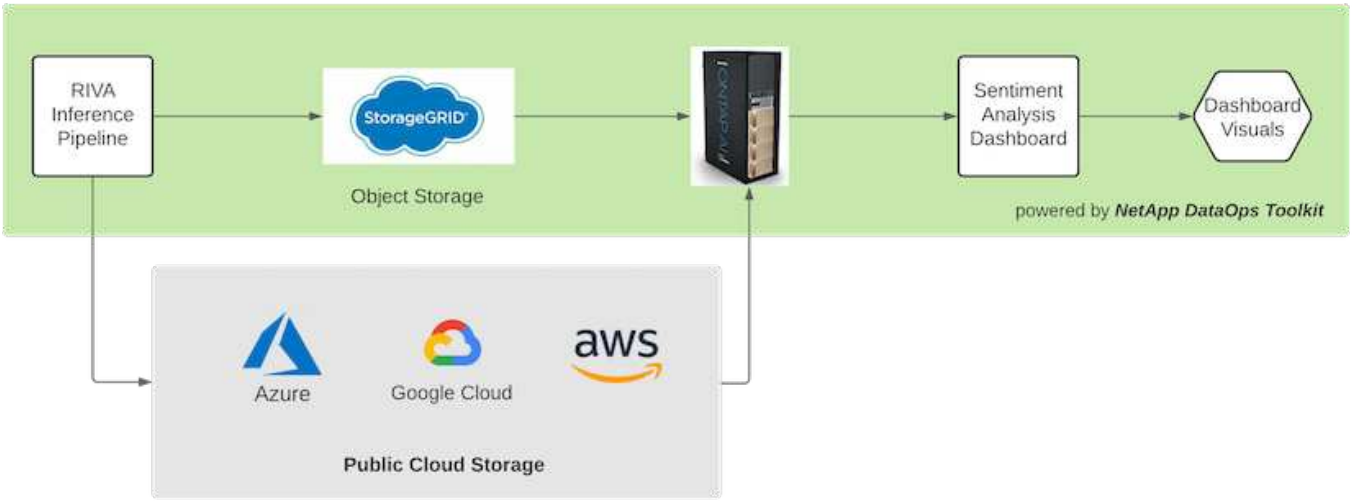
この解決策の中核をなすのは、次のユースケースです。どちらのユースケースでも、 TAIO ツールキットを使用してモデルの微調整を行い、 Rivea を使用してモデルを展開します。

- 音声テキスト
- 感情分析

従業員と顧客の間のサポートセンターのやり取りを分析するために、音声コールの形式で各顧客との会話をパ

イプラインを通じて実行し、文レベルの感情を抽出できます。そのような感情は、人間が感情を正当化するか、必要に応じて調整することができます。次に、ラベル付けされたデータが微調整ステップに渡され、感情の予測が改善されます。ラベル付きの感情データがすでに存在する場合は、モデルの微調整を迅速に行うことができます。どちらの場合も、パイプラインは音声の取り込みと文章の分類を必要とする他のソリューションに対して一般化可能です。

AI の感情に関するアウトプットは、外部クラウドデータベースまたは企業が管理するストレージシステムにアップロードされます。この大規模なデータベースからローカルストレージにセンチメント出力が転送され ' 管理者のセンチメント分析を表示するダッシュボード内で使用されますダッシュボードの主な機能は、カスタマーサービスのスタッフとリアルタイムで連携することです。マネージャは、コール中の従業員に関する評価やフィードバックを行い、各文章の感情を最新の状態に更新したり、従業員の過去のパフォーマンスや顧客からの反応を履歴的に確認したりすることができます。



。 "NetApp DataOps ツールキット" Riva 推論パイプラインが感情ラベルを生成した後も、データストレージ

システムの管理を継続できます。これらの AI 分析結果は、 NetApp DataOps ツールキットで管理するデータストレージシステムにアップロードできます。データストレージシステムは、数百ものインサートを管理し、毎分選択できる能力を備えている必要があります。ローカルデバイスストレージシステムは、大容量のデータストレージをリアルタイムで照会して抽出します。大規模なデータストレージインスタンスを照会して履歴データを照会することで、ダッシュボードのエクスペリエンスを強化することもできます。NetApp DataOps ツールキットを使用すると、データを迅速にクローニングし、データを使用するすべてのダッシュボードにデータを分散できるため、この 2 つの方法を簡単に使用できます。

対象読者

解決策の対象となるグループは次のとおりです。

- 従業員のマネージャー
- データエンジニア / データサイエンティスト
- IT 管理者（オンプレミス、クラウド、ハイブリッド）

会話中に感情を追跡することは、従業員のパフォーマンスを評価するための貴重なツールです。AI ダッシュボードを使用することで、マネージャーは従業員と発信者が自分の感情をリアルタイムでどのように変化させるかを確認できるため、ライブ評価やガイダンスセッションが可能になります。さらに、音声会話、テキストチャットボット、ビデオ会議に参加しているお客様から、価値ある顧客インサイトを得ることができます。このような顧客分析では、最新の AI モデルとワークフローを使用して、大規模なマルチモーダル処理の機能を活用しています。

データ側では、多数のオーディオファイルがサポートセンターによって毎日処理されます。NetApp DataOps ツールキットを使用すると、モデルの定期的な微調整と感情分析用ダッシュボードの両方で、このデータ処理タスクを容易に行うことができます。

IT 管理者は、 NetApp DataOps ツールキットを利用して、導入環境と本番環境の間でデータを迅速に移動することもできます。また、リアルタイム推論のためには、 NVIDIA 環境とサーバも管理、分散する必要があります。

アーキテクチャ

このサポートセンターの解決策のアーキテクチャは、 NVIDIA が構築したツールと NetApp DataOps ツールキットを中心にしています。NVIDIA のツールを使用すると、構築済みのモデルとパイプラインを使用して、ハイパフォーマンスな AI ソリューションを迅速に導入できます。NetApp DataOps ツールキットにより、さまざまなデータ管理タスクが簡易化され、開発期間が短縮されます。

解決策テクノロジ

"NVIDIA RIVA" GPU でリアルタイムのパフォーマンスを実現する、マルチモーダルな会話型 AI アプリケーションを構築するための GPU アクセラレーション対応 SDK です。NVIDIA Train 、 Adapt 、 Optimize （ TAO ）ツールキットは、トレーニングを高速化し、高精度で高性能なドメイン固有の AI モデルをすばやく簡単に作成する方法を提供します。

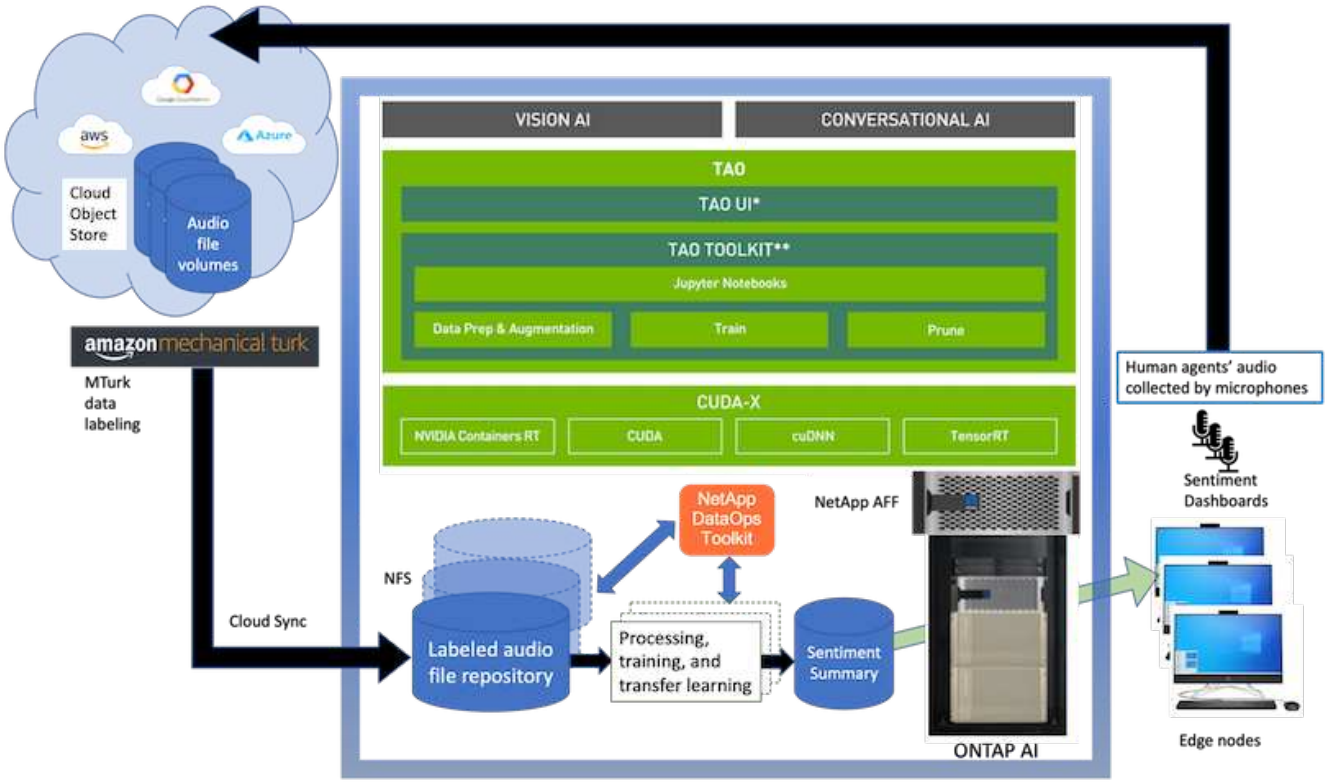
NetApp DataOps ツールキットは Python ライブラリで、開発者、データサイエンティスト、 DevOps エンジニア、データエンジニアはさまざまなデータ管理タスクを簡単に実行できます。これには、新しいデータボリュームまたは JupyterLab ワークスペースのほぼ瞬時のプロビジョニング、データボリュームまたは JupyterLab ワークスペースのほぼ瞬時のクローニング、データボリュームまたは JupyterLab ワークスペースのほぼ瞬時の Snapshot コピーによるトレーサビリティとベースライン設定が含まれます。

アーキテクチャ図

次の図は、解決策のアーキテクチャを示しています。環境には、クラウド、コア、エッジの 3 つのカテゴリがあります。各カテゴリは地理的に分散させることができます。たとえば、クラウドにはさまざまなリージョンのバケットにオーディオファイルが格納されたオブジェクトストアが含まれていますが、コアには高速ネットワークやNetApp BlueXPのコピーと同期でリンクされたデータセンターが含まれている場合があります。エッジノードは、ヒューマンエージェントの日常的な作業プラットフォームを表しています。このプラットフォームでは、対話型ダッシュボードツールとマイクを使用して感情を視覚化したり、顧客との会話から音声データを収集したりできます。

GPU によって高速化されたデータセンターでは、 NVIDIA を使用できます "リバ" 会話型 AI アプリケーションを構築するためのフレームワーク。それには、があります "Tao ツールキット" Transfer L ラーニング技術を使用して、モデルのフィニッチニングと再トレーニングを接続します。これらのコンピューティングアプリケーションとワークフローは、を基盤としています "NetApp DataOps ツールキット"ONTAP が提供する最高のデータ管理機能を実現します。このツールキットを使用すると、企業のデータチームは、スナップショットやクローンを使用して、構造化データと非構造化データでモデルのプロトタイプを迅速に作成できるため、トレーサビリティ、バージョン管理、 A/B テストを実現し、セキュリティ、ガバナンス、 コンプライアンスを実現できます。を参照してください "ストレージ設計" 詳細：

この解決策では、オーディオファイル処理、 NLP モデルトレーニング、トランスファーラーニング、およびデータ管理の詳細な手順について説明します。最終的なパイプラインが生成され、ヒューマンサポートエージェントのダッシュボードにリアルタイムで表示されるセンチメントの概要が生成されます。



ハードウェア要件

次の表に、解決策の実装に必要なハードウェアコンポーネントを示します。解決策の特定の実装で使用されるハードウェアコンポーネントは、お客様の要件に応じて異なる場合があります。

応答遅延テスト	時間（ミリ秒）
データ処理	10.
推論	10.

この応答時間テストは、 560 の会話で 50,000 以上のオーディオファイルで実行されました。各オーディオファイルのサイズは、 MP3 の場合は約 100 KB 、 WAV の場合は約 1 MB でした。データ処理手順では、 MP3 を WAV ファイルに変換します。推論の手順では、オーディオファイルをテキストに変換し、テキストから感情を抽出します。これらのステップは互いに独立しており、並列化することでプロセスを高速化できます。

ストア間でのデータ転送の遅延を考慮すると、マネージャは、文章の最後の 2 番目の時間内にリアルタイムの感情分析の更新を確認できるようになります。

NVIDIA Riva ハードウェア

ハードウェア	要件
OS	Linux x86_64
GPU メモリ（ASR）	ストリーミングモデル：最大 5600 MB の非ストリーミングモデル：約 3100 MB
GPU メモリ（NLP）	1 つの BERT モデルで最大 500MB

NVIDIA TAO ツールキットハードウェア

ハードウェア	要件
システム RAM	32 GB
GPU RAM	32 GB
CPU	8 コア
GPU	NVIDIA （A100 、 V100 、 RTX 30x0）
SSD の場合	100GB

フラッシュストレージシステム

NetApp ONTAP 9.

ネットアップの最新世代のストレージ管理ソフトウェア ONTAP 9.9 は、インフラの刷新とクラウド対応データセンターへの移行を可能にします。ONTAP は、業界をリードするデータ管理機能を活用して、データの格納場所に関係なく、単一のツールセットでデータの管理と保護を実現します。エッジ、コア、クラウドなど、必要な場所に自由にデータを移動することもできます。ONTAP 9.9 には、データ管理を簡素化し、重要なデータを高速化および保護し、ハイブリッドクラウドアーキテクチャ全体で次世代のインフラ機能を実現する、多数の機能が含まれています。

NetApp BlueXPのコピーと同期

"BlueXPのコピーと同期" は、高速でセキュアなデータ同期を実現するネットアップのサービスです。オンプレミスの NFS または SMB ファイル共有間で、次のいずれかのターゲットにファイルを転送できます。

- NetApp StorageGRID

- NetApp ONTAP S3
- NetApp Cloud Volumes Service の略
- Azure NetApp Files の特長
- Amazon Simple Storage Service （ Amazon S3 ）
- Amazon Elastic File System （ Amazon EFS ）
- Azure Blob の略
- Google クラウドストレージ
- IBM クラウドオブジェクトストレージ

BlueXPのCopy and Syncは、必要な場所に迅速かつ安全にファイルを移動します。転送されたデータは、ソースとターゲットの両方で完全に使用できます。BlueXPのCopy and Syncは、事前定義されたスケジュールに基づいてデータを継続的に同期し、差分のみを移動するため、データレプリケーションにかかる時間とコストを最小限に抑えることができます。BlueXPのCopy and Syncは、セットアップと使用が簡単なソフトウェアサービス（SaaS）ツールです。BlueXPのCopyとSyncによってトリガーされるデータ転送は、データブローカーによって実行されます。BlueXPのCopy and Syncデータブローカーは、AWS、Azure、Google Cloud Platform、オンプレミスに導入できます。

NetApp StorageGRID

StorageGRID の Software-Defined オブジェクトストレージスイートは、パブリッククラウド、プライベートクラウド、ハイブリッドマルチクラウド環境のすべてをシームレスにサポートし、幅広いユースケースに対応しています。業界をリードするイノベーションにより、 NetApp StorageGRID は、非構造化データを長期にわたって自動化されたライフサイクル管理などの多目的に保管、保護、保管します。詳細については、を参照してください "NetApp StorageGRID" サイト

ソフトウェア要件

次の表に、この解決策を実装するために必要なソフトウェアコンポーネントを示します。解決策の特定の実装で使用されるソフトウェアコンポーネントは、お客様の要件に応じて異なる場合があります。

ホストマシン	要件
Riva (以前の開発コード名 Jarv)	1.4.0
Tao ツールキット (以前の Transfer Learning Toolkit)	3.0
ONTAP	9.9.1
DGX OS	5.1
DTK	2.0.0

NVIDIA Riva ソフトウェア

ソフトウェア	要件
Docker です	>19.02 （ NVIDIA - Docker をインストール済み） >=19.03 （ DGX を使用していない場合
NVIDIA ドライバ	465.19.01 + 418.40 + 、 440.33 + 、 450.51 + 、 460.27 + （データセンターの GPU の場合

ソフトウェア	要件
コンテナ OS	Ubuntu 20.04
CUDA （ CUDA	11.3.0
cuBLAS	11.5.1.101
cuDNN	8.2.0.41
NCCL	2.9.6
TensorRT	7.2.3.4.
Triton Inference サーバ	2.9.0

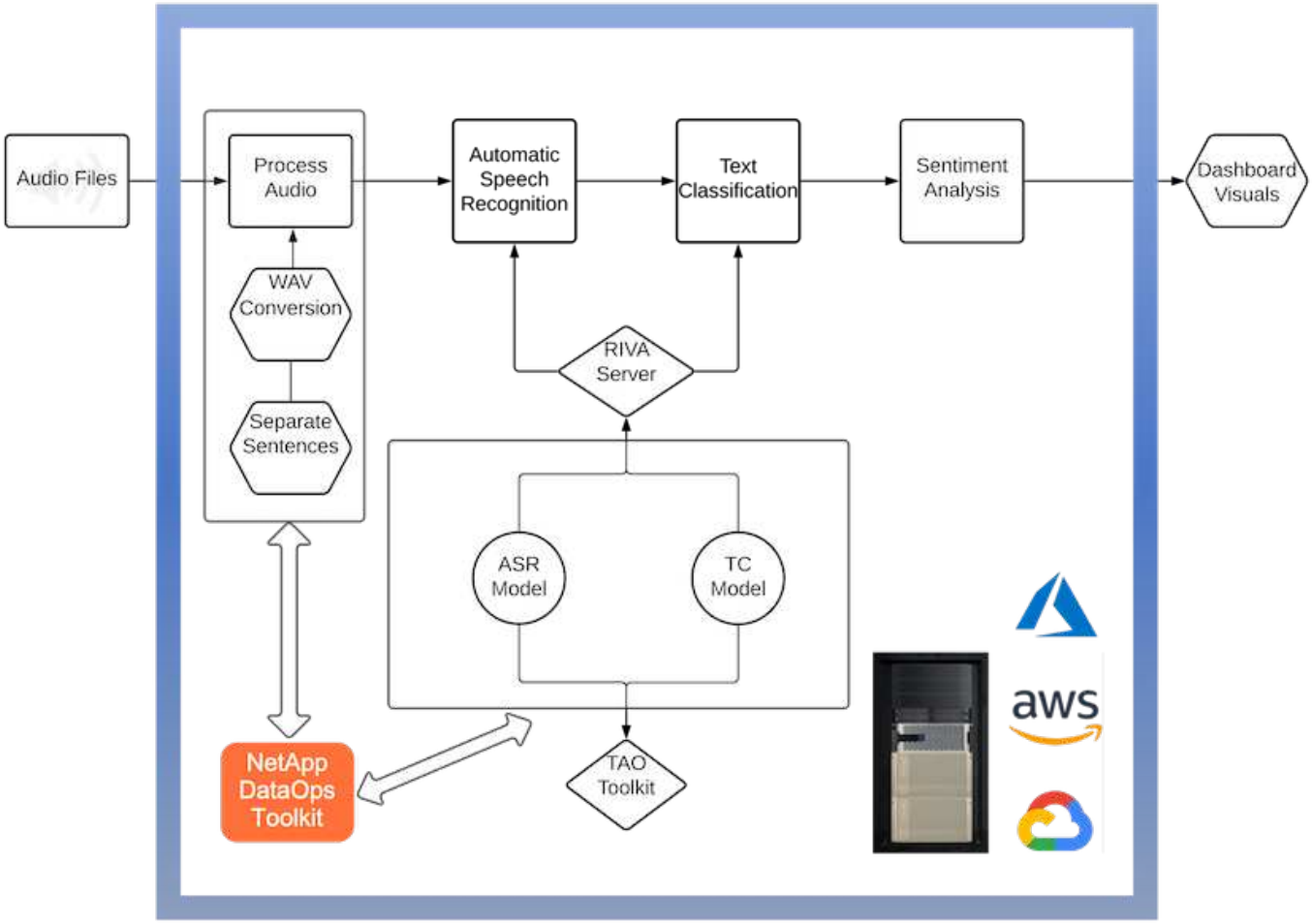
NVIDIA TAO ツールキットソフトウェア

ソフトウェア	要件
Ubuntu 18.04 LTS	18.04
Python	3.6.9 以上
Docker - CE	19.03.5
Docker - API	1.40
nvidia -container-toolkit	>1.3.0-1
nvidia Container - ランタイム	3.4.0 -1
nvidia - docker2	2.5.0-1
nVidia ドライバ	> 455
python-pip	>21.06
nvidia -pyindex	最新バージョン

ユースケースの詳細

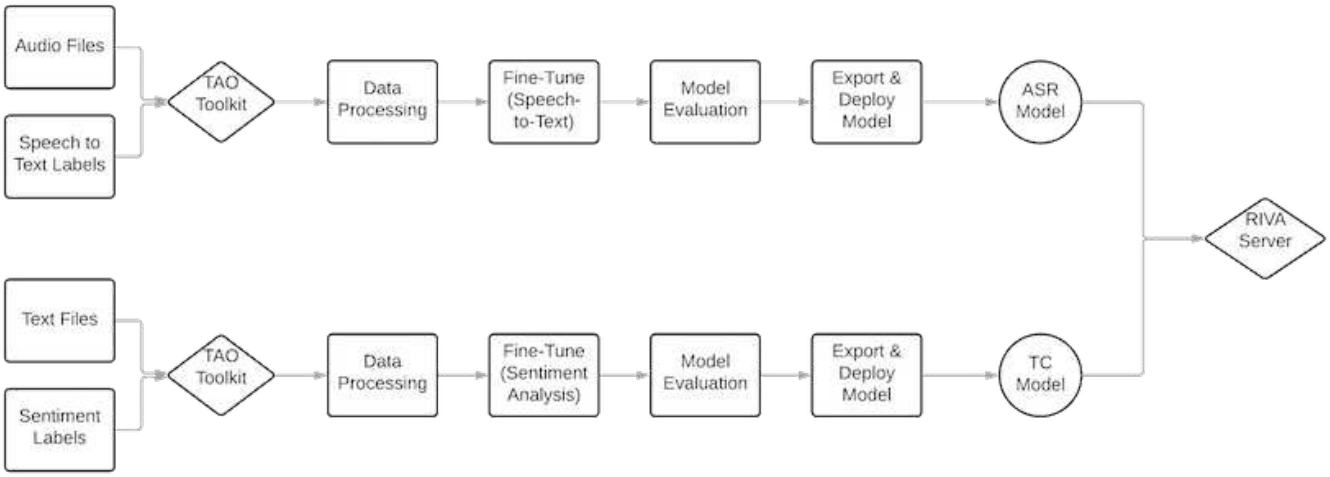
この解決策環境のユースケースは次のとおりです。

- 音声テキスト
- 感情分析



音声テキスト変換のユースケースは、まずサポートセンターの音声ファイルを取り込むことから始まります。このオーディオは、 Riva が必要とする構造に合わせて処理されます。オーディオファイルが解析単位に分割されていない場合は、オーディオを Riva に渡す前にこれを行う必要があります。オーディオファイルが処理されると、 API 呼び出しとして Riva サーバーに渡されます。サーバは、ホスティングしている多くのモデルの 1 つを採用し、応答を返します。この音声 / テキスト（自動音声認識の一部）は、音声のテキスト表現を返します。そこから、パイプラインはセンチメント分析部分に切り替わります。

感情分析では、自動音声認識からのテキスト出力がテキスト分類への入力として機能します。Text Classification は、任意の数のカテゴリにテキストを分類するための NVIDIA コンポーネントです。サポートセンターとの会話では、感情のカテゴリがプラスからマイナスになります。モデルのパフォーマンスは、ホールドアウトセットを使用して、微調整ステップの成功を判断することができます。



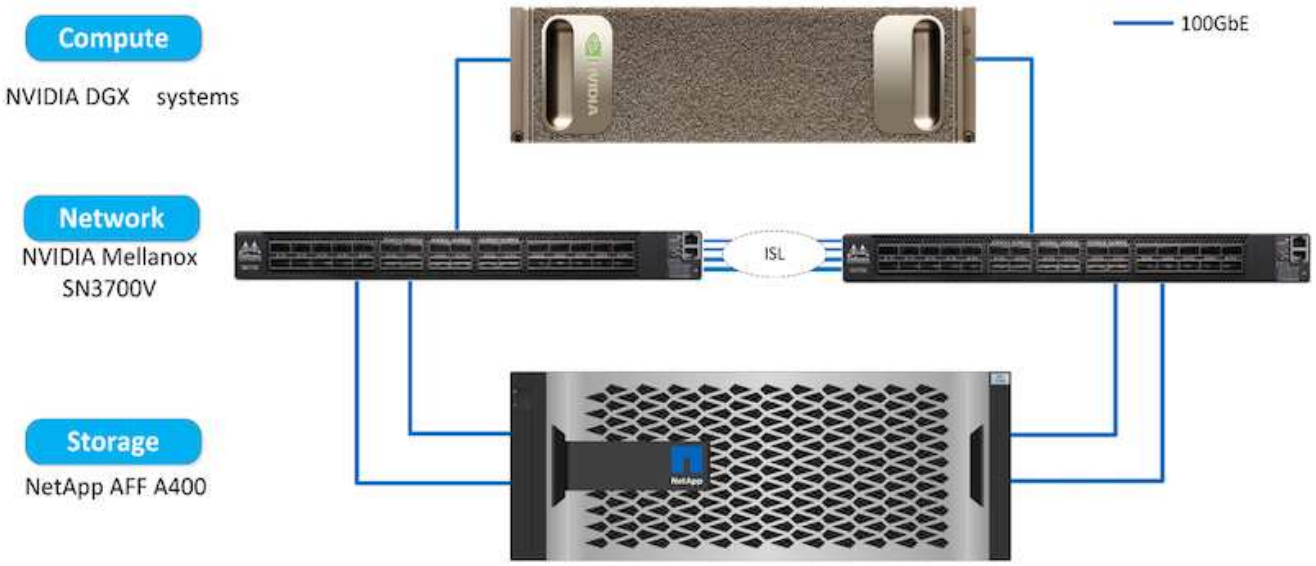
TAO ツールキット内の音声テキスト分析と感情分析にも、同様のパイプラインが使用されています。主な違いは、モデルの微調整に必要なラベルの使用です。TAO ツールキットパイプラインは、データファイルの処理から始まります。次に、事前にトレーニングされたモデル（から入手可能 "NVIDIA NGC カタログ"）は、サポートセンターのデータを使用して微調整されます。微調整されたモデルは、対応するパフォーマンス指標に基づいて評価され、事前トレーニングされたモデルよりもパフォーマンスが高い場合は、 Riva サーバに導入されます。

設計上の考慮事項

このセクションでは、この解決策のさまざまなコンポーネントの設計上の考慮事項について説明します。

ネットワークとコンピューティングの設計

データセキュリティの制限に応じて、すべてのデータはお客様のインフラストラクチャまたはセキュアな環境内に保持されている必要があります。



ストレージ設計

NetApp DataOps ツールキットは、ストレージシステムを管理するための主要なサービスです。DataOps ツールキットは Python ライブラリで、開発者、データサイエンティスト、 DevOps エンジニア、データエンジニアは、新しいデータボリュームや JupyterLab ワークスペースのほぼ瞬時のプロビジョニング、データボリュームや JupyterLab ワークスペースのほぼ瞬時のクローニングなど、さまざまなデータ管理タスクを簡単に実行できます。 トレーサビリティやベースライン設定のためのデータボリュームまたは JupyterLab ワークスペースのほぼ瞬時のスナップショット作成。この Python ライブラリは、任意の Python プログラムまたは Jupyter Notebook にインポートできるコマンドラインユーティリティまたは関数ライブラリとして機能します。

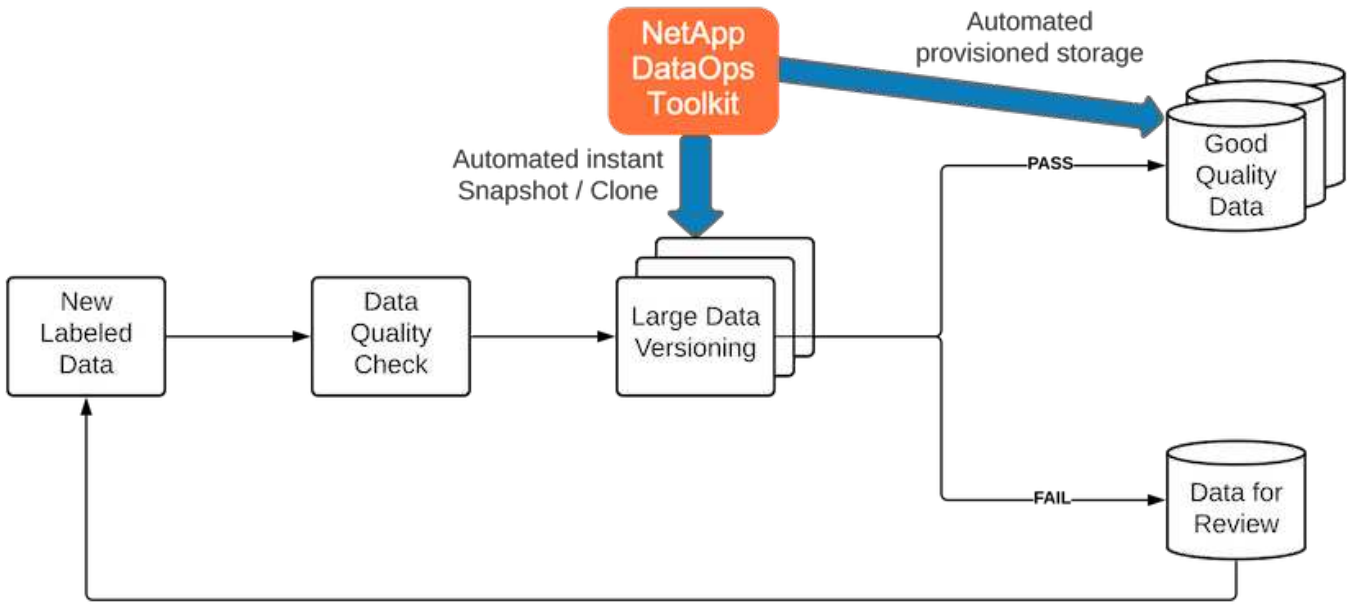
RIVA のベストプラクティス

NVIDIA はいくつかの一般的な機能を提供 "ベストプラクティスに基づくデータ保護" リベットを使用する場合：

- * 可能であれば、ロスレスのオーディオフォーマットを使用します。 * MP3 などの損失のあるコーデックを使用すると、品質が低下する可能性があります。
- * トレーニングデータの増加。 * 音声トレーニングデータにバックグラウンドノイズを追加することで、当初は精度を低下させながら堅牢性を高めることができます。
- * スクラップテキストを使用すれば語彙のサイズを制限しなさい。 * 多くのオンライン源にタイプミスまたは補助発音および珍しい単語を含んでいる。これらを削除すると、言語モデルが改善されます。
- * 可能であれば、最小サンプリングレート 16kHz を使用します。 * ただし、オーディオ品質が低下するため、リサンプルしないようにしてください。

これらのベストプラクティスに加えて、パイプラインの各ステップで正確なラベルを持つ代表的なサンプルデータセットの収集に優先順位を付ける必要があります。つまり、サンプルデータセットには、ターゲットデータセットに典型的な指定された特性を比例的に反映させる必要があります。同様に、データセットの注釈には、データの品質と量を最大化するために、正確性とラベル付けの速度のバランスをとる責任があります。たとえば、このサポートセンターの解決策には、音声ファイル、ラベル付きテキスト、および感情ラベルが必要です。この解決策は、シーケンシャルなので、パイプラインの開始時に発生したエラーが最後まで伝播されます音声ファイルの品質が悪い場合は、テキスト文字変換と感情ラベルも同様になります。

このエラーの伝播も同様に、環境 the models Trained on this data です。感情の予測が 100% 正確であるにもかかわらず、音声テキスト変換モデルのパフォーマンスが低い場合、最終的なパイプラインは最初の音声テキスト変換によって制限されます。開発者は、各モデルのパフォーマンスを個別に、また大きなパイプラインのコンポーネントとして考慮する必要があります。この場合、最終目標は、感情を正確に予測できるパイプラインを開発することです。そのため、パイプラインを評価する全体的な指標は感情の精度であり、音声からテキストへの変換は直接影響を与えます。



NetApp DataOps ツールキットは、ほぼ瞬時のデータクローニングテクノロジを使用して、データ品質チェックパイプラインを補完します。各ラベル付きファイルを評価し、既存のラベル付きファイルと比較する必要があります。これらの品質チェックをさまざまなデータストレージシステムに分散させることで、これらのチェックを迅速かつ効率的に実行できます。

サポートセンターのセンチメント分析の導入

解決策の導入には、次のコンポーネントが含まれます。

1. NetApp DataOps ツールキット
2. NGC の設定
3. NVIDIA Rivea サーバ
4. NVIDIA TAO ツールキット
5. TAO モデルを Riva にエクスポートします

導入を実行するには、次の手順を実行します。

NetApp DataOps ツールキット：センターのセンチメント分析をサポート

を使用します "NetApp DataOps ツールキット"、次の手順を実行します。

1. PIP でツールキットをインストールします。

```
python3 -m pip install netapp-dataops-traditional
```

2. データ管理を設定

```
netapp_dataops_cli.py config
```

NGC 構成：センターの感情分析をサポート

セットアップするには "NVIDIA NGC"、次の手順を実行します。

1. NGC をダウンロード

```
wget -O ngccli_linux.zip
https://ngc.nvidia.com/downloads/ngccli_linux.zip && unzip -o
ngccli_linux.zip && chmod u+x ngc
```

2. 現在のディレクトリをパスに追加します。

```
echo "export PATH=\"\$PATH:$(pwd)\"" >> ~/.bash_profile && source
~/.bash_profile
```

3. コマンドを実行できるように、 NGC CLI を設定する必要があります。次のコマンドを入力します。プロンプトが表示されたら、 API キーも入力します。

```
ngc config set
```

Linux ベースではないオペレーティングシステムについては、を参照してください "こちらをご覧ください"。

NVIDIA Rivea サーバ：センタ心理分析をサポートします

セットアップするには "NVIDIA RIVA"、次の手順を実行します。

1. NGC から Riva ファイルをダウンロード

```
ngc registry resource download-version
nvidia/riva/riva_quickstart:1.4.0-beta
```

2. Riva セットアップを初期化します (`Riva_init.sh`)
3. Riva サーバ (`Riva_start.sh`) を起動します
4. Riva クライアント (`Riva_start_client.sh`) を起動します
5. Riva クライアント内で、オーディオ処理ライブラリをインストールします（ "FFmpeg")

```
apt-get install ffmpeg
```

6. を起動します "Jupyter" サーバ
7. Riva Inference Pipeline Notebook を実行します。

NVIDIA TAO Toolkit ：センターの感情分析をサポートします

NVIDIA TAO Toolkit をセットアップするには、次の手順を実行します。

1. を準備してアクティブ化します "仮想環境" TAO ツールキット用。
2. をインストールします "必須パッケージ"。
3. トレーニング中および微調整中に使用したイメージを手動で引き出します。

```
docker pull nvcr.io/nvidia/tao/tao-toolkit-pyt:v3.21.08-py3
```

4. を起動します "Jupyter" サーバ
5. TAO 微調整ノートブックを実行します。

TAO モデルを **Riva** にエクスポート：センターの感情分析をサポートします

を使用してください "Rivea の Tao ツールキットモデル"、次の手順を実行します。

1. TAO 微調整ノートブックにモデルを保存します。
2. TAO トレーニング済みモデルを Riva モデルディレクトリにコピーします。
3. Riva サーバ (`Riva_start.sh`) を起動します

導入の障害です

独自の解決策を開発する際に留意すべき点をいくつかご紹介します。

- 最初に NetApp DataOps ツールキットをインストールし、データストレージシステムが最適に動作するようにします。
- NVIDIA NGC は、イメージとモデルのダウンロードを認証するため、それ以外のコンポーネントよりも先にインストールする必要があります。
- Rivea は、 TAO ツールキットの前にインストールする必要があります。Riva インストールでは、必要に応じてイメージをプルするように Docker デーモンが設定されます。
- DGX および Docker でモデルをダウンロードするには、インターネットアクセスが必要です。

検証結果

前のセクションで説明したように、 2 つ以上の機械学習モデルが順番に実行されている場合は常に、エラーがパイプライン全体に伝播されます。この解決策では、企業の株価リスクレベルを測定する上で最も重要な要因は、文章の感情です。音声対テキストモデルは、パイプラインに不可欠ですが、感情を予測する前に前処理単位として機能します。実際に重要なのは、基本的な真実文と予測された文の感情の違いです。これは、ワードエラーレート（ WER ）のプロキシとして機能します。音声とテキストの正確さは重要ですが、 WER は最終的なパイプラインメトリックでは直接使用されません。

```
PIPELINE_SENTIMENT_METRIC = MEAN(DIFF(GT_sentiment, ASR_sentiment))
```

これらの感情指標は、 F1 スコア、リコール、各文章の精度について計算できます。結果は集約され、各メトリックの信頼間隔とともに混乱マトリックス内に表示されます。

転送学習を使用する利点は、データ要件、トレーニング時間、コストの数分の 1 でモデルのパフォーマンスが向上することです。また、微調整されたモデルをベースラインバージョンと比較して、転送学習がインペアリングではなくパフォーマンスを向上させるようにする必要があります。つまり、調整済みモデルの方が、サポートセンターのデータのパフォーマンスが事前トレーニング済みモデルよりも優れているはずです。

パイプラインの評価

テストケース	詳細
テスト番号	パイプラインのセンチメント指標
テストの前提条件	音声 / テキストおよび感情分析モデル向けに微調整されたモデル
予想される結果	微調整されたモデルのセンチメント・メトリックは、元の事前トレーニング済みモデルよりも優れています。

パイプラインのセンチメント指標

1. ベースラインモデルのセンチメントメトリックを計算します。
2. 微調整モデルのセンチメントメトリックを計算します。
3. これらの指標間の差異を計算します。
4. すべての文の違いを平均化します。

ビデオとデモ

センチメント分析パイプラインを含むノートブックが 2 つあります。 "「サポート - センター - モデル - 転送 - 学習と微調整 .ipynb 」" および "「サポート - センタ - センチメント - 分析 - パイプライン .ipynb 」"。これらのノートブックは、ユーザーのデータに微調整された最先端のディープラーニングモデルを使用して、サポートセンターのデータを取り込み、各文から感情を抽出するパイプラインを開発する方法を示しています。

サポートセンター **-** 感情分析パイプライン **.ipynb**

このノートブックには、オーディオの取り込み、テキストへの変換、外部ダッシュボードで使用するための感情の抽出を行う推論 Riva パイプラインが含まれています。データセットは、まだダウンロードされていない場合は自動的にダウンロードされて処理されます。ノートブックの最初のセクションは、音声ファイルからテキストへの変換を処理する Speech to Text です。続いて、各テキスト文の感情を抽出し、それらの結果を提案されたダッシュボードと同様の形式で表示する感情分析セクションが表示されます。

> MP3 データセットをダウンロードして正しい形式に変換する必要があるため、このノートブックはモデルのトレーニングや微調整の前に実行する必要があります。

Call Center - Sentiment Analysis Pipeline

This notebook demonstrates how to build a pipeline for sentiment analysis of call center conversations. The goal of this pipeline is to develop sentiment analysis for use within an external dashboard.

This tutorial will guide you through the use of NVIDIA's RIVA for automatic speech recognition and text classification. This tutorial uses NetApp cloud storage for data storage and a pre-trained RIVA model.

Channels

These are the channels on which RIVA is hosting models.

- speech: `51051`
- voice: `61051`

These channels **must** be aligned with `riva_speech_api_port` and `riva_vision_api_port` within `config.sh`

In [4]:
```
speech_channel = "localhost:51051"
voice_channel = "localhost:61051"
```

Speech-To-Text

Automatic Speech Recognition (ASR) takes as input an audio stream or audio buffer and returns one or more text transcripts, along with additional optional metadata. ASR represents a full speech recognition pipeline that is GPU accelerated with optimized performance and accuracy. ASR supports synchronous and streaming recognition modes.

For more information on NVIDIA RIVA's Automatic Speech Recognition, visit here.

Constants

Use these constants to affect different aspects of this pipeline:

- `DATA_DIR` : base folder where data is stored
- `DATASET_NAME` : name of the call center dataset
- `COMPANY_DATE` : folder name identifying the particular call center conversation

サポートセンター **-** モデルトレーニングと微調整 **.ipynb**

ノートブックを実行する前に、 TAO Toolkit 仮想環境を設定する必要があります (インストール手順については、『 Commands Overview 』の TAO Toolkit の項を参照してください) 。

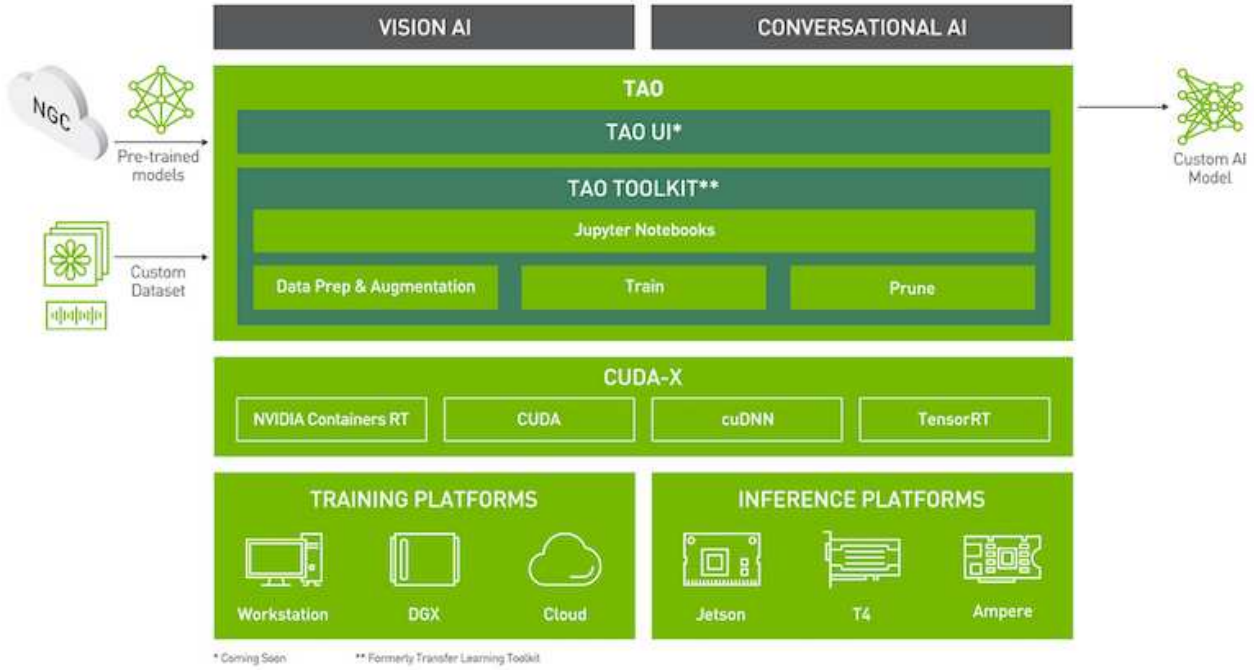
このノートブックは、 TAIO ツールキットを使用して、お客様のデータに基づいてディープラーニングモデルを微調整します。前のノートブックと同様に、この 2 つのセクションに分かれて、 Speech to Text コンポーネントと、センチメント分析コンポーネントが表示されます。各セクションでは、データ処理、モデルトレーニング、微調整、結果の評価、およびモデルのエクスポートについて説明します。最後に、 Riva で使用するために、両方の微調整済みモデルを導入するための最終セクションがあります。

Call Center - Model Transfer Learning and Fine-Tuning

TAO Toolkit is a python based AI toolkit for taking purpose-built pre-trained AI models and customizing them with your own data. Transfer learning extracts learned features from an existing neural network to a new one. Transfer learning is often used when creating a large training dataset is not feasible in order to enhance the base performance of state-of-the-art models.

For this call center solution, the speech-to-text and sentiment analysis models are fine-tuned on call center data to augment the model performance on business specific terminology.

For more information on the TAO Toolkit, please visit here.



Installing necessary dependencies

For ease of use, please install TAO Toolkit inside a python virtual environment. We recommend performing this step first and then launching the notebook from the virtual environment. Please refer to the README for these instructions.

まとめ

顧客体験が競争上の重要な戦場と見なされるようになった今、 AI を強化したグローバルサポートセンターは、ほぼすべての業界の企業が無視することができない重要な要素となっています。このテクニカルレポートで提案している解決策は、こうした優れたカスタマーエクスペリエンスの提供を支援することを実証しています。課題は、企業が AI インフラとワークフローを最新化するためのアクションを取ることです。

顧客サービスにおける AI の最適な実装は、人事担当者の代わりになるものではありません。AI は、リアルタイムの感情分析、紛争のエスカレーション、マルチモーダルの感情コンピューティングを通じて、優れた顧客体験を生み出す力を発揮します。これにより、包括的な AI モデルが大規模に推奨事項を提示し、個々のヒューマンエージェントが欠けている可能性のある点を補足する、言葉、言葉以外の顔の手がかりを検出できます。AI は、特定のお客様と現在対応可能なエージェントをよりよくマッチさせることもできます。AI を活用

することで、企業は、プロバイダの製品、サービス、ブランドイメージに対する顧客の考えや印象に関する価値ある感情を引き出すことができます。

解決策を使用して、客観的なパフォーマンス評価指標として機能するように、サポートエージェントの時系列データを作成することもできます。従来の顧客満足度調査では、十分な回答が得られないことが雇用者は、長期的な従業員や顧客の感情を収集することで、サポートエージェントのパフォーマンスに関して十分な情報に基づいた判断を下すことができます。

ネットアップ、 SFL Scientific 、オープンソースのオーケストレーションフレームワーク、 NVIDIA を組み合わせることで、最新テクノロジをマネージドサービスとして統合し、優れた柔軟性を提供することで、テクノロジの採用を促進し、新しい AI / ML アプリケーションの市場投入期間を短縮できます。これらの高度なサービスはオンプレミスで提供され、クラウドネイティブ環境やハイブリッド導入アーキテクチャへの移植が容易です。

追加情報の参照先

このドキュメントに記載されている情報の詳細については、以下のドキュメントや Web サイトを参照してください。

- 3D 対話型デモ

  "www.netapp.com/ai"

- ネットアップの AI スペシャリストと直接つながる

  "https://www.netapp.com/artificial-intelligence/"

- ネットアップ解決策が実現する NVDIA 基本コマンドプラットフォームの概要

  https://www.netapp.com/pdf.html?item=/media/32792-DS-4145-NVIDIA-Base-Command-Platform-with-NetApp.pdf

- AI 向けネットアップ 10 の理由を解説したインフォグラフィック

  "https://www.netapp.com/us/media/netapp-ai-10-good-reasons.pdf"

- ヘルスケア分野の AI ：『 Deep learning to identify COVID-19 nscions in lung CT scans 』ホワイトペーパーを参照してください

  https://www.netapp.com/pdf.html?item=/media/31240-WP-7342.pdf

- ヘルスケア分野の AI ：ヘルスケア設定におけるフェイスマスクの使用状況を監視するホワイトペーパーです

  https://www.netapp.com/pdf.html?item=/media/37490-NA-611-Monitoring-face-mask-usage-in-healthcare-settings.pdf

- 医療分野の AI ：診断画像診断テクニカルレポート

  https://www.netapp.com/pdf.html?item=/media/7395-tr4811.pdf

- 小売業向け AI ：ネットアップの会話型 AI で NVIDIA Riva を使用

  "https://docs.netapp.com/us-en/netapp-solutions/ai/cainvidia_executive_summary.html"

- NetApp ONTAP AI 解決策の概要

  https://www.netapp.com/pdf.html?item=/media/6736-sb-3939.pdf

- NetApp DataOps ツールキットの解決策概要

  https://www.netapp.com/pdf.html?item=/media/21480-SB-4111-1220-NA-Data-Science-Toolkit.pdf

- ネットアップの AI コントロールプレーン解決策の概要

  https://www.netapp.com/pdf.html?item=/media/6737-sb-4055.pdf

- 『データの活用で業界を変革』の E ブック

  "https://www.netapp.com/us/media/na-337.pdf"

- NetApp EF シリーズ AI 解決策の概要

  https://www.netapp.com/pdf.html?item=/media/26708-SB-4136-NetApp-AI-E-Series.pdf

- AI 推論向けのネットアップの AI と Lenovo ThinkSystem 解決策の概要

  https://www.netapp.com/pdf.html?item=/media/25316-SB-4129.pdf

- エンタープライズ AI および ML 向けのネットアップ AI と Lenovo ThinkSystem の解決策概要

  https://www.netapp.com/pdf.html?item=/media/25317-SB-4128.pdf

- ネットアップと NVIDIA – AI ビデオで可能なことを再定義

  https://www.youtube.com/watch?v=38xw65SteUc

Azure での分散トレーニング - クリックスルー率予測

TR-4904 ：『 Distributed Training in Azure - Click Through Rate Prediction 』

ネットアップ、 Verron Martina 、 Muneer Ahmad 、 Rick Huang 氏

データサイエンティストの仕事は、機械学習（ ML ）モデルと人工知能（ AI ）モデルのトレーニングと調整に集中する必要があります。しかし、 Google の調査によると、データサイエンティストは、モデルをエンタープライズアプリケーションと連携させ、大規模に運用する方法を検討する時間の約 80% を費やしています。

エンドツーエンドの AI / ML プロジェクトを管理するには、エンタープライズコンポーネントについてより広範な理解が必要です。DevOps がその定義、統合、導入を引き継ぎましたが、 ML の運用では、 AI や ML プロジェクトを含む同様のフローがターゲットとなります。エンドツーエンドの AI / ML パイプラインが企業内でどのように影響するかを知るには、次の必要なコンポーネントのリストを参照してください。

- ストレージ
- ネットワーキング
- データベース

- ファイルシステム
- コンテナ
- 継続的統合 / 継続的導入（ CI / CD ）パイプライン
- 統合開発環境（ IDE ）
- セキュリティ
- データアクセスポリシー
- ハードウェア
- クラウド
- 仮想化
- データサイエンスのツールセットとライブラリ

対象読者

データサイエンスの世界は、 IT とビジネスのさまざまな分野に影響をもたらしています。

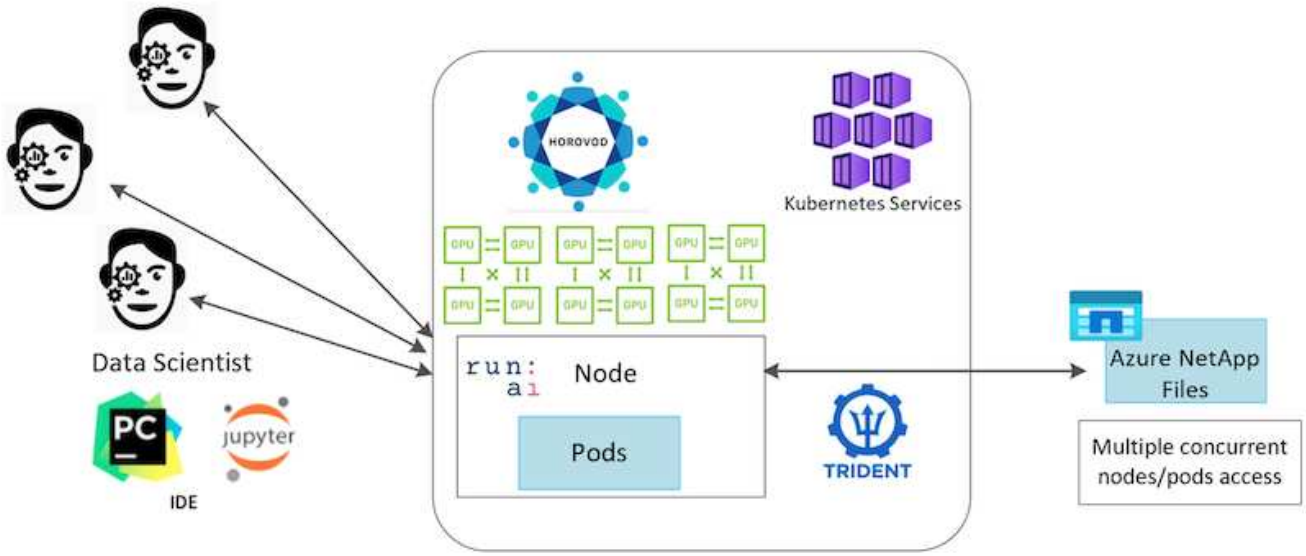
- データサイエンティストは、選択したツールとライブラリを柔軟に使用できる必要があります。
- データエンジニアは、データの流れと配置場所を把握する必要があります。
- DevOps エンジニアは、新しい AI / ML アプリケーションを CI / CD パイプラインに統合するためのツールを必要としています。
- クラウド管理者とアーキテクトは、 Azure リソースをセットアップおよび管理できる必要があります。
- ビジネスユーザは、 AI / ML アプリケーションにアクセスしたいと考えています。

このテクニカルレポートでは、 Azure NetApp Files 、 Rapids AI 、 Dask 、 Azure が、これらの各役割がビジネスにもたらす価値について説明します。

解決策の概要

この解決策は、 AI / ML アプリケーションのライフサイクルに従います。まず、データサイエンティストの仕事から始めて、データの準備やモデルのトレーニングに必要なさまざまなステップを定義します。Dask のラピッズを活用することで、 Azure Kubernetes Service （ AKS ）クラスタ全体で分散トレーニングを実施し、従来の Python の坐骨神経痛手法に比べてトレーニング時間を大幅に短縮しました。完全なサイクルを完了するには、パイプラインと Azure NetApp Files を統合します。

Azure NetApp Files は、さまざまなパフォーマンス階層を提供します。お客様はまず Standard 階層から始めて、データを移動することなく、スケールアウトしてハイパフォーマンス階層まで無停止でスケールアップできます。この機能により、データサイエンティストは、次の図に示すように、パフォーマンスの問題を発生させることなく、大規模なモデルのトレーニングを実施できます。クラスタ全体にデータサイロが発生することはありません。



テクノロジの概要

このページでは、この解決策で使用されているテクノロジの概要を説明します。

Microsoft とネットアップ

2019 年 5 月より、 Microsoft は Azure ネイティブのファーストパーティポータルサービスを提供し、 NetApp ONTAP テクノロジをベースとしたエンタープライズ NFS および SMB ファイルサービスを提供しています。この開発は、 Microsoft とネットアップの戦略的パートナーシップによって推進されており、ワールドクラスの ONTAP データサービスの Azure への対応範囲がさらに拡大しています。

Azure NetApp Files の特長

Azure NetApp Files サービスは、エンタープライズクラスの高パフォーマンスな従量課金制のファイルストレージサービスです。Azure NetApp Files は、あらゆる種類のワークロードに対応し、デフォルトで高可用性を実現します。サービスレベルとパフォーマンスレベルを選択し、サービスを使用して Snapshot コピーをセットアップできます。Azure NetApp Files は Azure ファーストパーティサービスで、コードを変更することなく、データベース、 SAP 、ハイパフォーマンスコンピューティングアプリケーションなど、クラウドで最も要件の厳しいエンタープライズファイルワークロードを移行して実行します。

このリファレンスアーキテクチャには、 IT 組織に次のようなメリットがあります。

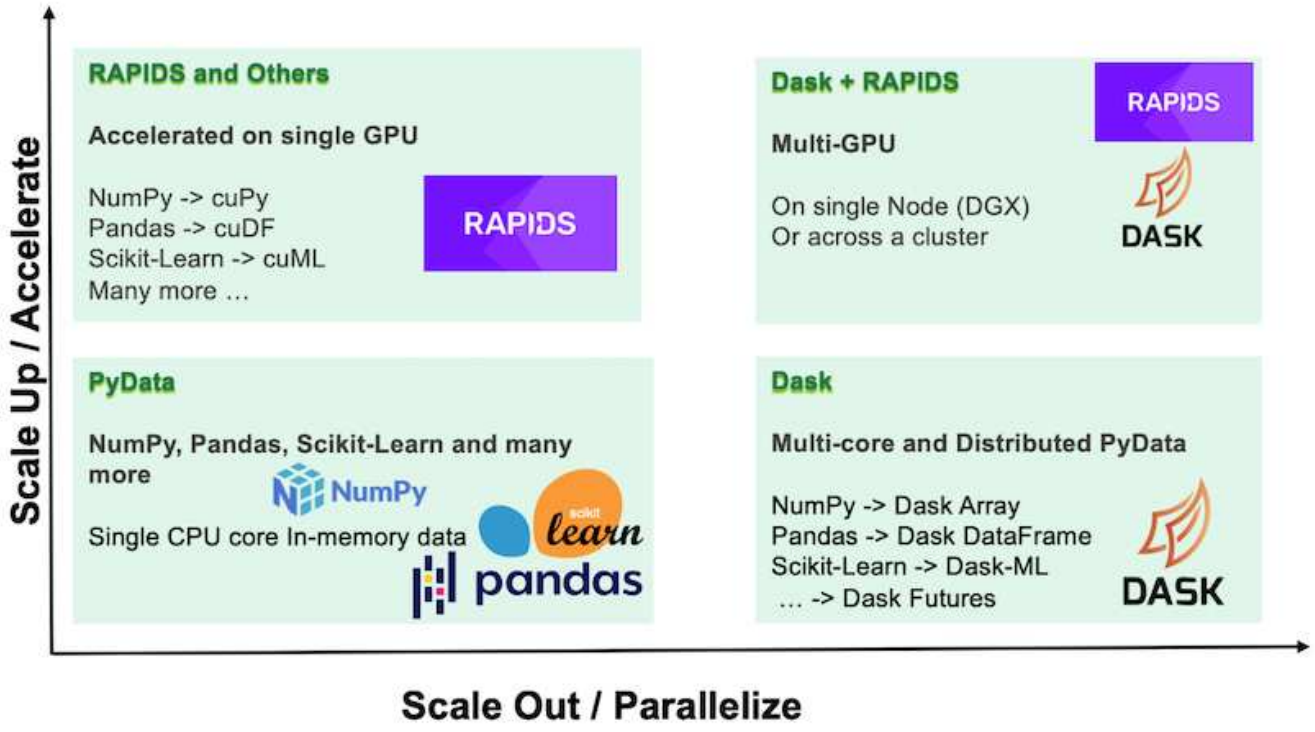
- 設計の複雑さを解消
- コンピューティングとストレージを個別に拡張できます
- 小規模構成から始めて、シームレスに拡張できます
- さまざまなパフォーマンスとコストを考慮して、幅広いストレージ階層を提供します

Dask と NVIDIA Rapids の概要

Dask は、 Python ライブラリを複数のマシン上で拡張し、大量のデータを高速処理する、オープンソースの並列コンピューティングツールです。これは、 Pandas 、 numpy 、 scikit learn などのシングルスレッド従来の Python ライブラリに類似した API を提供します。その結果、ネイティブの Python ユーザは、クラスタ全体でリソースを使用するために既存のコードを大幅に変更する必要がなくなります。

NVIDIA Rapids はオープンソースライブラリのスイートで、 GPU 上でエンドツーエンドの ML ワークフローとデータ分析ワークフローを完全に実行できます。Dask と組み合わせることで、 GPU ワークステーション（スケールアップ）からマルチノードのマルチ GPU クラスタ（スケールアウト）へ簡単に拡張できます。

クラスタに Dask を導入するには、 Kubernetes を使用してリソースのオーケストレーションを行います。次の図に示すように、ワーカーノードをプロセス要件に従ってスケールアップまたはスケールダウンすることもできます。これは、クラスタのリソース消費を最適化するのに役立ちます。



ソフトウェア要件

次の表に、この解決策に必要なソフトウェア要件を示します。

ソフトウェア	バージョン
Azure Kubernetes Service の略	1.18.14
Rapids および Dask の容器のイメージ	リポジトリ : "rapidsai/rapidsai" タグ :0.17-cudda11.1-runtime-ubuntu18.04
NetApp Trident	20.01.1
Helm	3.0.0

クラウドリソースの要件

このページでは、 Azure NetApp Files のクラウドリソースの設定について説明します。

Azure NetApp Files を設定します

の説明に従って、 Azure NetApp Files を設定します "クイックスタート： Azure NetApp Files をセットアップし、 NFS ボリュームを作成します"。

「 Azure NetApp Files 用 NFS ボリュームの作成」のセクションを過ぎても、 Trident を使用してボリュームを作成できます。続行する前に、次の手順を実行します。

1. Azure NetApp Files とネットアップのリソースプロバイダに（ Azure Shell を使用）登録します "リンク" ）。
2. Azure NetApp Files でアカウントを作成します（ "リンク"）。
3. 容量プール（必要に応じて、 4TB 以上の Standard または Premium ）をセットアップします（ "リンク" ）。次の表に、クラウドでのセットアップに必要なネットワーク構成を示します。Dask クラスタと Azure NetApp Files は同じ Azure Virtual Network （ VNet ）またはピア関係にある VNet に配置されている必要があります。

リソース	「 **/version** 」と入力します
Azure Kubernetes Service の略	1.18.14
エージェントノード	3x Standard_DS2_v2
GPU ノード	3x Standard_NC6s_v3
Azure NetApp Files の特長	標準的な容量のプールがある
容量（ TB ）	4.

クリックスルー率予測ユースケースの概要

このユースケースは、一般に公開されているに基づいています "[ログ をクリックします"^] データセットの作成元 "Crito AI Lab の略"。ML プラットフォームとアプリケーションの最近の進歩により、現在は大規模な学習が注目されています。クリックスルー率（ CTR ）は、オンライン広告インプレッション数 100 件あたりの平均クリックスルー数（パーセンテージ）と定義されています。デジタルマーケティング、小売、 E コマース、サービスプロバイダなど、さまざまな業界やユースケースで重要な指標として広く採用されています。CTR を潜在的な顧客トラフィックの重要な指標として使用する例を以下に示します。

- * デジタルマーケティング :* インチ "Google アナリティクス"、 CTR は、広告主または販売主のキーワード、広告、および無料リストがどの程度効果を発揮しているかを測定するために使用できます。クリック率が高いと、ユーザーは広告やリストを便利で関連性の高いものとして見つけることができます。CTR はまたあなたのキーワードの予想される CTR に貢献する、の構成要素である "広告ランク"。
- * e- コマース： * 活用に加えて "Google アナリティクス"E コマースバックエンドには、少なくともいくつかの訪問者統計情報があります。これらの統計情報は一目見ると有用ではないように見えますが、通常は読みやすく、他の情報よりも正確な情報になる可能性があります。このような統計で構成されるファーストパーティデータセットは独占的なものであり、 E コマースの販売者、購買担当者、プラットフォームに最も関連性があります。これらのデータセットは、ベンチマークの設定に使用でき、過去 1 年と過去 1 日の間に結果を比較するために、さらに詳細な分析を行うための時系列を作成します。
- * 小売： * 実店舗の小売業者は、訪問者数と顧客数を CTR に関連付けることができます。お客様の数は、販売時点の履歴から確認できます。小売業者のウェブサイトや広告トラフィックの CTR が、前述の売上につながる可能性があります。ロイヤルティプログラムは、オンライン広告や他の Web サイトからリダイレクトされたお客様が報奨を獲得するために参加する可能性があるため、別のユースケースです。小売業者は、ロイヤルティプログラムを通じて顧客を獲得し、販売履歴から行動を記録することで、さまざまなカテゴリーで消費者の購買行動を予測するだけでなく、クーポンをパーソナライズし、チャーンを減らす推奨システムを構築できます。

- * 通信事業者とインターネット・サービス・プロバイダーは、豊富なデータを提供するファーストパーティのユーザー・テレメトリ・データを使用して、洞察に富んだ AI 、 ML 、分析のユースケースを実現しています。たとえば、携帯電話会社の Web 閲覧のトップレベルのドメイン履歴ログを毎日活用して、既存のモデルを微調整して最新のオーディエンスセグメンテーションを作成したり、顧客の行動を予測したり、リアルタイム広告を配置してオンライン体験を向上させることができます。このようなデータ主導のマーケティングワークフローでは、 CTR はコンバージョンを反映する重要な指標です。

デジタルマーケティングの文脈では、 "Crito Terabyte のログをクリックします" 現在は、 ML プラットフォームとアルゴリズムのスケーラビリティを評価する際の参考データセットとなっています。広告主は、クリックスルーレートを予測することで、広告に対応する可能性が最も高い訪問者を選択し、閲覧履歴を分析し、ユーザーの関心に基づいて最も関連性の高い広告を表示できます。

このテクニカルレポートで紹介する解決策には、次のようなメリットがあります。

- 分散型または大規模なトレーニングにおける Azure NetApp Files の利点
- CUDA 対応のデータ処理（ cDF 、 cuPy など）と ML アルゴリズム（ cuML ）をラピッズで表示
- 分散型トレーニング用 Dask 並列コンピューティングフレームワーク

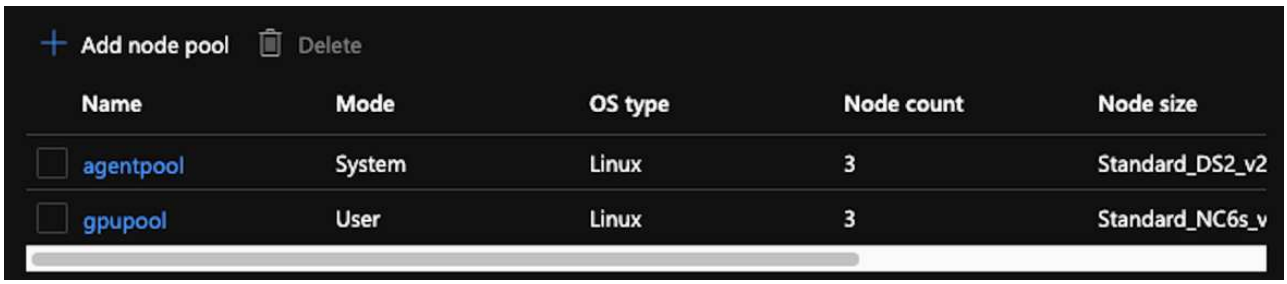
Rapids AI と Azure NetApp Files をベースに構築されたエンドツーエンドのワークフローでは、ランダムフォレストモデルのトレーニング時間が 2 桁単位で大幅に短縮されたことが示されています。この点は、構造化された表形式データが 45 GB （平均）の実世界のクリックログを毎日処理する場合の従来の Pandas アプローチと比べて大幅に改善されています。これは、約 20 億行を含む DataFrame に相当します。このテクニカルレポートでは、クラスタ環境のセットアップ、フレームワークとライブラリのインストール、データのロードと処理、従来型のトレーニングと分散型のトレーニング、可視化と監視について説明し、重要なエンドツーエンドのランタイム結果を比較します。

セットアップ（ **Setup** ）

AKS クラスタをインストールしてセットアップします

AKS クラスタをインストールしてセットアップする方法については、 Web ページを参照してください "AKS クラスタを作成します" 次に、次の手順を実行します。

1. ノードのタイプ（ system [CPU] ノードまたは worker[GPU] ノード）を選択するときは、次のいずれかを選択します。
   a. プライマリ・システム・ノードは ' 標準 DS2v2 （デフォルトでは 3 ノード）である必要があります
   b. 次に 'gpupool' という名前のユーザ・グループ（ GPU ノードの場合）のワーカー・ノード Standard_NC6s_v3 プール（最小 3 ノード）を追加します



2. 導入には 5 〜 10 分かかります。完了したら、 Connect to Cluster （クラスタへの接続）をクリックします。

3. 新しく作成した AKS クラスタに接続するには、ローカル環境（ラップトップ / PC ）から次のものをインストールします。
   a. を使用した Kubernetes コマンドラインツール "使用している OS に応じた手順が表示されます"
   b. 本ドキュメントに記載されている Azure CLI を使用して、 "Azure CLI をインストールします"
4. 端末から AKS クラスタにアクセスするには、「 AZ login 」と入力し、クレデンシャルを入力します。
5. 次の 2 つのコマンドを実行します。

```
az account set --subscription xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
aks get-credentials --resource-group resourcegroup --name aksclustername
```

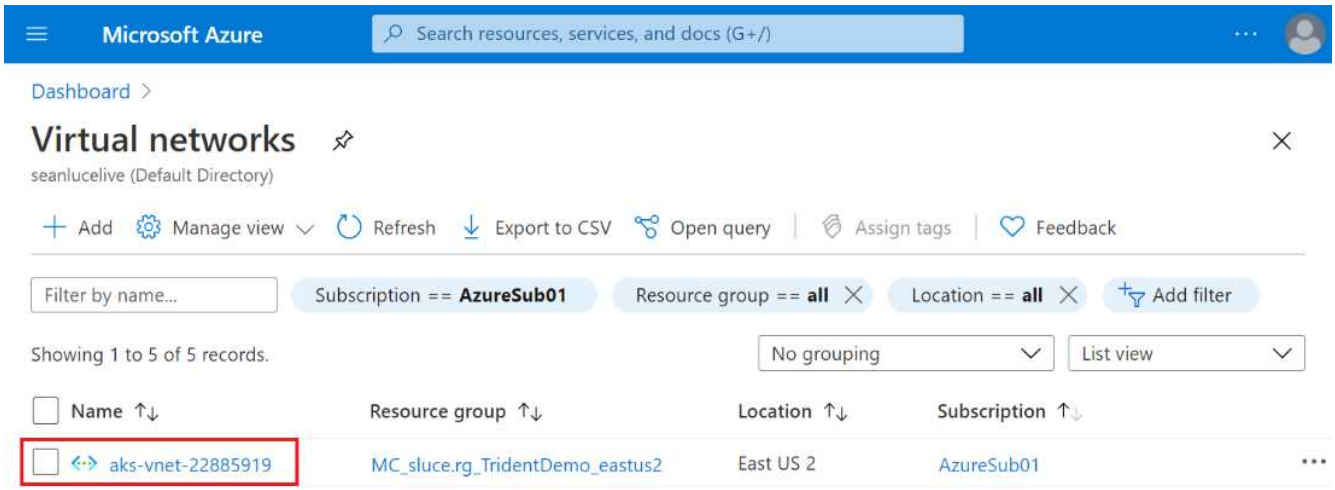
6. 「 Azure CLI ： kubectl get nodes 」と入力します。
7. 次の例に示すように、 6 つのノードがすべて稼働していれば、 AKS クラスタをローカル環境に接続することができます

```
verronmartina@verron-mac-0 ~ % kubectl get nodes
NAME                                STATUS   ROLES   AGE   VERSION
aks-agentpool-34613062-vmss000000   Ready    agent   22m   v1.18.14
aks-agentpool-34613062-vmss000001   Ready    agent   22m   v1.18.14
aks-agentpool-34613062-vmss000002   Ready    agent   22m   v1.18.14
aks-gpupool-34613062-vmss000000     Ready    agent   20m   v1.18.14
aks-gpupool-34613062-vmss000001     Ready    agent   20m   v1.18.14
aks-gpupool-34613062-vmss000002     Ready    agent   20m   v1.18.14
verronmartina@verron-mac-0 ~ %
```

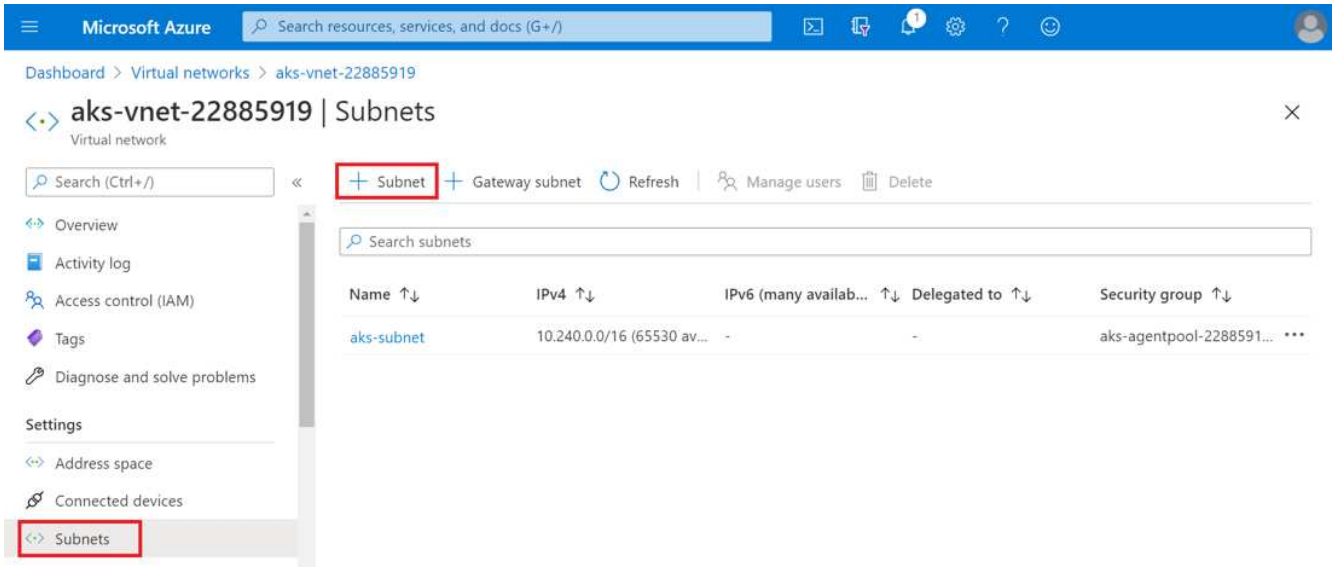
Azure NetApp Files の委譲されたサブネットを作成します

Azure NetApp Files の委任されたサブネットを作成するには、次の手順を実行します。

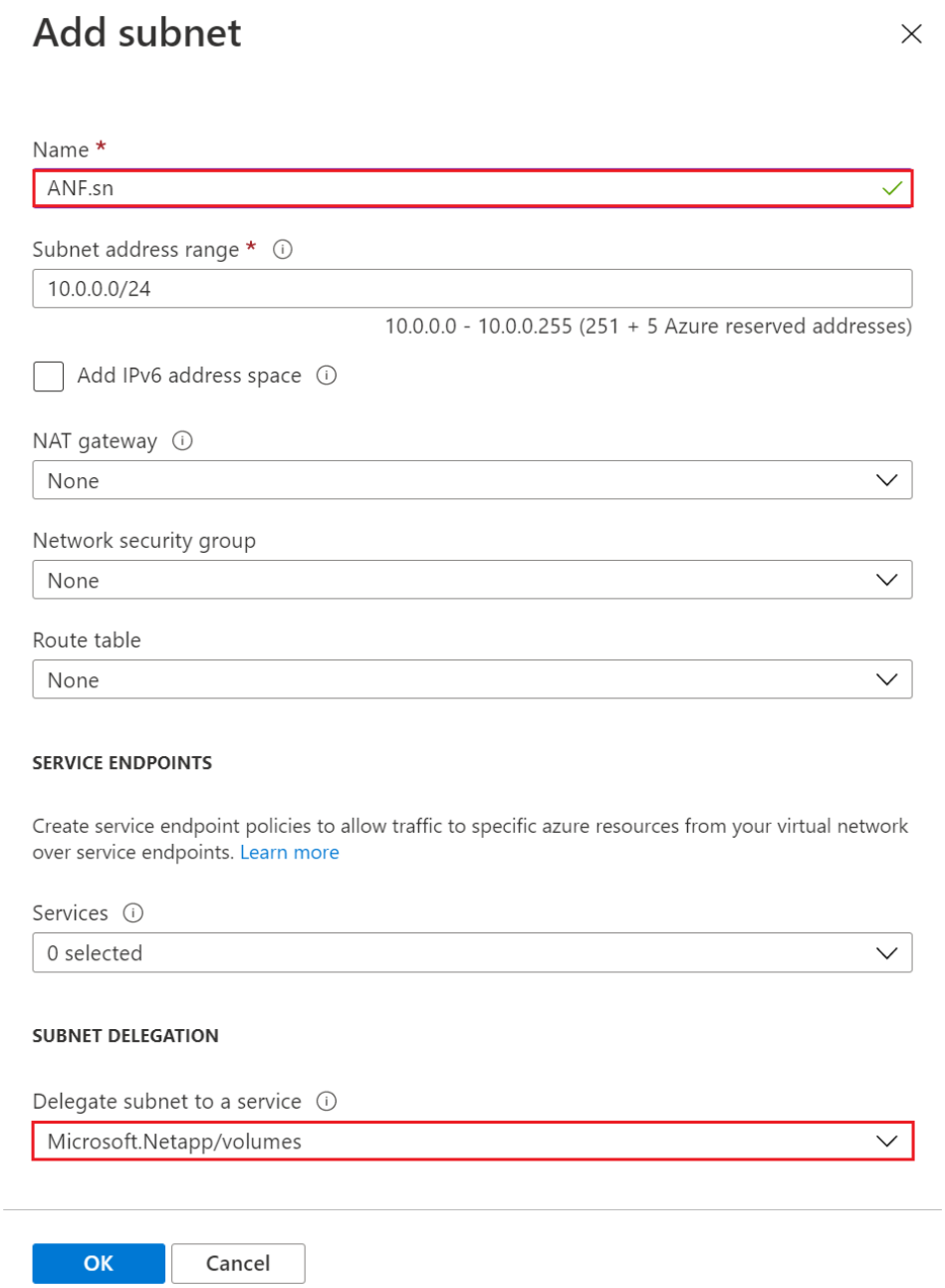
1. Azure ポータル内の仮想ネットワークに移動します。新しく作成した仮想ネットワークを検索します。「 AKs-vnet 」などのプレフィックスが必要です。
2. VNet の名前をクリックします。



3. [サブネット] をクリックし、上部のツールバーの [サブネット] をクリックします。



4. サブネットに「 ANF 」などの名前を付け、「サブネットの委任」見出しの下にある「 Microsoft.Netapp/volumes` 」を選択します。他のものは変更しないでください。[OK] をクリックします。



Azure NetApp Files ボリュームはアプリケーションクラスタに割り当てられ、 Kubernetes で永続ボリューム要求（ PVC ）として使用されます。その結果、 Jupyter ノートブック、サーバーレス関数などのさまざまなサービスに柔軟にマップできます。

サービスのユーザは、プラットフォームのストレージをさまざまな方法で消費できます。このテクニカルレポートでは NFS について説明しているため、 Azure NetApp Files の主なメリットは次のとおりです。

- ユーザに Snapshot コピーを使用できるようにする。
- ユーザが Azure NetApp Files ボリュームに大量のデータを格納できるようにする。
- 大容量のファイルセットでモデルを実行する場合、 Azure NetApp Files のパフォーマンスが向上します。

ピア AKS の VNet と Azure NetApp Files VNet

AKS VNet を Azure NetApp Files VNet にピアリングするには、次の手順を実行します。

1. 検索フィールドに Virtual Networks と入力します。
2. 「 vnet AK - vnet-name 」を選択します クリックして、検索フィールドに peerings と入力します。
3. + Add をクリックします。
4. 次の記述子を入力します。
   a. ピアリングリンク名は 'AKs-vnet-name_-to-anf' です
   b. VNet ピアリングパートナーとしての SubscriptionID および Azure NetApp Files VNet
   c. アスタリスク以外のすべてのセクションは、デフォルト値のままにします。
5. 追加をクリックします。

詳細については、を参照してください "仮想ネットワークピアリングを作成、変更、削除します"。

Trident をインストール

Helm を使用して Trident をインストールするには、次の手順を実行します。

1. Install Helm （インストール手順については、を参照してください） "ソース"）。
2. Trident 20.01.1 インストーラをダウンロードして展開します。

```
$wget
$tar -xf trident-installer-21.01.1.tar.gz
```

3. ディレクトリを 'trident-installer' に変更します

```
$cd trident-installer
```

4. tridentctl' をシステム「 $PATH 」のディレクトリにコピーします。

```
$sudo cp ./tridentctl /usr/local/bin
```

5. Kubernetes （ Kubernetes ）クラスタに Trident をインストールし、 Helm （を参照 "ソース"）：
   a. ディレクトリを 'helm' ディレクトリに変更します

   ```
   $cd helm
   ```

   b. Trident をインストール

```
$helm install trident trident-operator-21.01.1.tgz --namespace
trident --create-namespace
```

c. Trident ポッドのステータスを確認

```
$kubectl -n trident get pods
```

すべてのポッドが稼働中の場合は、 Trident がインストールされてから次のポッドに移動できます。

6. AKS の Azure NetApp Files バックエンドとストレージクラスをセットアップします。

a. Azure サービスプリンシパルを作成します。

サービスプリンシパルは、 Trident が Azure と通信して Azure NetApp Files リソースを操作する方法を示します。

```
$az ad sp create-for-rbac --name ""
```

出力は次の例のようになります。

```
{
"appId": "xxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
"displayName": "netapptrident",
"name": "",
"password": "xxxxxxxxxxxxxxx.xxxxxxxxxxxxxx",
"tenant": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx"
}
```

7. Trident バックエンド JSON ファイルを作成します。例：「 anf-backend.json 」
8. 任意のテキストエディタを使用して 'anf-backend.json ファイル内の次のフィールドに値を入力します

```
{
    "version": 1,
    "storageDriverName": "azure-netapp-files",
    "subscriptionID": "fakec765-4774-fake-ae98-a721add4fake",
    "tenantID": "fakef836-edc1-fake-bff9-b2d865eefake",
    "clientID": "fake0f63-bf8e-fake-8076-8de91e57fake",
    "clientSecret": "SECRET",
    "location": "westeurope",
    "serviceLevel": "Standard",
    "virtualNetwork": "anf-vnet",
    "subnet": "default",
    "nfsMountOptions": "vers=3,proto=tcp",
    "limitVolumeSize": "500Gi",
    "defaults": {
    "exportRule": "0.0.0.0/0",
    "size": "200Gi"
}
```

9. 次のフィールドを置き換えます。
   - ' スクリプト ID' 。お客様の Azure サブスクリプション ID
   - 「 tenantID 」。前の手順で「 AZ AD SP 」の出力から取得した Azure テナント ID 。
   - 「 clientID 」。前のステップで 'AZ ad sp' の出力からのあなたの appID 。
   - 「 clientSecret 」を入力します。前の手順で「 AZ ad sp 」の出力から得たパスワード。
10. 構成ファイルとして 'anf-backend.json を使用して 'trident'namespace に Azure NetApp Files バックエンドを作成するように Trident に指示します

```
$tridentctl create backend -f anf-backend.json -n trident
```

```
+------------------------+--------------------+--------------------------------------+--------+---------+
|          NAME          |   STORAGE DRIVER   |                 UUID                 | STATE  | VOLUMES |
+------------------------+--------------------+--------------------------------------+--------+---------+
| azurenetappfiles_86181 | azure-netapp-files | 2ca85462-59ac-4946-be05-c03f5575a2ad | online |       0 |
+------------------------+--------------------+--------------------------------------+--------+---------+
```

11. ストレージクラスを作成する。Kubernetes ユーザは、名前でストレージクラスを指定する PVC を使用してボリュームをプロビジョニングします。前の手順で作成した Trident バックエンドを参照するストレージクラス「 azurenetappfiles 」を作成するよう、 Kubernetes に指示します。
12. ストレージクラスおよびコピー用の YAML （ 'anf-storage-class.yaml ）ファイルを作成します。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
name: azurenetappfiles
provisioner: netapp.io/trident
parameters:
backendType: "azure-netapp-files"
$kubectl create -f anf-storage-class.yaml
```

13. ストレージクラスが作成されたことを確認します。

```
kubectl get sc azurenetappfiles
```

```
NAME               PROVISIONER            RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
azurenetappfiles   csi.trident.netapp.io  Delete          Immediate           false                  98s
```

Helm を使用して、 **AKS** で **Rapids** デプロイメントを使用して **Dask** をセットアップします

Helm を使用して AKS で Rapids を使用して Dask をセットアップするには、次の手順を実行します。

1. Dask with Rapids をインストールするための名前空間を作成します。

```
kubectl create namespace rapids-dask
```

2. クリックスルーレートデータセットを保存する PVC を作成します。
   a. 次の YAML コンテンツをファイルに保存して PVC を作成します。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-criteo-data
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1000Gi
  storageClassName: azurenetappfiles
```

   b. YAML ファイルを Kubernetes クラスタに適用します。

```
kubectl -n rapids-dask apply -f <your yaml file>
```

3. rapidsai git リポジトリを複製します ("https://github.com/rapidsai/helm-chart"）。

```
git clone https://github.com/rapidsai/helm-chart helm-chart
```

4. 値 .yaml を変更し、作業者および Jupyter ワークスペース用に前に作成した PVC を含めます。
   a. リポジトリの 'rapidsai' ディレクトリに移動します

```
cd helm-chart/rapidsai
```

   b. 「 values] .yaml ファイルを更新し、 PVC を使用してボリュームをマウントします。

```
dask:
  …
  worker:
    name: worker
    …
    mounts:
      volumes:
        - name: data
          persistentVolumeClaim:
            claimName: pvc-criteo-data
      volumeMounts:
        - name: data
          mountPath: /data
    …
  jupyter:
    name: jupyter
    …
    mounts:
      volumes:
        - name: data
          persistentVolumeClaim:
            claimName: pvc-criteo-data
      volumeMounts:
        - name: data
          mountPath: /data
    …
```

5. リポジトリのホーム・ディレクトリに移動し 'Helm を使用して AKS 上に 3 つのワーカー・ノードを持つ Dask を展開します

```
cd ..
helm dep update rapidsai
helm install rapids-dask --namespace rapids-dask rapidsai
```

Azure NetApp Files のパフォーマンス階層

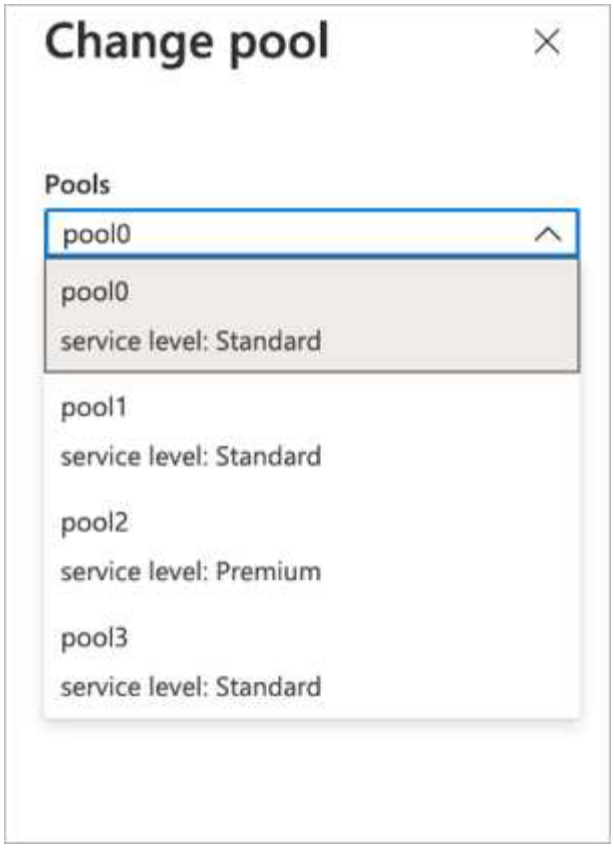
既存のボリュームのサービスレベルを変更するには、そのボリュームに必要なサービスレベルを使用する別の容量プールにボリュームを移動します。この解決策を使用することで、お客様は、まず小規模なデータセットと少数の GPU を標準階層に配置し、データ量と GPU の増加に合わせてスケールアウトまたは Premium Tier へのスケールアップを行うことができます。Premium Tier は、 Standard 階層のテラバイトあたりスループットの 4 倍を提供し、ボリュームのサービスレベルを変更するためにデータを移動することなくスケールアップを実行できます。

ボリュームのサービスレベルを動的に変更する

ボリュームのサービスレベルを動的に変更するには、次の手順を実行します。

1. Volumes （ボリューム）ページで、サービスレベルを変更するボリュームを右クリックします。［ プールの変更 ］ を選択します

2. プールの変更ウィンドウで、ボリュームの移動先となる容量プールを選択します。



3. [OK] をクリックします。

パフォーマンス階層の変更を自動化

パフォーマンス階層の変更を自動化するには、次のオプションを使用できます。

- 現在も動的サービスレベルの変更はパブリックプレビューで有効になっており、デフォルトでは有効になっていません。Azure サブスクリプションでこの機能を有効にする方法については、このドキュメントを参照してください "ボリュームのサービスレベルを動的に変更する"。
- Azure CLI の volume pool change コマンドについては、を参照してください "ボリュームプールの変更に関するドキュメント" 次に例を示します。

```
az netappfiles volume pool-change -g mygroup --account-name myaccname
--pool-name mypoolname --name myvolname --new-pool-resource-id
mynewresourceid
```

- PowerShell "set-AzNetAppFilesVolumePool コマンドレット" Azure NetApp Files ボリュームのプールを変更し、次の例に示すようにします。

```
Set-AzNetAppFilesVolumePool
-ResourceGroupName "MyRG"
-AccountName "MyAnfAccount"
-PoolName "MyAnfPool"
-Name "MyAnfVolume"
-NewPoolResourceId 7d6e4069-6c78-6c61-7bf6-c60968e45fbf
```

クリックスルー率予測データの処理とモデルトレーニング

データ処理およびモデルトレーニング用のライブラリ

次の表に、このタスクの構築に使用されたライブラリとフレームワークを示します。これらのコンポーネントはすべて、 Azure の役割ベースのアクセスおよびセキュリティ制御と完全に統合されています。

ライブラリ / フレームワーク	説明
Dask cuML	ML を GPU で動作させるには、を使用します "cuML ライブラリ" Dask を使用して Rapids cuML パッケージにアクセスできます。Rapids cuML は、クラスタリング、寸法縮小、回帰アプローチなどの一般的な ML アルゴリズムを高性能 GPU ベースの実装で実装し、 CPU ベースのアプローチで最大 100 倍のスピードアップを実現します。
Dask cuDF	cuDF には、データのサブ設定、変換、ワンホットエンコーディングなど、 GPU アクセラレーションによる抽出、変換、読み込み（ ETL ）をサポートするその他のさまざまな機能があります。Rapids チームはを維持する "dask -cudf ライブラリ" これには、 Dask および cuDF を使用するためのヘルパーメソッドが含まれています。
Scikit learn	Scikit-Learn には、数十の機械学習アルゴリズムとモデルが組み込まれています。これらは、試算ツールと呼ばれます。各 "エスティメータ" は、を使用して一部のデータに装着できます "フィット" メソッド

2 つのノートブックを使用して、比較のための ML パイプラインを構築しました。 1 つは従来の Pandas の坐骨坐骨学習アプローチで、もう 1 つは Rapids および Dask との分散トレーニングです。各ノートブックを個別にテストして、パフォーマンスを時間と規模の観点から確認できます。各ノートブックについて個別に説明し、 Rapids および Dask を使用した分散型トレーニングの利点を示します。

Pandas で **Logs Day 15** をクリックして、 **scikit** に学習したランダムフォレストモデルをトレーニングします

このセクションでは、 Pandas と Dask DataFrames を使用して、 Criteo Terabyte データセットから Click Logs データをロードする方法について説明します。このユースケースは、広告交換のためのデジタル広告において、広告がクリックされるかどうかを予測したり、交換品が自動化されたパイプラインで正確なモデルを使用していないかどうか

を予測したりすることで、ユーザーのプロファイルを作成する場合に適しています。

Click Logs データセットから 15 日目のデータをロードし、合計 45GB にしました。Jupyter ノートブックの ctr-pandasrf-colated で次のセルを実行すると、最初の 5,000 万行を含む Pandas DataFrame が作成され、scikit 学習ランダムフォレストモデルが生成されます。

```
%%time
import pandas as pd
import numpy as np
header = ['col'+str(i) for i in range (1,41)] #note that according to
criteo, the first column in the dataset is Click Through (CT). Consist of
40 columns
first_row_taken = 50_000_000 # use this in pd.read_csv() if your compute
resource is limited.
# total number of rows in day15 is 20B
# take 50M rows
"""
Read data & display the following metrics:
1. Total number of rows per day
2. df loading time in the cluster
3. Train a random forest model
"""
df = pd.read_csv(file, nrows=first_row_taken, delimiter='\t',
names=header)
# take numerical columns
df_sliced = df.iloc[:, 0:14]
# split data into training and Y
Y = df_sliced.pop('col1') # first column is binary (click or not)
# change df_sliced data types & fillna
df_sliced = df_sliced.astype(np.float32).fillna(0)
from sklearn.ensemble import RandomForestClassifier
# Random Forest building parameters
# n_streams = 8 # optimization
max_depth = 10
n_bins = 16
n_trees = 10
rf_model = RandomForestClassifier(max_depth=max_depth,
n_estimators=n_trees)
rf_model.fit(df_sliced, Y)
```

トレーニングされたランダムフォレストモデルを使用して予測を実行するには、このノートブックで次の段落を実行します。重複を避けるために、 15 日目から最後の 100 万行をテストセットとして使用しました。セルはまた、モデルの発生率として定義された予測精度を計算し、ユーザーが広告をクリックするかどうかを正確に予測します。このノートブックの構成部品を確認するには、を参照してください "公式の坐骨神経痛 - 学習文書"。

```
# testing data, last 1M rows in day15
test_file = '/data/day_15_test'
with open(test_file) as g:
    print(g.readline())

# dataFrame processing for test data
test_df = pd.read_csv(test_file, delimiter='\t', names=header)
test_df_sliced = test_df.iloc[:, 0:14]
test_Y = test_df_sliced.pop('col1')
test_df_sliced = test_df_sliced.astype(np.float32).fillna(0)
# prediction & calculating error
pred_df = rf_model.predict(test_df_sliced)
from sklearn import metrics
# Model Accuracy
print("Accuracy:",metrics.accuracy_score(test_Y, pred_df))
```

Dask の 15 日目をロードし、 Dask cuML ランダムフォレストモデルをトレーニングします

前のセクションと同様に、 Pandas の Logs Day 15 をロードして、 scikit に学習したランダムフォレストモデルをトレーニングします。この例では、 Dask cuDF を使用して DataFrame のロードを実行し、 Dask cuML でランダムなフォレストモデルのトレーニングを行いました。セクションのトレーニング時間と規模の違いを比較しました "「トレーニング時間の比較」"

Crito_dASK_RF.ipynb

このノートブックは、次の例に示すように、「 numpy`, cuml` 」、および必要な「 Ask 」ライブラリをインポートします。

```
import cuml
from dask.distributed import Client, progress, wait
import dask_cudf
import numpy as np
import cudf
from cuml.dask.ensemble import RandomForestClassifier as cumlDaskRF
from cuml.dask.common import utils as dask_utils
```

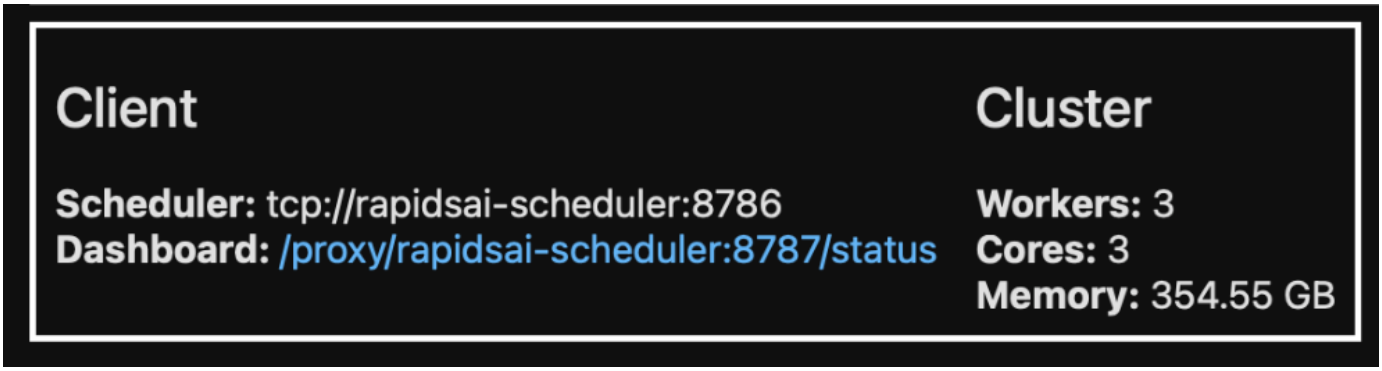
Dask Client() を開始します。

```
client = Client()
```

クラスタが正しく設定されていれば、ワーカーノードのステータスを確認できます。

```
client
workers = client.has_what().keys()
n_workers = len(workers)
n_streams = 8 # Performance optimization
```

AKS クラスタでは、次のステータスが表示されます。



Dask は遅延実行パラダイムを採用しています。処理コードを瞬時に実行するのではなく、 Dask は Directed Acyclic Graph （ DAG ）を実行します。DAG には、各ワーカーが実行する必要のある一連のタスクとそのやり取りが含まれています。このレイアウトは、ユーザーが Dask に一方の方法または別の方法で実行するように指示するまで、タスクが実行されないことを意味します。Dask を使用すると、 3 つの主なオプションがあります。

- * DataFrame 上のコールコンピュート () 。 * このコールはすべてのパーティションを処理し、結果をスケジューラに返して最終的な集約を行い、 cuDF DataFrame に変換します。このオプションは慎重に使用してください。スケジューラノードのメモリが不足しないかぎり、結果が大幅に低下する場合にのみ使用してください。
- * DataFrame 上で Persist() を呼び出します。 * この呼び出しはグラフを実行しますが、スケジューラノードに結果を返すのではなく、クラスタ内で結果をメモリに保持するため、ユーザは同じ処理を再実行することなくパイプライン内で中間結果を再利用できます。
- * DataFrame 上のコールヘッド () 。 * cuDF と同様に、この呼び出しは 10 件のレコードをスケジューラノードに返します。このオプションを使用すると、 DataFrame に目的の出力形式が含まれているかどうか、または処理と計算に応じてレコード自体が適切かどうかをすばやく確認できます。

したがって、ユーザがこれらのアクションのいずれかをコールしない限り、スケジューラが処理を開始するのを待機するアイドル状態になります。この遅延実行パラダイムは、 Apache Spark などの最新の並列および分散コンピューティングフレームワークで一般的です。

次の段落では、 Dask cuML を使用して、 GPU アクセラレーションによる分散コンピューティングを行い、モデル予測精度を計算することにより、ランダムなフォレストモデルのトレーニングを行います。

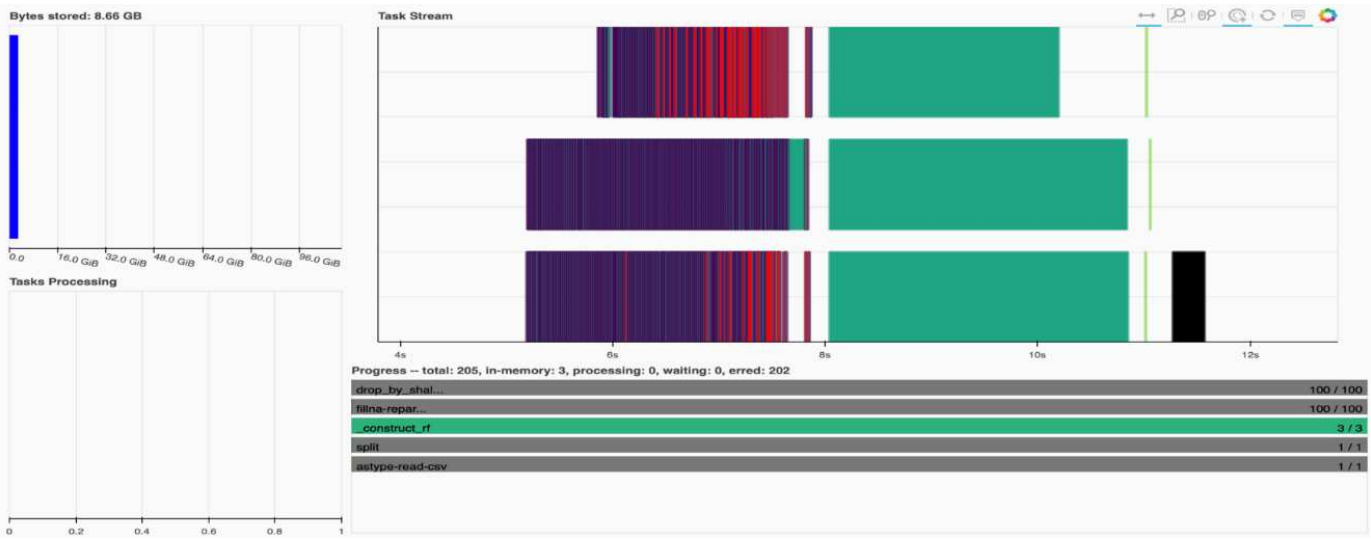
```
Adsf
# Random Forest building parameters
n_streams = 8 # optimization
max_depth = 10
n_bins = 16
n_trees = 10
cuml_model = cumlDaskRF(max_depth=max_depth, n_estimators=n_trees,
n_bins=n_bins, n_streams=n_streams, verbose=True, client=client)
cuml_model.fit(gdf_sliced_small, Y)
# Model prediction
pred_df = cuml_model.predict(gdf_test)
# calculate accuracy
cu_score = cuml.metrics.accuracy_score( test_y, pred_df )
```

ネイティブタスクストリームダッシュボードを使用して **Dask** を監視します

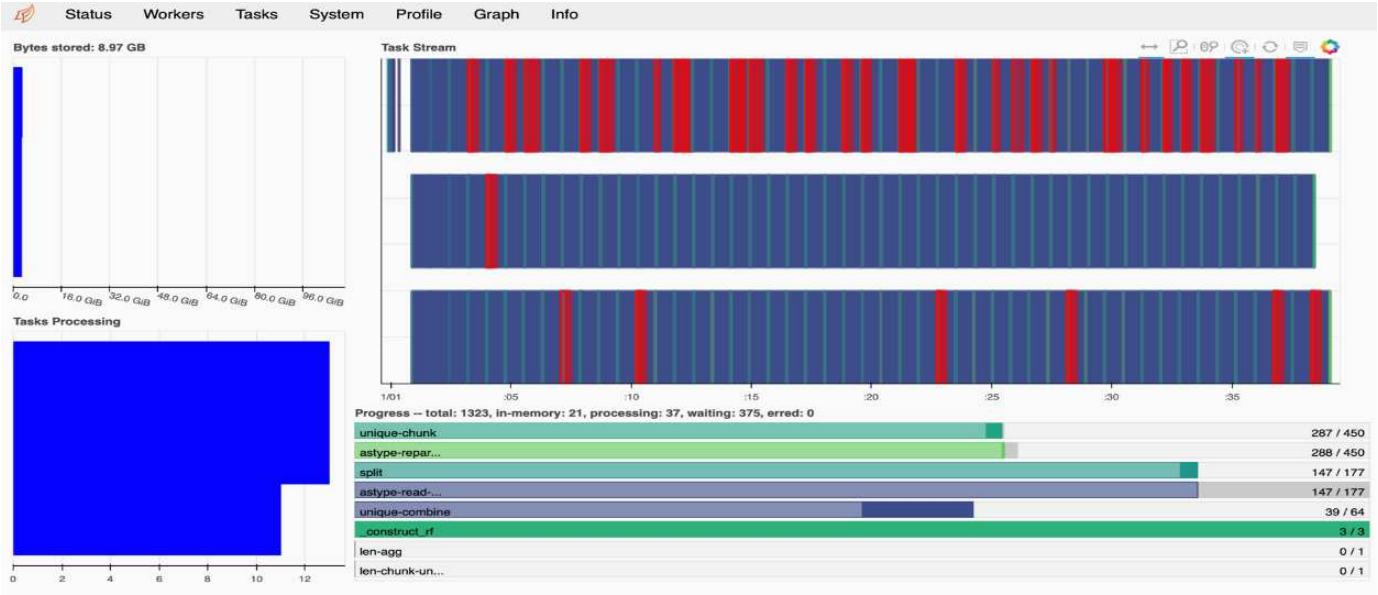
。 "Dask 分散スケジューラ" ライブフィードバックは、次の 2 つの形式で提供します。

- ライブ情報を含む多数のプロットやテーブルを含むインタラクティブなダッシュボード
- コンソールやノートブックでの対話型の使用に適したプログレスバーです

この場合、次の図は、保存されたバイト数、ストリーム数の詳細な内訳を示すタスクストリーム、実行された関連機能を持つタスク名ごとの進捗状況を監視する方法を示しています。この例では、ワーカーノードが 3 つあるため、ストリームには 3 つの主要なチャンクがあり、各ストリーム内で異なるタスクを示すカラーコードがあります。



個々のタスクを分析し、実行時間をミリ秒単位で調査するか、障害や障害を特定することができます。たとえば、次の図は、ランダムフォレストモデルフィッティングステージのタスクストリームを示しています。実行される関数は、 DataFrame 処理用の一意のチャンク、ランダムフォレストをフィッティングするための _construct_RF など、はるかに多くあります。Criteo のクリックログに含まれる 1 日分のデータのサイズ（ 45GB ）が大きいため、 DataFrame の処理にほとんどの時間が費やされていました。



トレーニング時間の比較

このセクションでは、従来の Pandas を使用したモデルのトレーニング時間を Dask と比較します。Pandas では、メモリオーバーフローを回避するために、処理時間が遅くなるため、より少量のデータをロードしました。そのため、結果を補間して公平な比較を行いました。

次の表は、 Pandas ランダムフォレストモデルに使用されるデータが大幅に少ない場合の、生のトレーニング時間の比較を示しています (データセットの 1 日あたりの 2000 億行のうち、 5,000 万行) 。このサンプルでは、使用可能なすべてのデータの 0.25% 未満しか使用されていません。DASK cuML の場合は '20 億行すべての使用可能なローについてランダムフォレストモデルをトレーニングしましたこの 2 つのアプローチでは、同等のトレーニング時間が得られました

アプローチ	トレーニング時間
Scikit - Learn ：トレーニングデータとして day15 の 50 M 行のみを使用します	47 分 21 秒
Rapids-DASK ：トレーニングデータとして、 Day15 のすべての 20B 行を使用します	1 時間 12 分 11 秒

次の表に示すように、トレーニング時間の結果を直線的に補間する場合、 Dask を使用した分散型トレーニングを使用すると大きな利点があります。従来の Pandas の scikit 学習アプローチでは、クリックログ 1 日あたり 45 GB のデータを処理してトレーニングするのに 13 日かかりますが、 Rapids-Dask アプローチでは同じ量のデータを処理するのにかかる時間は 262.39 倍になります。

アプローチ	トレーニング時間
Scikit - Learn ：トレーニングデータとして day15 のすべての 20B 行を使用します	13 日、 3 時間、 40 分、 11 秒
Rapids-DASK ：トレーニングデータとして、 Day15 のすべての 20B 行を使用します	1 時間 12 分 11 秒

前の表では、 Dask と Rapids を使用してデータ処理とモデルトレーニングを複数の GPU インスタンスに分散することで、従来の Pandas DataFrame 処理と比較して、 scikit 学習モデルトレーニングでの実行時間が大

幅に短縮されたことを確認できます。このフレームワークを使用すると、マルチノードのマルチ GPU クラスタ内だけでなく、クラウド内でもオンプレミスでのスケールアップとスケールアウトが可能です。

Prometheus と **Grafana** で **Dask** と **Rapids** を監視します

すべてのデータを導入したら、新しいデータに対して推論を実行します。このモデルは、ユーザーが閲覧アクティビティに基づいて広告をクリックするかどうかを予測します。予測の結果は Dask cuDF に格納されます。Prometheus で結果を監視し、 Grafana ダッシュボードで視覚化できます。

詳細については、を参照してください "Rapids AI 培地ポスト"。

NetApp DataOps ツールキットを使用したデータセットとモデルのバージョン管理

NetApp DataOps Toolkit for Kubernetes は、ストレージリソースと Kubernetes ワークロードをデータサイエンスのワークスペースレベルまで抽象化します。これらの機能は、データサイエンティストとデータエンジニア向けに設計された、使いやすいシンプルなインターフェイスにパッケージ化されています。使い慣れた Python プログラムを使用しており、データサイエンティストやエンジニアは JupyterLab ワークスペースをわずか数秒でプロビジョニングおよび削除できます。これらのワークスペースには、テラバイト、あるいはペタバイト規模のストレージ容量が含まれることがあり、データサイエンティストは、すべてのトレーニングデータセットをプロジェクトのワークスペースに直接格納できます。ワークスペースとデータボリュームを個別に管理する時代は終わりました。

詳細については、ツールキットを参照してください "GitHub リポジトリ"。

Jupyter ノートブックを参考にしてください

このテクニカルレポートには、 Jupyter ノートブックが 2 つ関連付けられています。

- "*CTR - PandasRF - 照合済み。 ipynb. *" このノートブックは Crito Terabyte Logs データセットから 15 日目を読み込み、データを Pandas DataFrame に処理してフォーマットし、 Scikit-learn ランダムフォレストモデルのトレーニングを行い、予測を実行し、精度を計算します。
- "* Crito_dAsk _RF.ipynb.*" このノートブックは Crito Terabyte Logs データセットから 15 日目をロードし、データを Dask cuDF に処理してフォーマットし、 Dask cuML ランダムフォレストモデルのトレーニングを行い、予測を実行し、精度を計算します。GPU を搭載した複数のワーカーノードを活用することで、この分散データとモデルの処理とトレーニングのアプローチを非常に効率的に行うことができます。処理するデータが多いほど、従来の ML アプローチに比べて時間を大幅に節約できます。このノートブックは、ネットワークセットアップによってデータやモデルの配布が自由に移動できる限り、クラウド、オンプレミス、または Kubernetes クラスタにコンピューティングとストレージが異なる場所に配置されているハイブリッド環境に導入できます。

まとめ

Azure NetApp Files 、 Rapids 、 Dask は、 Docker や Kubernetes などのオーケストレーションツールと統合することで、大規模な ML 処理とトレーニングの導入を高速化し、簡易化します。エンドツーエンドのデータパイプラインを統合する解決策ことで、

多くの高度なコンピューティングワークロードに特有のレイテンシと複雑さを軽減し、開発と運用のギャップを効果的に解消します。データサイエンティストは、大規模なデータセットでクエリを実行し、トレーニングフェーズ中にデータやアルゴリズムのモデルを他のユーザーと安全に共有できます。

独自の AI / ML パイプラインを構築する場合、アーキテクチャ内のコンポーネントの統合、管理、セキュリティ、およびアクセス性の設定は困難な作業です。開発者に環境へのアクセスと管理を許可することには、もう 1 つの課題があります。

クラウドにエンドツーエンドの分散トレーニングモデルとデータパイプラインを構築することで、 GPU によって高速化されたデータ処理フレームワークやコンピューティングフレームワークを活用していない従来のオープンソースアプローチと比較して、ワークフロー全体の完了時間が 2 桁向上することを実証しました。

ネットアップ、 Microsoft 、オープンソースのオーケストレーションフレームワーク、 NVIDIA を組み合わせることで、最新テクノロジをマネージドサービスとして統合し、優れた柔軟性を実現してテクノロジの採用を促進し、新しい AI / ML アプリケーションの市場投入期間を短縮できます。これらの高度なサービスはクラウドネイティブ環境で提供され、オンプレミス環境やハイブリッド導入アーキテクチャで簡単に移行できます。

追加情報の参照先

このドキュメントに記載されている情報の詳細については、次のリソースを参照してください。

- Azure NetApp Files の特長
  - Azure NetApp Files のソリューションアーキテクチャのページです

    "https://docs.microsoft.com/azure/azure-netapp-files/azure-netapp-files-solution-architectures"
- コンテナ向けの Trident 永続的ストレージ：
  - Azure NetApp Files と Trident

    "https://netapptrident.readthedocs.io/en/stablev20.07/kubernetes/operations/tasks/backends/anf.html"
- Dask および Rapids ：
  - Dask

    "https://docs.dask.org/en/latest/"
  - Dask をインストールします

    "https://docs.dask.org/en/latest/install.html"
  - Dask API

    "https://docs.dask.org/en/latest/api.html"
  - Dask Machine Learning の略

    "https://examples.dask.org/machine-learning.html"
  - Dask Distributed Diagnostics の実行

"https://docs.dask.org/en/latest/diagnostics-distributed.html"

- ML フレームワークとツール：
  - TensorFlow ：あらゆる環境に対応するオープンソースの機械学習フレームワーク

    "https://www.tensorflow.org/"

  - Docker です

    "https://docs.docker.com"

  - Kubernetes

    "https://kubernetes.io/docs/home/"

  - クビフロー

    "http://www.kubeflow.org/"

  - Jupyter Notebook Server の 2 つのツールを使用

    "http://www.jupyter.org/"

TR-4886 ：『 Distributed training in Azure ： Lane detection - 解決策 design 』

Muneer Ahmad 氏と Verron Martina 氏、ネットアップの Ronen Dar 、 RUN ： AAI

2019 年 5 月より、 Microsoft は Azure ネイティブのファーストパーティポータルサービスを提供し、 NetApp ONTAP テクノロジを基盤とするエンタープライズ NFS および SMB ファイルサービスを提供しています。この開発は、 Microsoft とネットアップの戦略的パートナーシップによって推進されており、ワールドクラスの ONTAP データサービスの Azure への対応範囲がさらに拡大しています。

業界をリードするクラウドデータサービスプロバイダであるネットアップは、 AI インフラを仮想化する企業である AI の運用を共同で開始し、 GPU 利用率を最大限に活用して AI の実験を高速化しました。このパートナーシップでは、多数の実験を並行して実行し、データへの高速アクセスを実現し、無限のコンピューティングリソースを活用することで、 AI を加速できます。Run ： AI は、リソースの割り当てを自動化することで GPU 利用率を最大限に高めます。実績のある Azure NetApp Files のアーキテクチャにより、データパイプラインに障害が生じることをなくし、あらゆる実験を最高の速度で実行できます。

ネットアップと RUN ： AI が力を合わせて、お客様に Azure で AI 導入を実現するための将来を見据えたプラットフォームを提供しています。分析やハイパフォーマンスコンピューティング（ HPC ）から自律判断（お客様は必要なときに必要なものだけを購入することで IT 投資を最適化できる）まで、ネットアップと RUN のアライアンスによって、 AI は Azure クラウドでの単一の統合エクスペリエンスを提供します。

解決策の概要

このアーキテクチャでは、 AI や機械学習（ ML ）の分散型トレーニングプロセスであるレーン検出において、最も演算負荷の高い部分に焦点が当てられています。車線検知は、自動運転で最も重要な作業の 1 つであり、車線区分線の位置を特定することで車両

を誘導するのに役立ちます。車線標示などの静的コンポーネントは、車両を高速道路でインタラクティブかつ安全に走行させる。

畳み込みニューラルネットワーク（ CNN ）ベースのアプローチでは、シーンの理解とセグメント化が新たなレベルにまで押しつけられています。長い構造やリージョンが含まれているオブジェクト（ポール、車線の陰など）は適切に機能しませんが、空間的畳み込みニューラルネットワーク（ SCNN ）は、 CNN を豊かな空間レベルに一般化します。同一層のニューロン間で情報を伝播できるため、車線、ポール、トラックなどの構造化された物体（オ結論を含む）に最適です。この互換性は、空間情報を強化し、滑らかさと連続性を維持できるためです。

モデルがデータセット内のさまざまなコンポーネントを学習し、区別できるように、数千ものシーンイメージをシステムに挿入する必要があります。これらのイメージは天候、日中か夜、マルチレーンハイウェーの道および他の交通条件を含んでいる。

トレーニングには、質の高いデータと量のニーズがあります。1 つの GPU または複数の GPU でトレーニングを完了するには、数日から数週間かかることがあります。データ分散トレーニングは、マルチノードの GPU を複数使用することでプロセスを高速化できます。Horovod は、分散トレーニングを提供する一方で、GPU のクラスタ間でデータを読み取ることは障害となる可能性があるフレームワークの 1 つです。Azure NetApp Files は、超高速、高スループット、一貫した低レイテンシを実現し、スケールアウト / スケールアップ機能を提供して、 GPU がコンピューティング容量の最適な値に活用されるようにします。当社の実験では、 SCNN を使用してレーン検出をトレーニングするために、クラスタ全体のすべての GPU が平均で 96% 以上使用されていることが確認されました。

対象読者

データサイエンスには、 IT とビジネスに関する複数の分野が組み込まれているため、ターゲットを絞ったオーディエンスには複数のペルソナが含まれます。

- データサイエンティストは、選択したツールとライブラリを柔軟に使用する必要があります。
- データエンジニアは、データフローの仕組みと、データが格納されている場所を把握する必要があります。
- 自動運転のユースケースエキスパート。
- クラウド（ Azure ）リソースのセットアップと管理を担当するクラウド管理者およびアーキテクト。
- DevOps エンジニアは、新しい AI / ML アプリケーションを継続的統合 / 継続的導入（ CI / CD ）パイプラインに統合するためのツールを必要としています。
- ビジネスユーザは、 AI / ML アプリケーションにアクセスしたいと考えています。

このドキュメントでは、 Azure NetApp Files 、 Run ： AI 、 Microsoft Azure の 3 つの役割がそれぞれビジネスにもたらす価値について説明します。

解決策テクノロジ

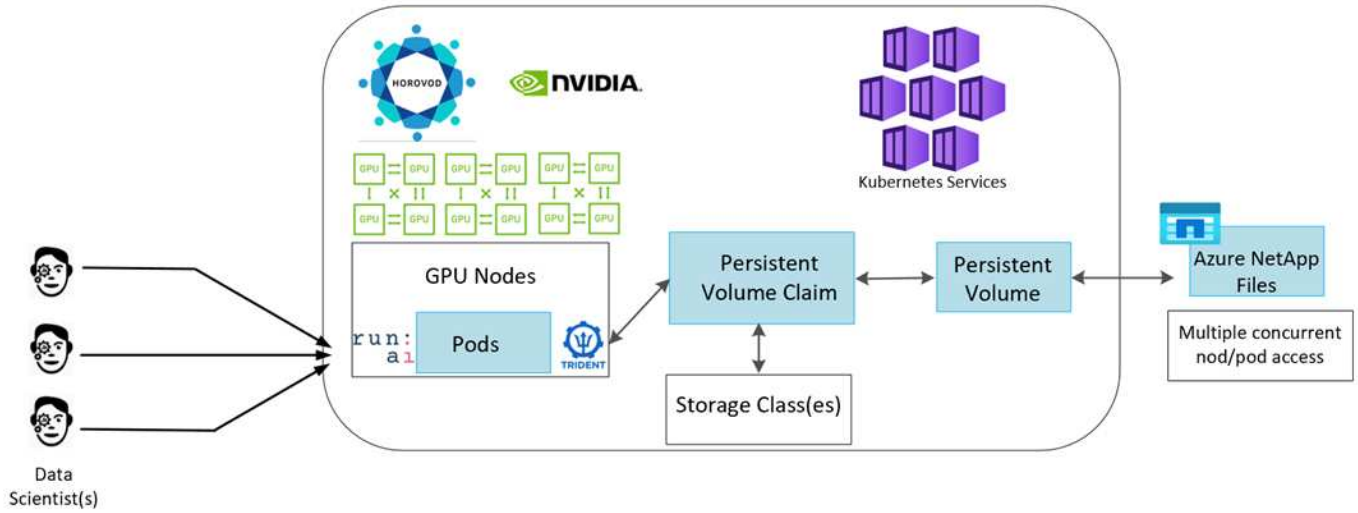
このセクションでは、 Azure クラウドで完全に稼働する規模の分散トレーニング解決策を実装することで、レーン検出のユースケースに必要なテクノロジについて説明します。次の図は、解決策アーキテクチャの概要を示しています。

この解決策で使用される要素は次のとおりです。

- Azure Kubernetes Service （ AKS ）
- NVIDIA GPU を搭載した Azure コンピューティング SKU

- Azure NetApp Files の特長
- 実行： AI
- NetApp Trident

ここに記載されているすべての要素へのリンクをに示します "追加情報" セクション。



クラウドリソースとサービスの要件

次の表に、解決策の実装に必要なハードウェアコンポーネントを示します。解決策の実装で使用されるクラウドコンポーネントは、お客様の要件に応じて異なる場合があります。

クラウド	数量
AK	少なくとも 3 つのシステムノードと 3 つの GPU ワーカーノードが必要です
仮想マシン（ VM ） SKU システムノード	3 つの Standard_DS2_v2
VM SKU GPU ワーカーノード	3 つの Standard_NC6s_v3
Azure NetApp Files の特長	4TB の標準ティア

ソフトウェア要件

次の表に、解決策の実装に必要なソフトウェアコンポーネントを示します。解決策の実装で使用されるソフトウェアコンポーネントは、お客様の要件に応じて異なる場合があります。

ソフトウェア	バージョンまたはその他の情報
AK - Kubernetes バージョン	1.18.14
AI CLI を実行	v2.2.25
実行： AI Orchestration Kubernetes Operator バージョン	1.0.109
ホロボド	0.21.2
NetApp Trident	20.01.1

ソフトウェア	バージョンまたはその他の情報
Helm	3.0.0

レーン検出–実行による分散トレーニング： **AI**

このセクションでは、ランオーケストレーションツール AI を使用して大規模なレーン検出分散トレーニングを実行するためのプラットフォームの設定について詳しく説明します。ここでは、すべての解決策要素のインストールと、前述のプラットフォームでの配布トレーニングジョブの実行について説明します。ML のバージョン管理には、 NetApp SnapshotTM を使用し、 RUN ： AI の実験によってデータとモデルの再現性を達成しました。ML のバージョン管理は、モデルの追跡、チームメンバー間での作業の共有、結果の再現性、生産への新しいモデルバージョンのローリング、データソースの作成に重要な役割を果たします。ネットアップの ML バージョン管理（ Snapshot ）は、各実験に関連するデータ、トレーニング済みモデル、ログのポイントインタイムバージョンをキャプチャできます。豊富な API サポートにより、実行中の AI プラットフォームとの統合が容易になります。トレーニングの状態に基づいてイベントをトリガーするだけで済みます。また、コードや Kubernetes （ Kubernetes ）上で実行されているコンテナに何も変更を加えずに、実験全体の状態をキャプチャする必要もあります。

最後に、このテクニカルレポートは、 AKS を介して複数の GPU 対応ノードでパフォーマンスを評価する方法をラップアップします。

TuSimple データセットを使用したレーン検出のユースケースに関する分散トレーニング

このテクニカルレポートでは、レーン検出用の TuSimple データセットに対して分散トレーニングを実行します。Horovod は、 AKS を使用して Kubernetes クラスタ内の複数の GPU ノードでデータ分散トレーニングを同時に実施するためのトレーニングコードで使用されます。コードは、 TuSimple データのダウンロードおよび処理用のコンテナイメージとしてパッケージされています。処理されたデータは、 NetApp Trident プラグインによって割り当てられた永続ボリュームに格納されます。トレーニングでは、 1 つ以上のコンテナイメージが作成され、データのダウンロード時に作成された永続ボリュームに格納されたデータが使用されます。

データとトレーニングのジョブを送信するには、 run ： AI を使用してリソースの割り当てと管理をオーケストレーションします。Run ： AI では、 Horovod に必要な Message Passing Interface （ MPI ；メッセージパッシングインターフェイス）処理を実行できます。このレイアウトでは、複数の GPU ノードが相互に通信して、各トレーニング mini バッチの実行後にトレーニングの重みを更新できます。また、 UI や CLI からトレーニングを監視できるため、実験の進捗状況を簡単に監視できます。

NetApp Snapshot はトレーニングコードに統合されており、あらゆる実験に対応するデータの状態とトレーニング済みモデルをキャプチャします。この機能を使用すると、使用するデータとコードのバージョン、および生成された関連するトレーニング済みモデルを追跡できます。

AK のセットアップとインストール

AKS クラスタのセットアップとインストールは、に進みます "AKS クラスタを作成します"。次に、次の一連の手順を実行します。

1. ノードのタイプ（システム（ CPU ）ノードまたはワーカー（ GPU ）ノードのいずれであるか）を選択するときは、次のいずれかを選択します。

a. 「 agentpool 」という名前のプライマリ・システム・ノードを 'Standard _ DS2_v2' サイズに追加しますデフォルトの 3 つのノードを使用します。

b. 'Standard_NC6s_v3' プール・サイズを使用して 'Worker ノード gpupool' を追加しますGPU ノードには最小 3 ノードを使用します。

導入には 5 〜 10 分かかります。

2. 導入が完了したら、 Connect to Cluster （クラスタへの接続）をクリックします。新しく作成した AKS クラスタに接続するには、ローカル環境（ラップトップ / PC ）から Kubernetes コマンドラインツールをインストールします。にアクセスします "ツールをインストールします" OS に応じてインストールします。

3. "ローカル環境に Azure CLI をインストールします"。

4. 端末から AKS クラスタにアクセスするには、まず「 AZ login 」と入力し、クレデンシャルを入力します。

5. 次の 2 つのコマンドを実行します。

```
az account set --subscription xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx
aks get-credentials --resource-group resourcegroup --name aksclustername
```

6. Azure CLI で、次のコマンドを入力します。

```
kubectl get nodes
```

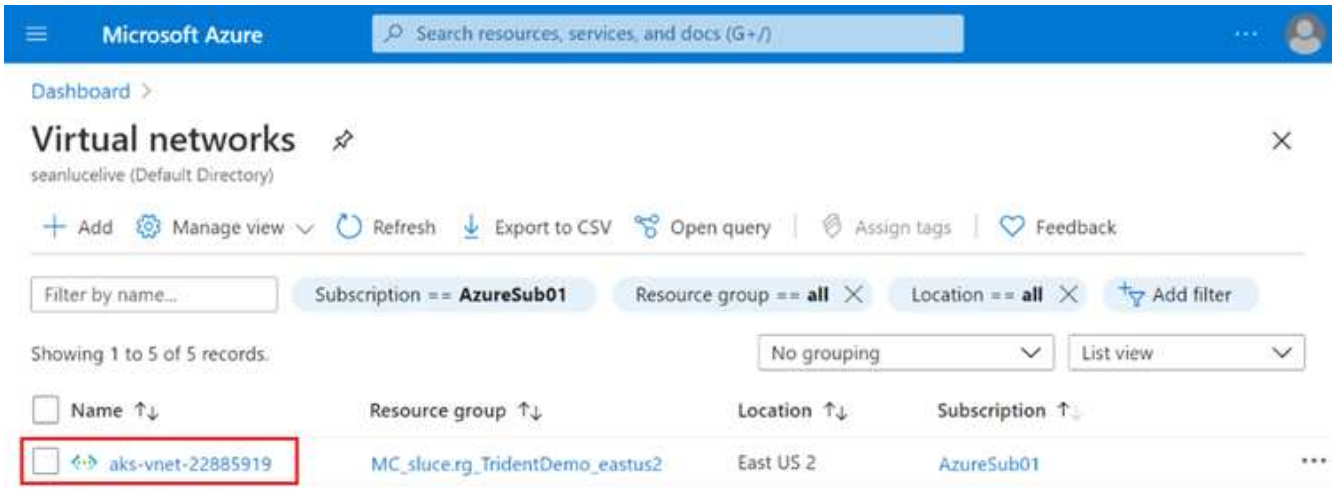
ここで示したように 6 つのノードがすべて稼働していれば、 AKS クラスタをローカル環境に接続することができます。

```
verronmartina@verron-mac-0 ~ % kubectl get nodes
NAME                                STATUS   ROLES   AGE   VERSION
aks-agentpool-34613062-vmss000000   Ready    agent   22m   v1.18.14
aks-agentpool-34613062-vmss000001   Ready    agent   22m   v1.18.14
aks-agentpool-34613062-vmss000002   Ready    agent   22m   v1.18.14
aks-gpupool-34613062-vmss000000     Ready    agent   20m   v1.18.14
aks-gpupool-34613062-vmss000001     Ready    agent   20m   v1.18.14
aks-gpupool-34613062-vmss000002     Ready    agent   20m   v1.18.14
verronmartina@verron-mac-0 ~ %
```

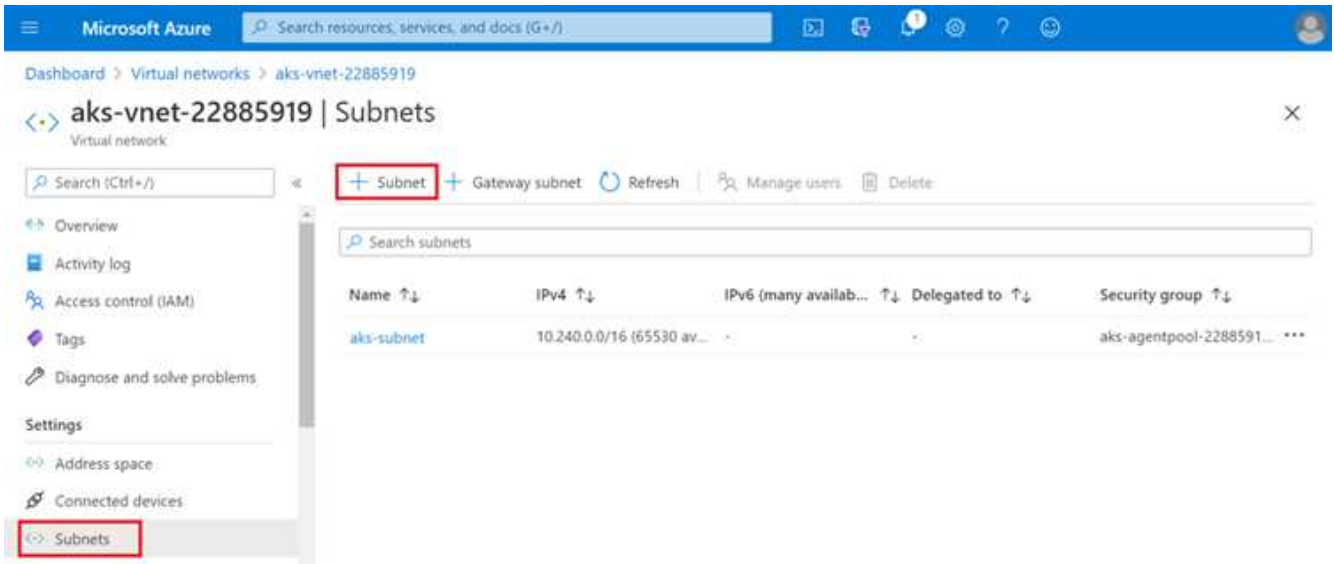
Azure NetApp Files の委譲されたサブネットを作成します

Azure NetApp Files の委任されたサブネットを作成するには、次の手順を実行します。

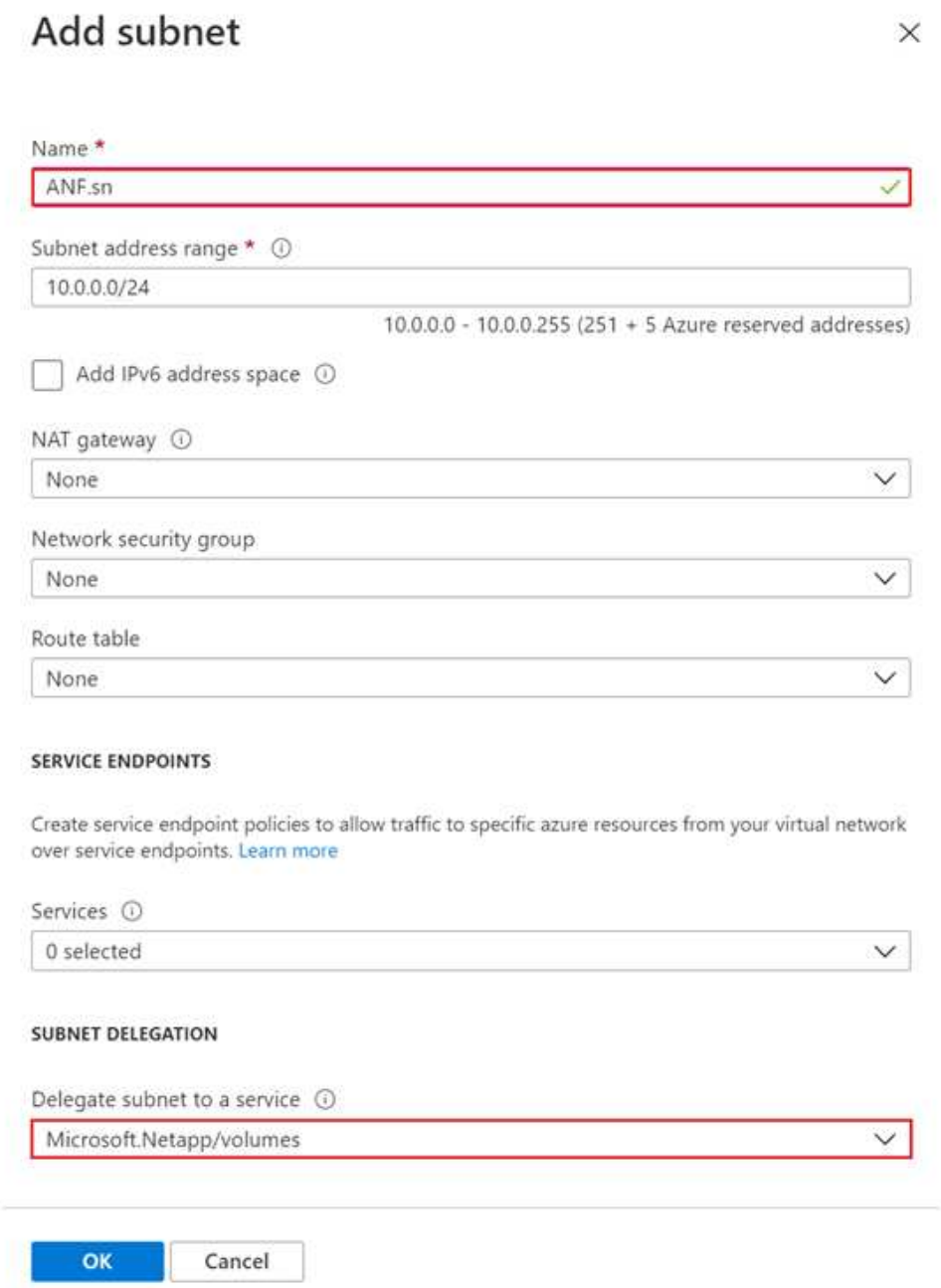
1. Azure ポータル内の仮想ネットワークに移動します。新しく作成した仮想ネットワークを検索します。この例では、 AKs-vnet などのプレフィックスが必要です。仮想ネットワークの名前をクリックします。



2. [サブネット] をクリックし、上部のツールバーから [サブネット +] を選択します。



3. サブネットに「 ANF 」などの名前を付け、サブネットの委任の見出しの下で、Microsoft.NetApp/volumes を選択します。他のものは変更しないでください。[OK] をクリックします。



Azure NetApp Files ボリュームはアプリケーションクラスタに割り当てられ、 Kubernetes で永続ボリューム要求（ PVC ）として使用されます。この割り当てにより、ボリュームをさまざまなサービスに柔軟にマッピングし、 Jupyter ノートブック PC やサーバーレス関数などに対応することができます

サービスのユーザは、プラットフォームのストレージをさまざまな方法で消費できます。Azure NetApp Files の主な利点は次のとおりです。

- スナップショットを使用できるようになります。
- Azure NetApp Files ボリュームに大量のデータを格納できます。
- Azure NetApp Files ボリュームのモデルを大量のファイルセットで実行する場合は、そのモデルのパフォーマンス向上が必要になります。

Azure NetApp Files セットアップ

Azure NetApp Files のセットアップを完了するには、まず、の説明に従って を設定する必要があります "クイックスタート： Azure NetApp Files をセットアップし、 NFS ボリュームを作成します"。

ただし、 Trident からボリュームを作成するため、 Azure NetApp Files 用の NFS ボリュームを作成する手順は省略できます。続行する前に、次のものがあることを確認してください。

1. "Azure NetApp Files とネットアップのリソースプロバイダに登録（ Azure Cloud Shell を使用）"。
2. "Azure NetApp Files でアカウントを作成"。
3. "容量プールをセットアップする" （必要に応じて、 4TiB Standard または Premium 以上）。

AKS 仮想ネットワークおよび Azure NetApp Files 仮想ネットワークのピアリング

次に、次の手順に従って、 Azure NetApp Files VNet とともに AKS 仮想ネットワーク（ VNet ）のピア関係を設定します。

1. Azure ポータル上部の検索ボックスに「 virtual networks 」と入力します。
2. vnet AK - vnet-name をクリックして、検索フィールドにピアを入力します。
3. + Add をクリックして、次の表に示す情報を入力します。

フィールド	**Value** または概要のいずれかです
ピアリングリンク名	AKs-vnet-name_-to-anf
サブスクリプション ID	ピアリング先の Azure NetApp Files VNet のサブスクリプション
VNet ピアリングパートナー	Azure NetApp Files VNet の略

> デフォルトでは、アスタリスク以外のすべてのセクションはそのままにしておきます

4. [Add] または [OK] をクリックして、仮想ネットワークにピアリングを追加します。

詳細については、を参照してください "仮想ネットワークピアリングを作成、変更、削除します"。

Trident

Trident は、アプリケーションコンテナの永続的ストレージ向けにネットアップが管理しているオープンソースプロジェクトです。Trident は、ポッドとして実行される外部プロビジョニングコントローラとして実装され、ボリュームを監視し、プロビジョニングプロセスを完全に自動化します。

NetApp Trident では、トレーニングデータセットとトレーニング済みモデルを格納する永続的ボリュームを作成して接続することで、 Kubernetes との円滑な統合が可能です。データサイエンティストやデータエンジニアは、データセットを手動で保存して管理する手間をかけることなく、 Kubernetes クラスタを簡単に使用できます。Trident では、論理的な API 統合を通じてデータ管理関連のタスクが統合されるため、データサイエンティストは新しいデータプラットフォームの管理を習得する必要もありません。

Trident をインストール

Trident ソフトウェアをインストールするには、次の手順を実行します。

1. "最初に Helm をインストールします"。
2. Trident 21.01.1 インストーラをダウンロードして展開します。

```
wget
https://github.com/NetApp/trident/releases/download/v21.01.1/trident-
installer-21.01.1.tar.gz
tar -xf trident-installer-21.01.1.tar.gz
```

3. ディレクトリを 'trident-installer' に変更します

```
cd trident-installer
```

4. tridentctl' をシステムの $path.` のディレクトリにコピーします

```
cp ./tridentctl /usr/local/bin
```

5. Helm を使用して Kubernetes クラスタに Trident をインストールします。
   a. ディレクトリを Helm ディレクトリに変更します。

   ```
   cd helm
   ```

   b. Trident をインストール

   ```
   helm install trident trident-operator-21.01.1.tgz --namespace trident
   --create-namespace
   ```

   c. Trident ポッドのステータスを通常の Kubernetes クラスタの方法で確認します。

   ```
   kubectl -n trident get pods
   ```

   d. すべてのポッドが稼働中の場合は、 Trident がインストールされているので移行を推奨します。

Azure NetApp Files のバックエンドとストレージクラスをセットアップする

Azure NetApp Files バックエンドとストレージクラスをセットアップするには、次の手順を実行します。

1. ホームディレクトリに切り替えます。

```
cd ~
```

2. をクローニングします "プロジェクトリポジトリ" lane -detection -SCNN-horovod`
3. 'trident-config' ディレクトリに移動します

```
cd ./lane-detection-SCNN-horovod/trident-config
```

4. Azure サービスの原則を作成します（サービスの原則は、 Trident が Azure と通信して Azure NetApp Files リソースにアクセスする方法です）。

```
az ad sp create-for-rbac --name
```

出力は次の例のようになります。

```
{
  "appId": "xxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
   "displayName": "netapptrident",
    "name": "http://netapptrident",
    "password": "xxxxxxxxxxxxxxx.xxxxxxxxxxxxxx",
    "tenant": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx"
 }
```

5. Trident のバックエンド JSON ファイルを作成します。
6. 任意のテキストエディタを使用して 'anf-backend.json ファイル内の下の表の次のフィールドに入力します

フィールド	価値
サブスクリプション ID	お客様の Azure サブスクリプション ID
tenantID のこと	Azure テナント ID （前の手順での AZ AD SP の出力から取得）
ClientID	自分の appID （前のステップでの AZ 広告 SP の出力から）
clientSecret	パスワード（前の手順での AZ AD SP の出力からの）

ファイルは次の例のようになります。

```
{
    "version": 1,
    "storageDriverName": "azure-netapp-files",
    "subscriptionID": "fakec765-4774-fake-ae98-a721add4fake",
    "tenantID": "fakef836-edc1-fake-bff9-b2d865eefake",
    "clientID": "fake0f63-bf8e-fake-8076-8de91e57fake",
    "clientSecret": "SECRET",
    "location": "westeurope",
    "serviceLevel": "Standard",
    "virtualNetwork": "anf-vnet",
    "subnet": "default",
    "nfsMountOptions": "vers=3,proto=tcp",
    "limitVolumeSize": "500Gi",
    "defaults": {
    "exportRule": "0.0.0.0/0",
    "size": "200Gi"
}
```

7. 構成ファイルとして 'anf-backend.json を使用して 'trident' 名前空間に Azure NetApp Files バックエンドを作成するように Trident に指示します

```
tridentctl create backend -f anf-backend.json -n trident
```

8. ストレージクラスを作成します。

   a. k8 ユーザは、ストレージクラスを名前で指定する PVC を使用してボリュームをプロビジョニングします。次のコマンドを使用して ' 前の手順で作成した Azure NetApp Files バックエンドを参照するストレージ・クラス 'azurenetappfiles' を作成するよう 'Kubernetes クラスタに指示します

   ```
   kubectl create -f anf-storage-class.yaml
   ```

   b. 次のコマンドを使用して、ストレージクラスが作成されたことを確認します。

   ```
   kubectl get sc azurenetappfiles
   ```

   出力は次の例のようになります。

```
NAME               PROVISIONER             RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
azurenetappfiles   csi.trident.netapp.io   Delete          Immediate           false                  98s
```

ボリューム **Snapshot** コンポーネントを **AKS** に導入してセットアップします

適切なボリューム Snapshot コンポーネントがあらかじめクラスタにインストールされていない場合は、次の手順を実行して、これらのコンポーネントを手動でインストールできます。

AK 1.18.14 には Snapshot コントローラが事前にインストールされていません。

1. 次のコマンドを使用して、スナップショットベータ版の CRD をインストールします。

```
kubectl create -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
3.0/client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl create -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
3.0/client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.yam
l
kubectl create -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
3.0/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

2. GitHub の次のドキュメントを使用して、 Snapshot Controller をインストールします。

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-3.0/deploy/kubernetes/snapshot-
controller/rbac-snapshot-controller.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-3.0/deploy/kubernetes/snapshot-
controller/setup-snapshot-controller.yaml
```

3. ボリュームスナップショットを作成する前に 'K8s'volumesnapshotclass' を設定します "ボリューム Snapshot クラス" セットアップが完了している必要があります。Azure NetApp Files のボリューム Snapshot クラスを作成し、ネットアップの Snapshot テクノロジを使用して ML のバージョン管理を実現します。volumesnapshotclass NetApp-csi-snapclass' を作成し ' 次のようにデフォルトの ` volumesnapshotclass 」に設定します

```
kubectl create -f netapp-volume-snapshot-class.yaml
```

出力は次の例のようになります。

```
volumesnapshotclass.snapshot.storage.k8s.io/netapp-csi-snapclass created
```

4. 次のコマンドを使用して、ボリュームの Snapshot コピークラスが作成されたことを確認します。

```
kubectl get volumesnapshotclass
```

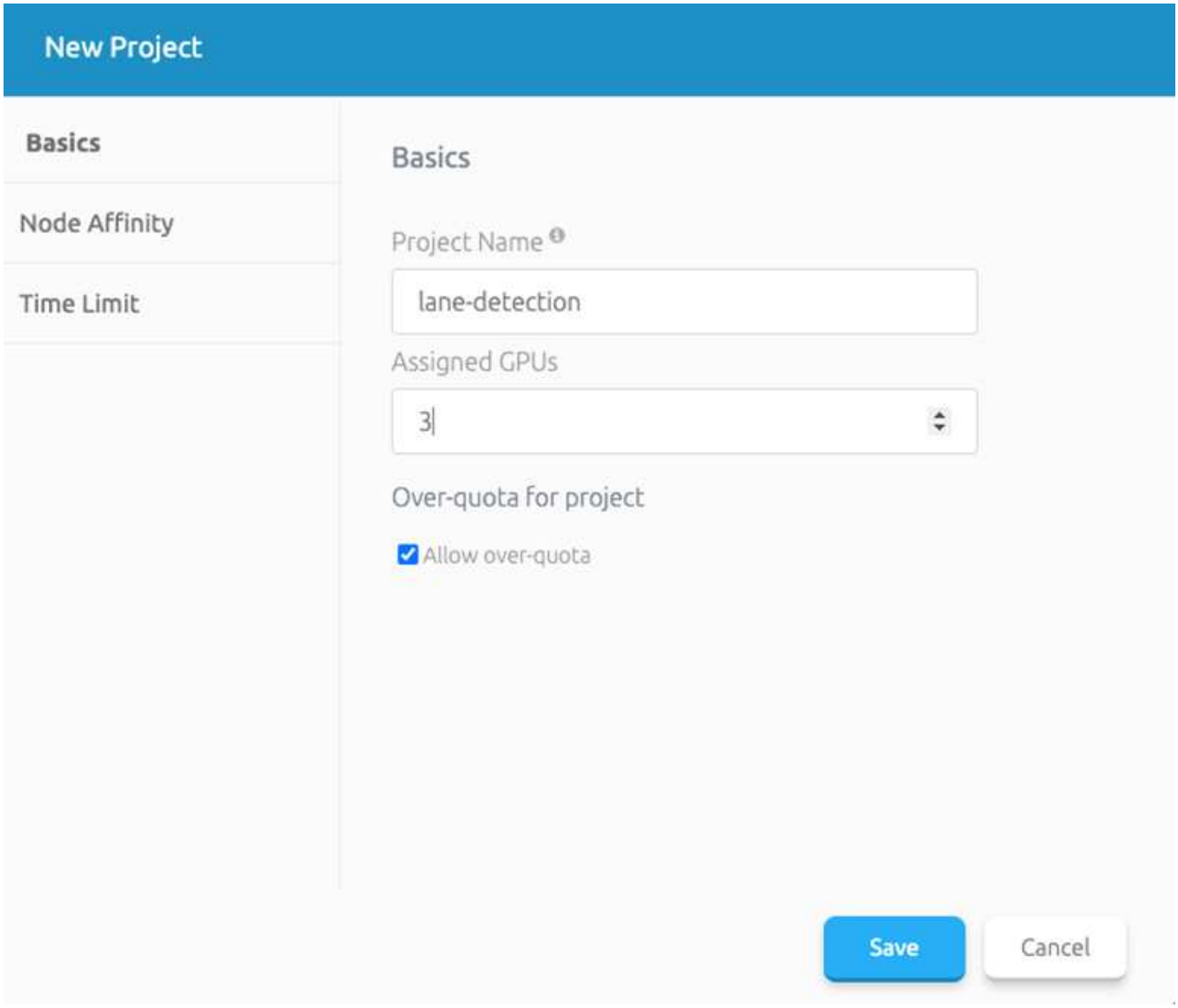
出力は次の例のようになります。

```
NAME                    DRIVER                  DELETIONPOLICY   AGE
netapp-csi-snapclass    csi.trident.netapp.io   Delete           63s
```

「 **AI Installation** 」を実行します

Run ： AI をインストールするには、次の手順を実行します。

1. "Run ： AI クラスタを AKS にインストールします"。
2. app.runai.ai にアクセスし、 [新しいプロジェクトの作成] をクリックして、レーン検出という名前を付けます。'runai' で始まる名前空間を Kubernetes クラスタに作成し ' そのあとにプロジェクト名を付けますこの場合、作成される名前空間は runai-lane detection になります。



3. "インストール実行： AI CLI"。
4. ターミナルで、次のコマンドを使用して、 LANE 検出をデフォルトの実行として AI プロジェクトに設定します。

```
`runai config project lane-detection`
```

出力は次の例のようになります。

```
Project lane-detection has been set as default project
```

5. Create ClusterRole and ClusterRoleBinding for the project namespace (`lane detection など) 」という名前空間に属するデフォルトのサービスアカウントには ' ジョブの実行中に "volumeSnapshot" 操作を実行する権限があります

   a. 次のコマンドを使用して、名前空間を一覧表示し、「 runai-lane -detection 」が存在することを確認します。

   ```
   kubectl get namespaces
   ```

   次のような出力が表示されます。

```
NAME                    STATUS   AGE
default                 Active   130m
kube-node-lease         Active   130m
kube-public             Active   130m
kube-system             Active   130m
runai                   Active   4m44s
runai-lane-detection    Active   13s
trident                 Active   102m
```

6. 次のコマンドを使用して、 ClusterRole 「 netappsnapshot 」および ClusterRoleBinding 「 netappsnapshot 」を作成します。

```
`kubectl create -f runai-project-snap-role.yaml`
`kubectl create -f runai-project-snap-role-binding.yaml`
```

TuSimple データセットを実行時の AI ジョブとしてダウンロードして処理します

TuSimple データセットを実行としてダウンロードして処理するプロセス。 AI ジョブはオプションです。このプロセスでは、次の手順を実行します。

1. Docker イメージをビルドしてプッシュするか、既存の Docker イメージを使用する場合は、この手順を省略します（「 m uneer7589/download-tusimple:1.0 」など）

   a. ホームディレクトリに移動します。

   ```
   cd ~
   ```

b. プロジェクト「 lane detection - SCNN-horovod` のデータディレクトリに移動します。

```
cd ./lane-detection-SCNN-horovod/data
```

c. 「 build_image.sh 」シェル・スクリプトを変更し、 Docker リポジトリを自分のものに変更します。たとえば、「 `m uneer7589` 」を Docker リポジトリ名に置き換えます。Docker イメージ名とタグ（「 ownload -tusimple 」や「 1.0 」など）を変更することもできます。

```
#!/bin/bash
#
# A simple script to build the Docker image.
#
# $ build_image.sh
set -ex

IMAGE=muneer7589/download-tusimple
TAG=1.0

# Build image
echo "Building image: "$IMAGE
docker build . -f Dockerfile \
 --tag "${IMAGE}:${TAG}"
echo "Finished building image: "$IMAGE

# Push image
echo "Pushing image: "$IMAGE
docker push "${IMAGE}:${TAG}"
echo "Finished pushing image: "$IMAGE
```

d. スクリプトを実行して Docker イメージを構築し、次のコマンドを使用して Docker リポジトリにプッシュします。

```
chmod +x build_image.sh
./build_image.sh
```

2. 実行を送信します。 AI ジョブをダウンロードして抽出し、前処理し、 TupSimple LANE 検出データセットを「 pvc 」に格納します。このデータセットは、 NetApp Trident によって動的に作成されます。

a. 実行ファイルを送信するには、次のコマンドを使用します。 AI job ：

```
runai submit
--name download-tusimple-data
--pvc azurenetappfiles:100Gi:/mnt
--image muneer7589/download-tusimple:1.0
```

b. 次の表に情報を入力して、実行ファイルを送信します。 AI job ：

フィールド	**Value** または概要のいずれかです
名前	ジョブの名前
- PVC	[StorageClassName]: Size:ContainerMountPath という形式の PVC では、ストレージクラス azurenetappfiles で Trident を使用して、オンデマンドで PVC を作成します。この場合の永続ボリューム容量は 100Gi で、パス /mnt にマウントされます。
イメージ（ Image ）	このジョブのコンテナの作成時に使用する Docker イメージ

出力は次の例のようになります。

```
The job 'download-tusimple-data' has been submitted successfully
You can run `runai describe job download-tusimple-data -p lane-detection` to check the job status
```

c. 送信された RUN ： AI ジョブのリストを表示します。

```
runai list jobs
```

```
Showing jobs for project lane-detection
NAME                  STATUS            AGE  NODE                          IMAGE                          TYPE  PROJECT        USER          GPUs Allocated (Requested)
PODs Running (Pending)  SERVICE URL(S)
download-tusimple-data  ContainerCreating  1m   aks-agentpool-34613062-vmss00000a  muneer7589/download-tusimple:1.0  Train  lane-detection  verronmartina  0 (0)
1 (0)
```

d. 送信されたジョブログを確認してください。

```
runai logs download-tusimple-data -t 10
```

```
751150K .......... .......... .......... .......... ..........  6% 16.2M 20m37s
751200K .......... .......... .......... .......... ..........  6% 11.1M 20m37s
751250K .......... .......... .......... .......... ..........  6% 12.5M 20m36s
751300K .......... .......... .......... .......... ..........  6% 11.3M 20m36s
751350K .......... .......... .......... .......... ..........  6% 15.2M 20m36s
751400K .......... .......... .......... .......... ..........  6% 10.5M 20m36s
751450K .......... .......... .......... .......... ..........  6% 15.2M 20m36s
751500K .......... .......... .......... .......... ..........  6% 14.1M 20m36s
751550K .......... .......... .......... .......... ..........  6% 24.3M 20m36s
751600K .......... .......... .......... .......... ..........  6% 26.3M 20m36s
```

e. 作成された「 pvc 」をリストします。次のステップでトレーニングを行うには ' この pvc コマンドを

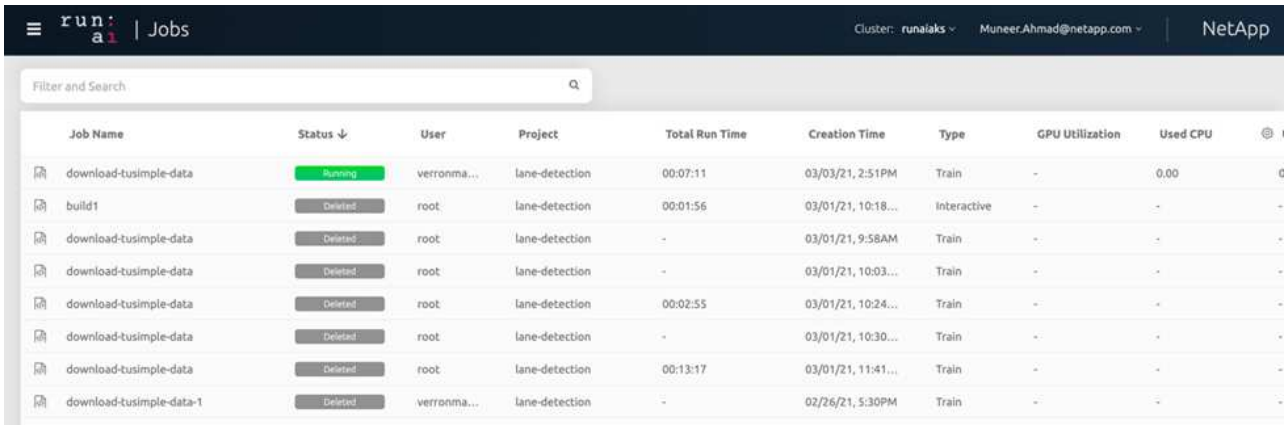
使用します

```
kubectl get pvc | grep download-tusimple-data
```

出力は次の例のようになります。

```
pvc-download-tusimple-data-0   Bound    pvc-bb03b74d-2c17-40c4-a445-79f3de8d16d5   100Gi      RWO            azurenetappfiles   4m47s
```

a. 実行中のジョブを確認します： AI UI （または app.run.ai` ）。



Horovod を使用して、分散レーン検出トレーニングを実施します

Horovod を使用した分散型レーン検出トレーニングの実行は、オプションのプロセスです。ただし、実行する手順は次のとおりです。

1. Docker イメージをビルドしてプッシュするか、既存の Docker イメージを使用する場合はこの手順を省略します（例：「 muneer7589/dist lane -detection ： 3.1 ）：」
   a. ホームディレクトリに切り替えます。

   ```
   cd ~
   ```

   b. プロジェクトディレクトリの lane -detection -SCNN-horovod. に移動します

   ```
   cd ./lane-detection-SCNN-horovod
   ```

   c. 「 build_image.sh 」シェルスクリプトを変更し、 Docker リポジトリを自分のものに変更します（たとえば、「 m uneer7589 」を Docker リポジトリ名に置き換えます）。Docker イメージ名とタグも変更できます（「 dist-dlane detection 」や「 3.1 」など）。

```
#!/bin/bash
#
# A simple script to build the distributed Docker image.
#
# $ build_image.sh
set -ex

IMAGE=muneer7589/dist-lane-detection
TAG=3.0

# Build image
echo "Building image: "$IMAGE
docker build . -f Dockerfile \
 --tag "${IMAGE}:${TAG}"
echo "Finished building image: "$IMAGE

# Push image
echo "Pushing image: "$IMAGE
docker push "${IMAGE}:${TAG}"
echo "Finished pushing image: "$IMAGE
```

d. スクリプトを実行して Docker イメージを構築し、 Docker リポジトリにプッシュします。

```
chmod +x build_image.sh
./build_image.sh
```

2. RUN ：「分散型トレーニング（ MPI ）実行のための AI ジョブ」を提出します。

a. 実行の送信を使用：前述のステップで PVC を自動的に作成するための AI （データのダウンロード用）のみ RWO アクセスを許可します。これにより、複数のポッドまたはノードが分散トレーニング用に同じ PVC にアクセスすることはできません。アクセスモードを ReadWriteMany に更新し、Kubernetes パッチを使用して更新します。

b. まず、次のコマンドを実行して PVC のボリューム名を取得します。

```
kubectl get pvc | grep download-tusimple-data
```

```
root@ai-w-gpu-2:/mnt/ai_data/anf_runai/lane-detection-SCNN-horovod# kubectl get pvc | grep download-tusimple-data
pvc-download-tusimple-data-0   Bound    pvc-bb03b74d-2c17-40c4-a445-79f3de8d16d5   100Gi      RWX            azurenetappfiles   2d4h
```

c. ボリュームにパッチを適用し、アクセスモードを ReadWriteMany に更新します（次のコマンドでは、ボリューム名を各自のに置き換えてください）。

```
kubectl patch pv pvc-bb03b74d-2c17-40c4-a445-79f3de8d16d5 -p
'{"spec":{"accessModes":["ReadWriteMany"]}}'
```

d. 次の表の情報を使用して、分散トレーニングジョブを実行するための AI MPI ジョブを実行します。

```
runai submit-mpi
--name dist-lane-detection-training
--large-shm
--processes=3
--gpu 1
--pvc pvc-download-tusimple-data-0:/mnt
--image muneer7589/dist-lane-detection:3.1
-e USE_WORKERS="true"
-e NUM_WORKERS=4
-e BATCH_SIZE=33
-e USE_VAL="false"
-e VAL_BATCH_SIZE=99
-e ENABLE_SNAPSHOT="true"
-e PVC_NAME="pvc-download-tusimple-data-0"
```

フィールド	**Value** または概要のいずれかです
名前	配布トレーニングジョブの名前
大きなシャン	大容量の /dev/shm デバイスを RAM にマウントする共有ファイルシステムであり、複数の CPU ワーカーがバッチを処理して CPU RAM にロードするために十分な共有メモリを提供します。
プロセス	配布されたトレーニングプロセスの数
GPU	このジョブでジョブに割り当てる GPU / プロセスの数には、 3 つの GPU ワーカープロセスがあります（ --processes=3 ）。各プロセスは 1 つの GPU で割り当てられます（ --GPU 1 ）。
PVC	前のジョブ（ download-tusimple-data-0 ）によって作成された既存の永続ボリューム（ pvc -pdownload -tusimple-data-0 ）を使用し、パス /mnt にマウントします
イメージ（ Image ）	このジョブのコンテナの作成時に使用する Docker イメージ
コンテナで設定する環境変数を定義します	
ワーカーを使用します	引数を true に設定すると、マルチプロセスのデータロードがオンになります
num_Workers	データローダーワーカープロセスの数
batch_size	トレーニングバッチサイズ
使用 _ VAL	引数を true に設定すると、検証が可能になります
Val_batch_size	検証バッチサイズ
Snapshot の有効化	引数を true に設定すると、 ML バージョン管理のためにデータとトレーニング済みのモデルスナップショットを取得できます

フィールド	**Value** または概要のいずれかです
pvc_name	スナップショットを作成する PVC の名前。上記のジョブ送信では、データセットとトレーニング済みモデルで構成される Pvc-de-download-tusimple-data-0 のスナップショットを作成します

出力は次の例のようになります。

```
The job 'dist-lane-detection-training' has been submitted successfully
You can run `runai describe job dist-lane-detection-training -p lane-detection` to check the job status
```

e. 送信されたジョブを一覧表示します。

```
runai list jobs
```

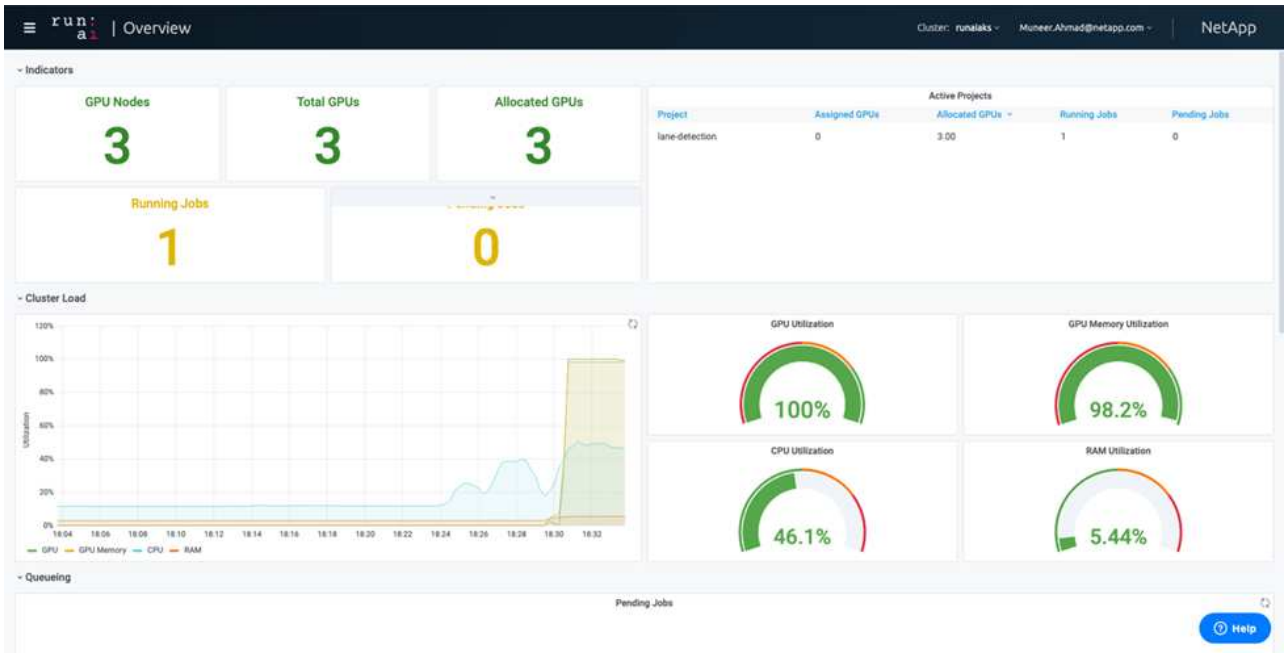
```
NAME                          STATUS     AGE  NODE        IMAGE                            TYPE   PROJECT         USER           GPUs Allocated (Requested)  PODs
 SERVICE URL(S)
download-tusimple-data        Succeeded  1d                muneer7589/download-tusimple:1.0   Train  lane-detection  verronmartina  - (0)                       0 (0)

dist-lane-detection-training  Init:0/1   2m   <multiple>  muneer7589/dist-lane-detection:3.1 Train  lane-detection  root           3 (3)                       4 (0)
```

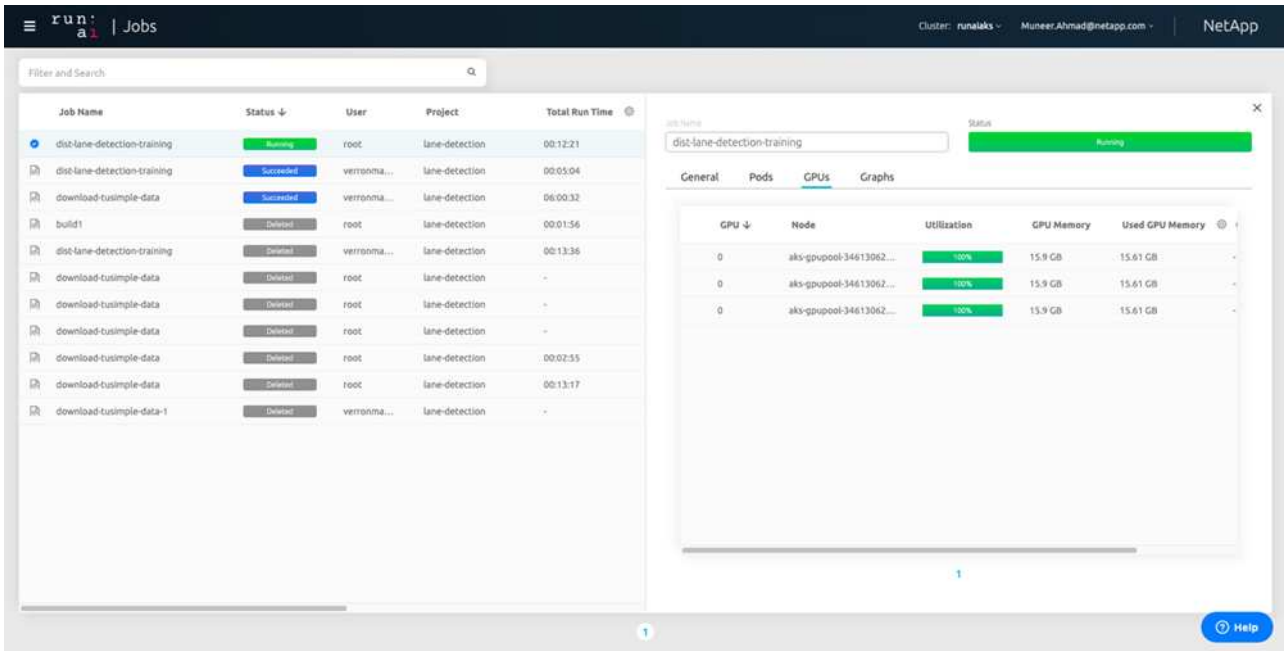
f. 送信されたジョブログ：

```
runai logs dist-lane-detection-training
```

```
root@ai-w-gpu-2:~/runai# runai logs dist-lane-detection-training
Running with 3 workers
2021-03-04 17:29:23.158449: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] Successfully opened dynamic library libcudart.so.10.1
+ POD_NAME=dist-lane-detection-training-worker-0
+ [ d = - ]
+ shift
+ /opt/kube/kubectl cp /opt/kube/hosts dist-lane-detection-training-worker-0:/etc/hosts_of_nodes
+ POD_NAME=dist-lane-detection-training-worker-2
+ [ d = - ]
+ shift
+ /opt/kube/kubectl cp /opt/kube/hosts dist-lane-detection-training-worker-2:/etc/hosts_of_nodes
+ POD_NAME=dist-lane-detection-training-worker-1
```

g. 実行中のトレーニングジョブを確認します。次の図に示すように、 AI GUI （または app.runai.ai): run ： AI Dashboard ）。最初の図は、分散トレーニングジョブ用に割り当てられた 3 つの GPU を AKS の 3 つのノードに分散し、 2 番目の実行である AI ジョブの詳細を示しています。





h. トレーニングが完了したら、作成され、実行済みの NetApp Snapshot コピーである AI ジョブを確認します。

```
runai logs dist-lane-detection-training --tail 1
```

```
[1,0]<stdout>:Snapshot snap-pvc-download-tusimple-data-0-dist-lane-detection-training-launcher-2021-03-05-16-23-42 created in namespace runai-lane-detection
```

```
kubectl get volumesnapshots | grep download-tusimple-data-0
```

NetApp Snapshot コピーからデータをリストアします

NetApp Snapshot コピーからデータをリストアするには、次の手順を実行します。

1. ホームディレクトリに切り替えます。

```
cd ~
```

2. プロジェクトディレクトリの lane -detection -SCNN-horovod' に移動します

```
cd ./lane-detection-SCNN-horovod
```

3. 「 restore-snaphot-pvc.yaml 」を変更し、「 ataSource `name」 フィールドをデータのリストア元の Snapshot コピーに更新します。また、データを復元する PVC 名を変更することもできます。この例では、「 restored-tusimple」 です。

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: restored-tusimple
spec:
  storageClassName: azurenetappfiles
  dataSource:
    name: snap-pvc-download-tusimple-data-0-dist-lane-detection-training-launcher-2021-03-05-16-23-42
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 100Gi
```

4. 「 restore -snapshot-pvc.yaml 」を使用して新しい PVC を作成します。

```
kubectl create -f restore-snapshot-pvc.yaml
```

出力は次の例のようになります。

```
persistentvolumeclaim/restored-tusimple created
```

5. 復元されたばかりのデータをトレーニングに使用する場合、ジョブ送信は以前と同じです。次のコマンドに示すように、トレーニングジョブの送信時に「 pvc_name 」を復元された「 pvc_name 」に置き換えるだけです。

```
runai submit-mpi
--name dist-lane-detection-training
--large-shm
--processes=3
--gpu 1
--pvc restored-tusimple:/mnt
--image muneer7589/dist-lane-detection:3.1
-e USE_WORKERS="true"
-e NUM_WORKERS=4
-e BATCH_SIZE=33
-e USE_VAL="false"
-e VAL_BATCH_SIZE=99
-e ENABLE_SNAPSHOT="true"
-e PVC_NAME="restored-tusimple"
```

パフォーマンス評価

解決策のリニアな拡張性を示すために、 GPU × 1 と GPU × 3 という 2 つのシナリオでパフォーマンステストを実施しました。GPU 割り当て、 GPU とメモリの使用率、シングルノードと 3 ノードの異なるメトリックは、 TuSimple LANE 検出データセットのトレーニング中に取得されました。データは、トレーニングプロセス中のリソース使用率を分析するために 5 倍に増加します。

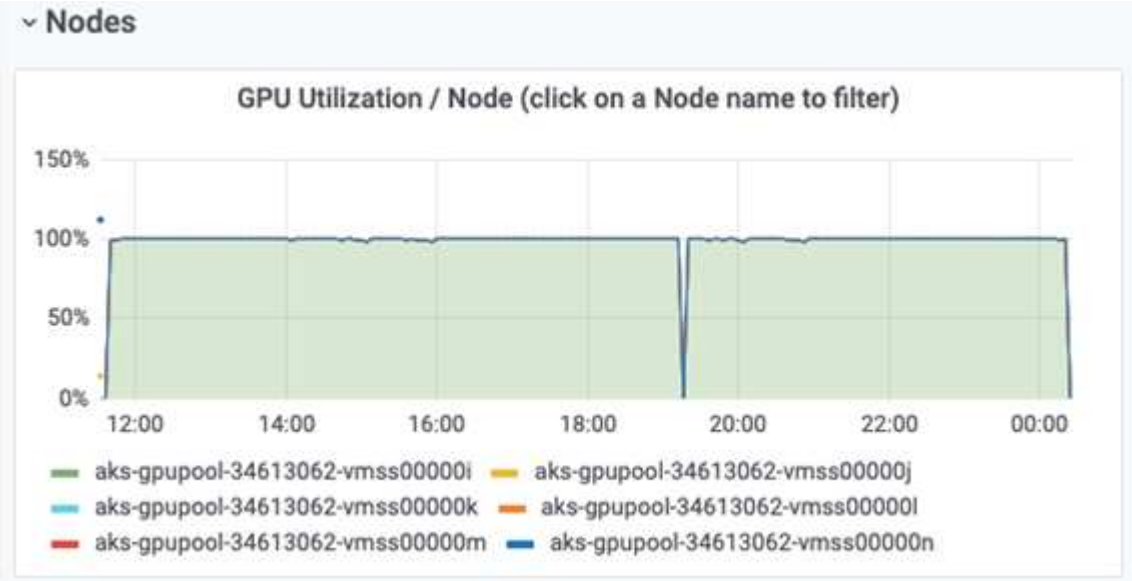
解決策を使用すると、まず小規模なデータセットを配置し、一部の GPU で作業を開始できます。GPU の需要とデータ量が増加した場合、標準階層ではテラバイト規模まで動的にスケールアウトし、 Premium 階層にすばやくスケールアップして、データを移動することなく、テラバイトあたりのスループットを 4 倍にすることができます。このプロセスの詳細については、を参照してください。 "Azure NetApp Files サービスレベル"。

1 つの GPU での処理時間は 12 時間 45 分でした。3 つのノードにまたがる 3 つの GPU での処理時間は約 4 時間 30 分でした。

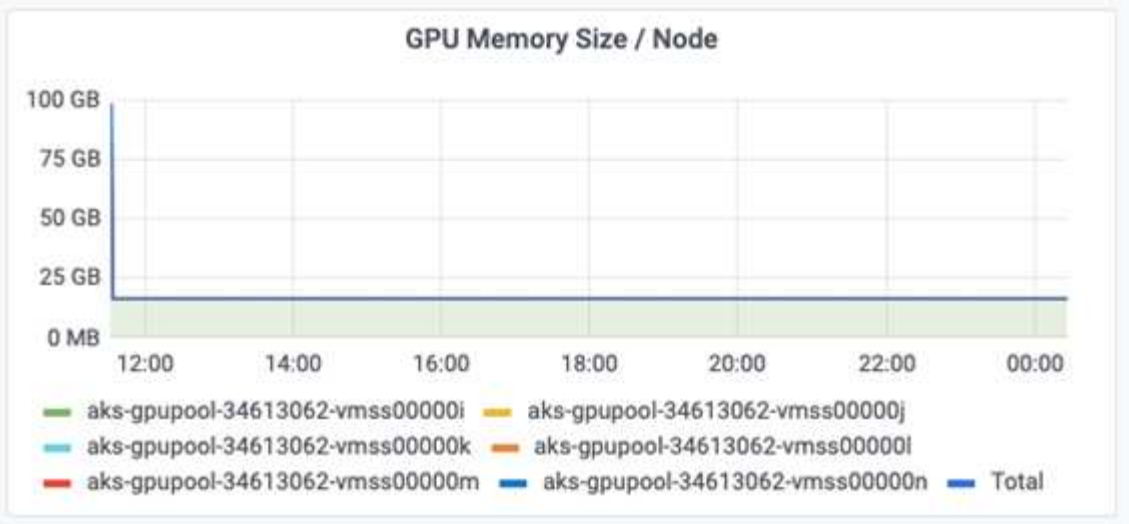
本ドキュメントの以降の各セクションにある図は、個々のビジネスニーズに基づくパフォーマンスと拡張性の例を示しています。

次の図は、 1 つの GPU 割り当てとメモリ使用率を示しています。

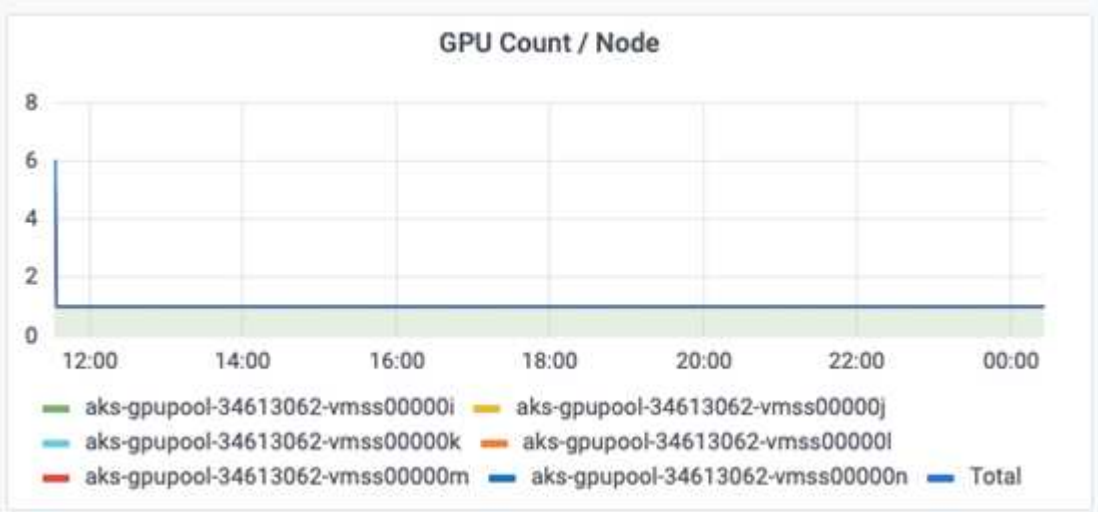
次の図は、シングルノードの GPU 利用率を示しています。



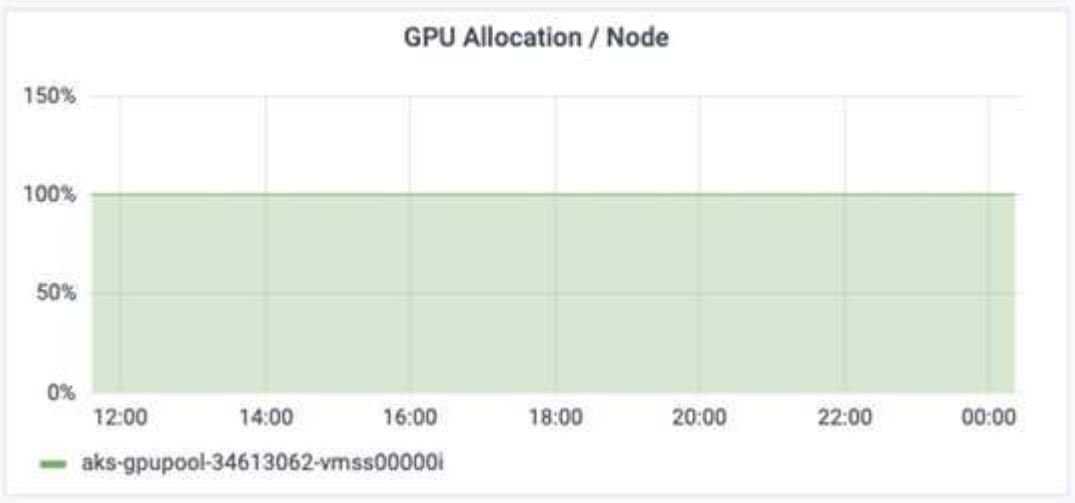
次の図は、シングルノードのメモリサイズ（ 16GB ）を示しています。



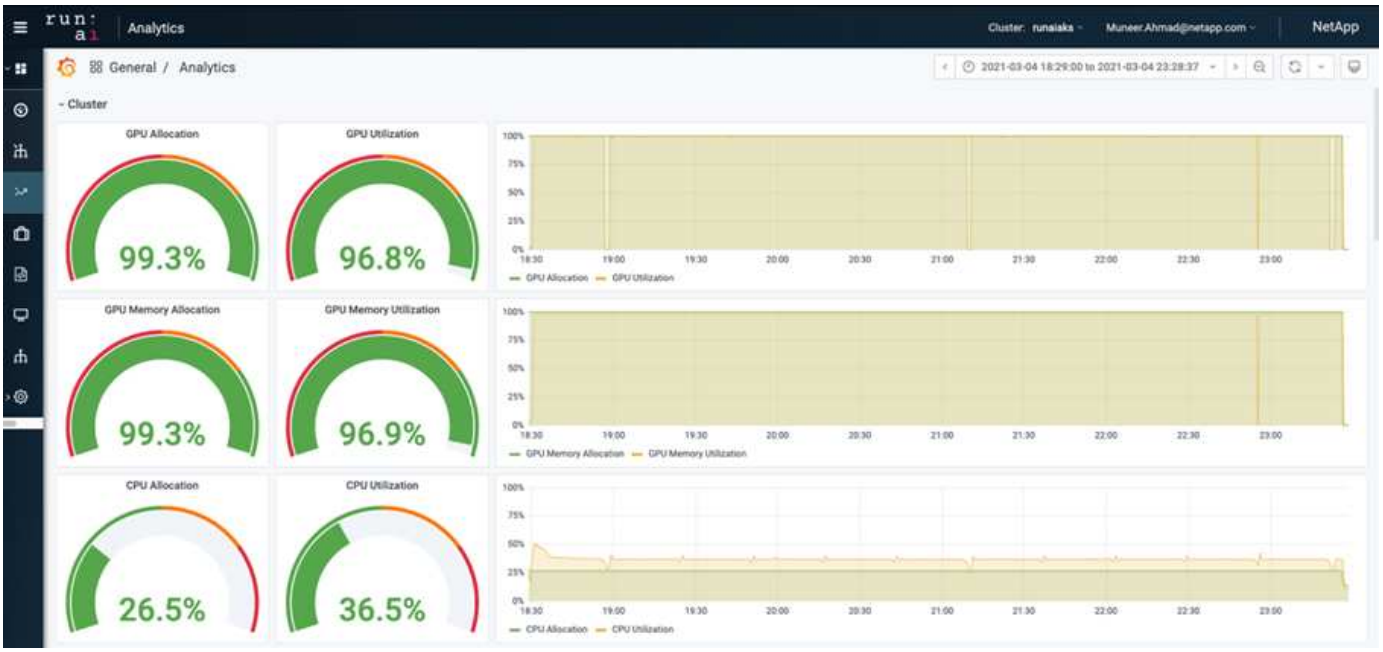
次の図は、シングルノードの GPU 数（ 1 ）を示しています。



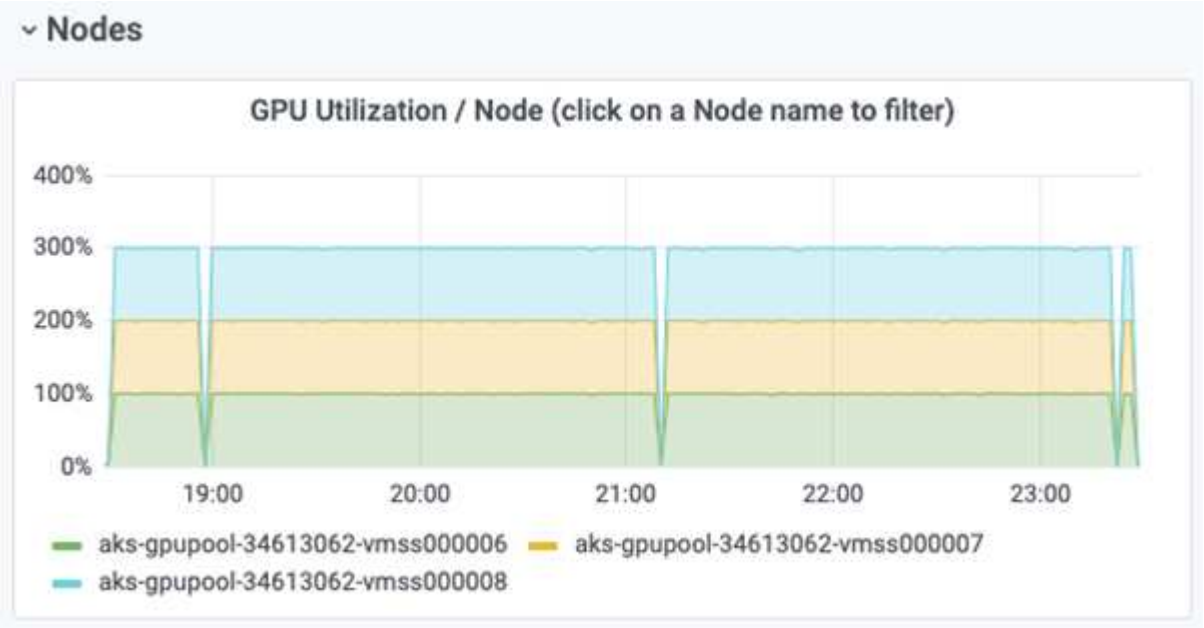
次の図は、シングルノードの GPU 割り当て（ % ）を示しています。



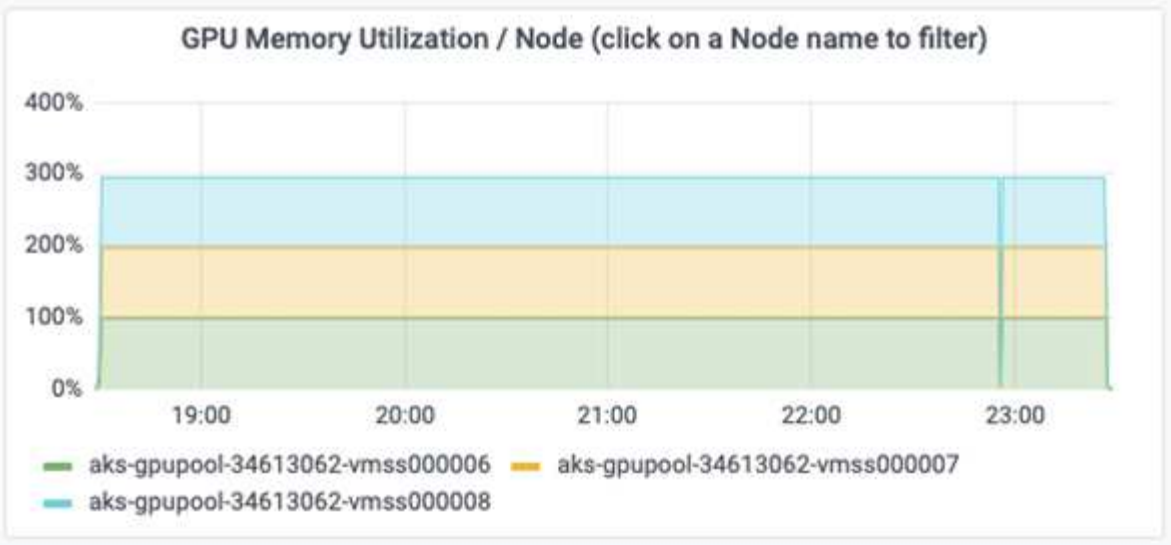
次の図は、 GPU の割り当てとメモリという 3 つのノードにまたがる 3 つの GPU を示しています。



次の図は、 3 つのノードの使用率（ % ）にまたがる 3 つの GPU を示しています。



次の図は、 3 つのノードにまたがる 3 つの GPU のメモリ利用率（ % ）を示しています。



Azure NetApp Files サービスレベル

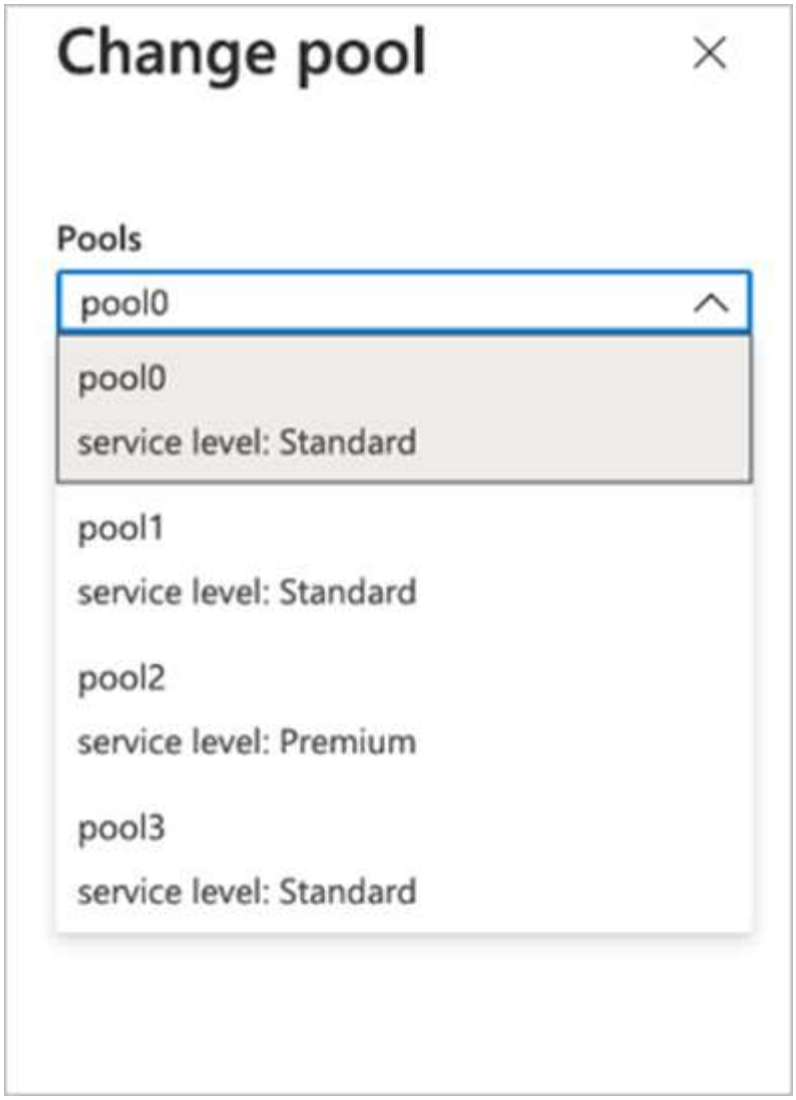
既存のボリュームのサービスレベルを変更するには、を使用する別の容量プールにボリュームを移動します "サービスレベル" 必要なのはボリュームです。ボリュームの既存のサービスレベル変更では、データを移行する必要はありません。また、ボリュームへのアクセスにも影響しません。

ボリュームのサービスレベルを動的に変更する

ボリュームのサービスレベルを変更するには、次の手順を実行します。

1. Volumes （ボリューム）ページで、サービスレベルを変更するボリュームを右クリックします。［ プールの変更 ］ を選択します

2. プールの変更ウィンドウで、ボリュームの移動先とする容量プールを選択します。[OK] をクリックします。



サービスレベルの変更を自動化

動的サービスレベルの変更は現在、パブリックプレビューで有効になっていますが、デフォルトでは有効になっていません。Azure サブスクリプションでこの機能を有効にするには、次の手順を実行します "ボリュームのサービスレベルを動的に変更する"」

- Azure では、 CLI コマンドでも次のコマンドを使用できます。Azure NetApp Files のプール・サイズの変更の詳細については、を参照してください "AZ netappfiles ボリューム： Azure NetApp Files （ ANF ）ボリュームリソースの管理"。

```
az netappfiles volume pool-change -g mygroup
--account-name myaccname
-pool-name mypoolname
--name myvolname
--new-pool-resource-id mynewresourceid
```

- ここに示す 'set-aznetappfilesvolumepool' コマンドレットを使用すると、 Azure NetApp Files ボリュームのプールを変更できます。ボリュームプールのサイズ変更の詳細については、を参照してください "Azure

NetApp Files ボリュームのプールを変更します"。

```
Set-AzNetAppFilesVolumePool
-ResourceGroupName "MyRG"
-AccountName "MyAnfAccount"
-PoolName "MyAnfPool"
-Name "MyAnfVolume"
-NewPoolResourceId 7d6e4069-6c78-6c61-7bf6-c60968e45fbf
```

まとめ

ネットアップと Run ： AI は、このテクニカルレポートの作成時にパートナー関係を結び、 Azure NetApp Files 独自の機能と、 AI ワークロードのオーケストレーションを簡易化する AI プラットフォームを、 RUN の手法で実証しています。このテクニカルレポートでは、分散レーン検出トレーニングのためにデータパイプラインとワークロードオーケストレーションのプロセスを合理化するリファレンスアーキテクチャを提供します。

その結果、大規模な分散トレーニング（特にパブリッククラウド環境）に関しては、リソースのオーケストレーションとストレージのコンポーネントは解決策の重要な要素となります。データ管理によって複数の GPU 処理が妨げられることがないようにすることで、 GPU サイクルの利用率を最適化できます。そのため、大規模な分散トレーニングのために、システムをできるだけ費用対効果の高いものにすることができます。

ネットアップが提供するデータファブリックを使用すると、データサイエンティストやデータエンジニアは、手動操作なしでオンプレミスとクラウドを連携させ、同期データを保持できるため、この課題を克服できます。つまり、データファブリックによって、 AI ワークフローを複数の場所に分散して管理するプロセスがスムーズになります。また、コンピューティングと分析、トレーニング、検証に必要なときに必要な場所でデータを利用できるため、オンデマンドでのデータ可用性が容易になります。この機能により、データ統合だけでなく、データパイプライン全体の保護とセキュリティも実現できます。

追加情報

このドキュメントに記載されている情報の詳細については、以下のドキュメントや Web サイトを参照してください。

- データセット： TuSimple

  "https://github.com/TuSimple/tusimple-benchmark/tree/master/doc/lane_detection"

- ディープラーニングネットワークアーキテクチャ：空間的な畳み込みニューラルネットワーク

  "https://arxiv.org/abs/1712.06080"

- 分散型ディープラーニングトレーニングフレームワーク： Horovod

  "https://horovod.ai/"

- 実行： AI コンテナオーケストレーション解決策： run ： AI 製品の概要

"https://docs.run.ai/home/components/"

- 実行： AI インストールドキュメント

  "https://docs.run.ai/Administrator/Cluster-Setup/cluster-install/#step-3-install-runai"
  "https://docs.run.ai/Administrator/Researcher-Setup/cli-install/#runai-cli-installation"

- 実行時のジョブの送信： AI CLI

  "https://docs.run.ai/Researcher/cli-reference/runai-submit/"

  "https://docs.run.ai/Researcher/cli-reference/runai-submit-mpi/"

- Azure クラウドリソース： Azure NetApp Files

  "https://docs.microsoft.com/azure/azure-netapp-files/"

- Azure Kubernetes Service の略

  "https://azure.microsoft.com/services/kubernetes-service/-features"

- Azure VM SKUs

  "https://azure.microsoft.com/services/virtual-machines/"

- Azure VM と GPU SKU

  "https://docs.microsoft.com/azure/virtual-machines/sizes-gpu"

- NetApp Trident

  "https://github.com/NetApp/trident/releases"

- ネットアップのデータファブリック

  "https://www.netapp.com/data-fabric/what-is-data-fabric/"

- ネットアップの製品マニュアル

  "https://www.netapp.com/support-and-training/documentation/"

TR-4841 ：『 Hybrid Cloud AI Operating System with Data Caching 』

ネットアップ Yochay Ettun 、 cnvrg.io 、 David Arnette 、 Rick Huang 氏

データの急増と ML と AI の急激な成長により、独自の開発と実装の課題を抱えるゼタバイト経済が生まれました。

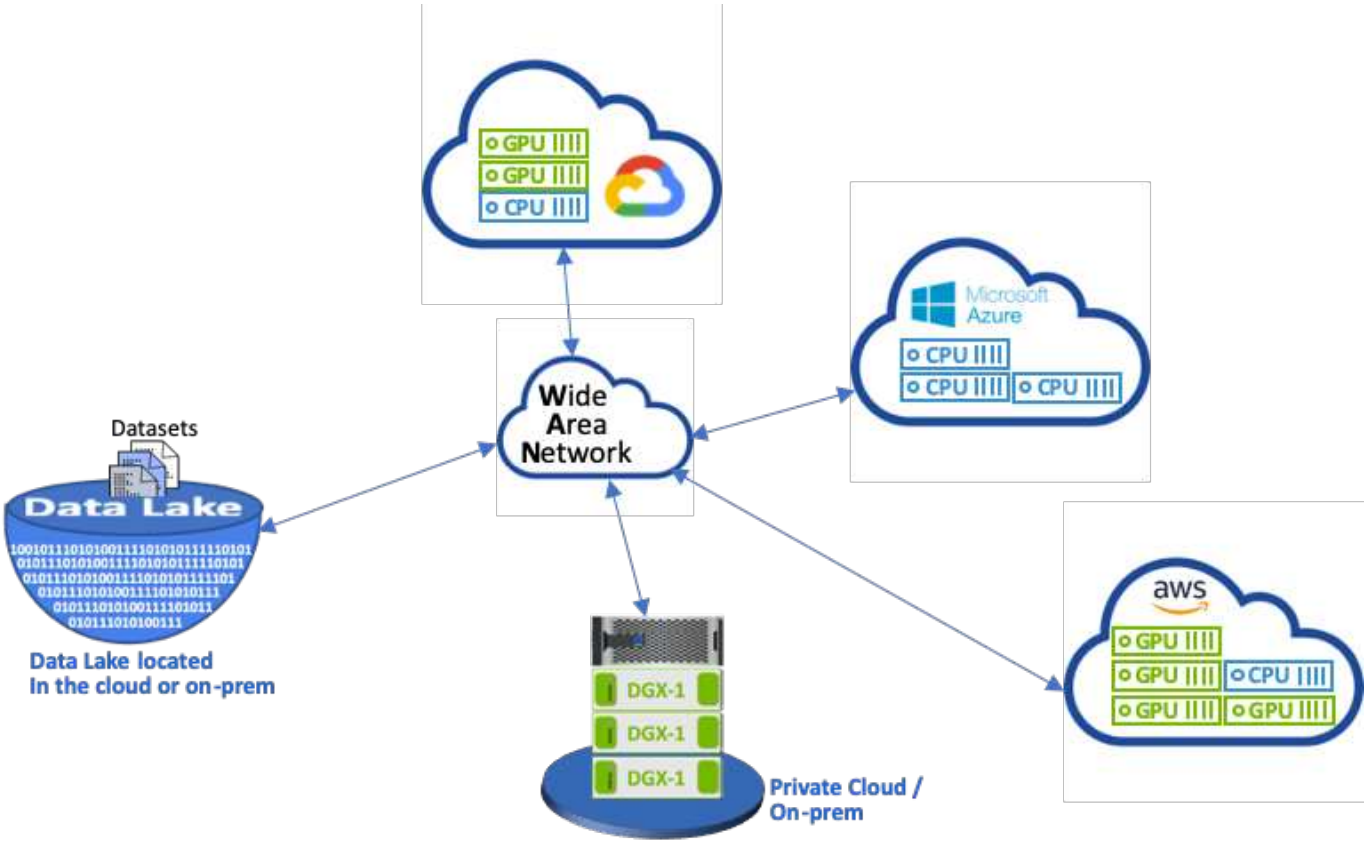
ML モデルは大量のデータを必要とし、コンピューティングリソースにはハイパフォーマンスのデータストレージが必要であることは広く知られていますが、実際には、このモデルを実装するのはそれほど簡単ではありません。特にハイブリッドクラウドインスタンスや柔軟なコンピューティングインスタンスを使用する場合はそうです。一般に、大量のデータが低コストのデータレイクに保存されます。このデータレイクでは、 GPU などのハイパフォーマンスな AI コンピューティングリソースは効率的にアクセスできません。この問題は、

一部のワークロードがクラウドで動作し、一部のワークロードがオンプレミス環境または別の HPC 環境に完全に配置されているハイブリッドクラウドインフラにさらに悪化しています。

このドキュメントでは、 IT プロフェッショナルやデータエンジニアがトポロジに対応したデータハブで真のハイブリッドクラウド AI プラットフォームを構築できる、新しい解決策を紹介します。これにより、データサイエンティストは、コンピューティングリソースに近接してデータセットのキャッシュを瞬時に自動作成できます。 どこにいても、その結果、高性能なモデルトレーニングを実施できるだけでなく、データセットバージョンハブ内のデータセットキャッシュ、バージョン、リネージにすぐにアクセスできる複数の AI 専門家のコラボレーションなど、さらなるメリットが得られます。

ユースケースの概要と問題点

データセットとデータセットのバージョンは通常、 NetApp StorageGRID オブジェクトベースストレージなどのデータレイクに配置されるため、コストの削減やその他の運用上のメリットが得られます。データサイエンティストは、これらのデータセットを取得して複数の手順でエンジニアを配置し、特定のモデルを使用したトレーニングに備えます。多くの場合、途中で複数のバージョンが作成されます。次のステップとして、データサイエンティストは、モデルを実行するために最適化されたコンピューティングリソース（ GPU 、ハイエンド CPU インスタンス、オンプレミスクラスタなど）を選択する必要があります。次の図は、 ML コンピューティング環境にデータセットの距離がないことを示しています。



ただし、複数のトレーニング実験を異なるコンピューティング環境で並行して実行する必要があります。それぞれの環境では、データレイクからデータセットをダウンロードする必要があります。これはコストと時間のかかるプロセスです。データセットがコンピューティング環境（特にハイブリッドクラウド）に近接していることは保証されません。また、同じデータセットで独自の実験を行う他のチームメンバーも、同じ複雑なプロ

セスを実行する必要があります。データアクセスが遅いことが明らかなだけでなく、データセットのバージョン、データセットの共有、コラボレーション、再現性の追跡にも困難が伴います。

お客様の要件

リソースを効率的に使用しながら、高パフォーマンスの ML を実行するためには、お客様の要件が異なる場合があります。たとえば、次のような場合があります。

- を実行する各コンピューティングインスタンスからデータセットに高速アクセス 高額なダウンロードやデータアクセスの複雑さを伴わないトレーニングモデル
- は任意のコンピューティングインスタンス（ GPU または CPU ）を使用する クラウドでもオンプレミスでも、場所を気にする必要はありません 」と入力します
- で複数のトレーニング実験を実行することで、効率と生産性が向上します を使用せずに、同一データセット上の異なるコンピューティングリソースと並行して実行できます 不要な遅延とデータ遅延
- コンピューティングインスタンスのコストを最小限に抑えます
- データセット、そのリネージ、バージョン、およびその他のメタデータの詳細の記録を保持するツールにより、再現性が向上しました
- 共有とコラボレーションを強化して、の権限を持つすべてのメンバーをサポートします チームはデータセットにアクセスして実験を実行できます

NetApp ONTAP データ管理ソフトウェアにデータセットのキャッシングを実装するには、次のタスクを実行する必要があります。

- コンピューティングリソースに最も近い NFS ストレージを構成して設定します。
- キャッシュするデータセットとバージョンを決定します。
- キャッシュされたデータセットにコミットされた合計メモリと、追加のキャッシュコミットに使用できる NFS ストレージの量（キャッシュ管理など）を監視します。
- 特定の時間内に使用されなかったデータセットは、キャッシュ内でエージングアウトします。デフォルトは 1 日で、その他の設定オプションも使用できます。

解決策の概要

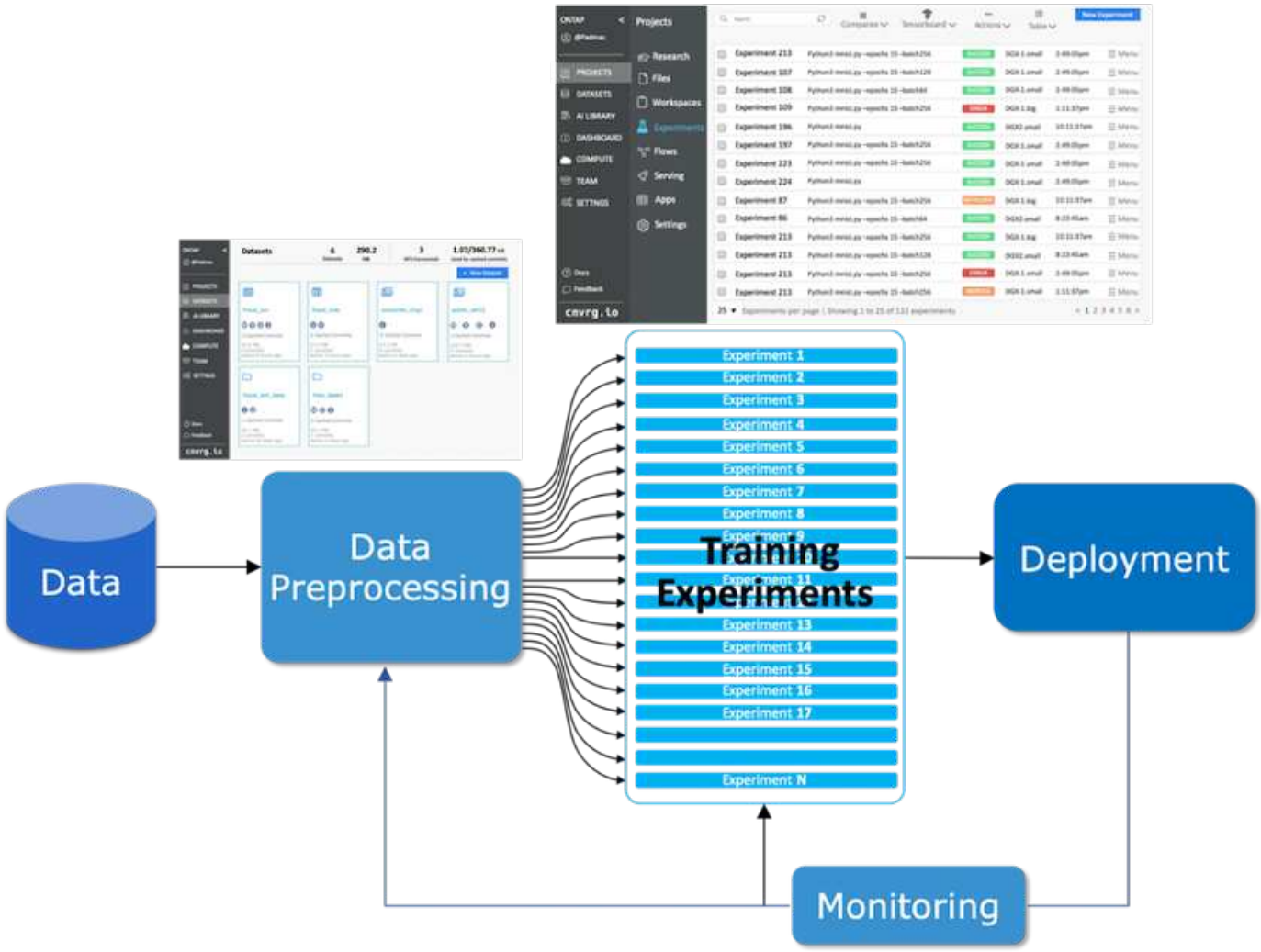
このセクションでは、従来のデータサイエンスパイプラインとその欠点について説明します。また、提案するデータセットキャッシング解決策のアーキテクチャについても説明します。

従来のデータサイエンスパイプラインと欠点

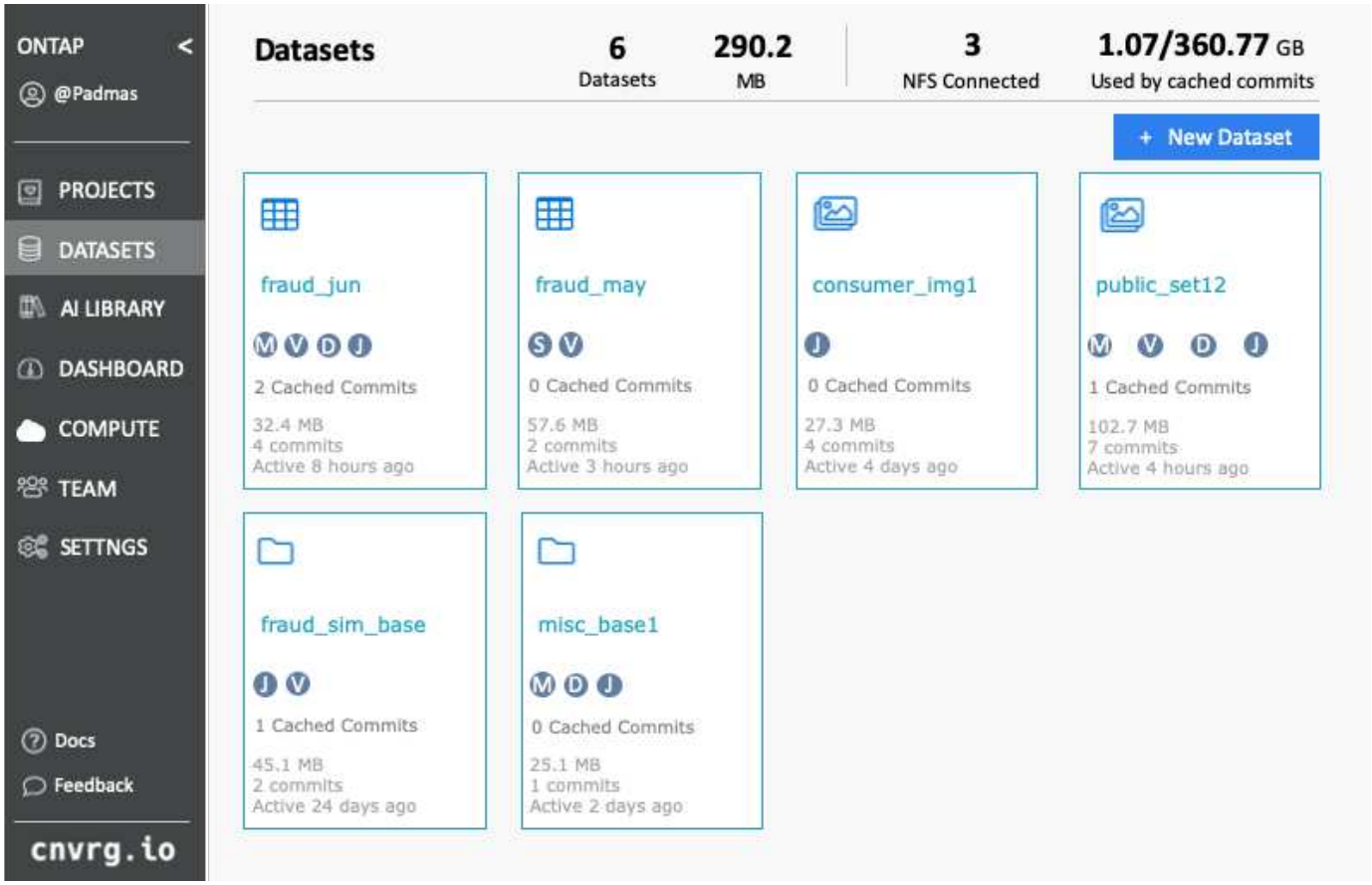
ML モデルの開発と導入の一般的な手順には、次のような反復的な手順が含まれます。

- データの取り込み
- データの前処理（データセットの複数のバージョンを作成）
- HyperParameter の最適化、さまざまなモデルなどを含む複数の実験を実行する
- 導入
- Monitoringcnvrg.io は、研究から導入までのすべてのタスクを自動化する包括的なプラットフォームを開発しました。次の図に、パイプラインに関するダッシュボードのスクリーンショットのごく一部を示しま

す。



パブリックリポジトリやプライベートデータから複数のデータセットを使用するのは非常に一般的です。また、データセットのクリーンアップやフィーチャーエンジニアリングによって、各データセットに複数のバージョンが生成されることもよくあります。次の図に示すように、データセットハブとバージョンハブを提供するダッシュボードは、コラボレーションツールと整合性ツールをチームで確実に使用できるようにするために必要です。



パイプラインの次のステップではトレーニングを行います。トレーニングモデルには複数の並行インスタンスが必要で、それぞれがデータセットと特定のコンピューティングインスタンスに関連付けられている必要があります。データセットを特定のコンピューティングインスタンスと特定の実験にバインドすることは、一部の実験は Amazon Web Services （ AWS ）の GPU インスタンスによって実行され、それ以外の実験は DGX-1 インスタンスまたは DGX-2 インスタンスによってオンプレミスで実行される可能性があるため、難しい課題です。GCP の CPU サーバーでは他の実験が実行され、データセットの場所がトレーニングを実行するコンピューティングリソースにあまり近接していない場合があります。合理的なプロキシミティには、データセットストレージからコンピューティングインスタンスへの完全な 10GbE 以上の低レイテンシ接続が必要です。

データサイエンティストが、トレーニングを実行し、実験を実行するコンピューティングインスタンスにデータセットをダウンロードするのは、一般的に行われます。ただし、この方法にはいくつかの潜在的な問題があります。

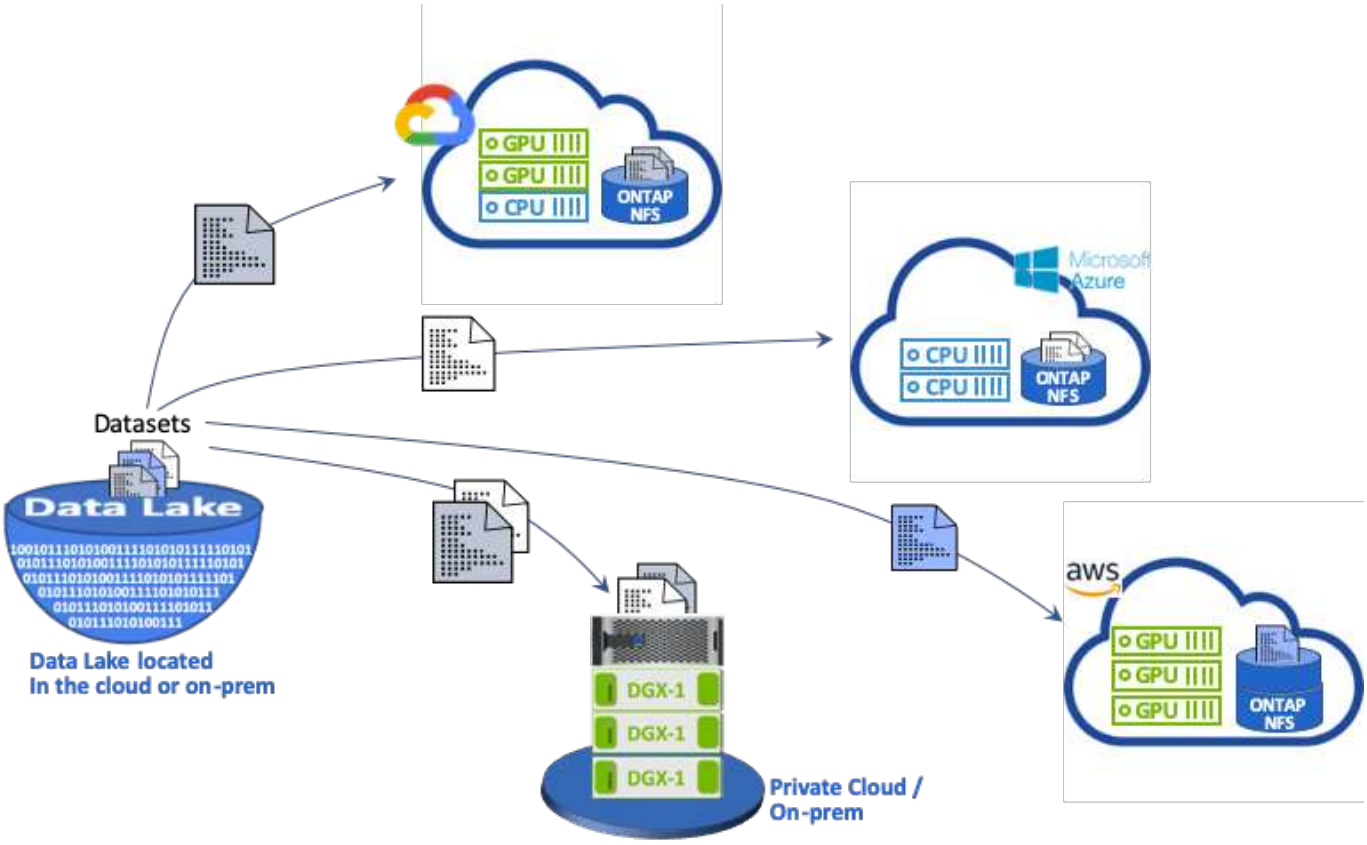
- データサイエンティストがデータセットをコンピューティングインスタンスにダウンロードした場合、統合コンピューティングストレージのパフォーマンスが高くても保証はありません（ハイパフォーマンスシステムの例としては ONTAP AFF A800 NVMe 解決策が挙げられます）。
- ダウンロードしたデータセットが 1 つのコンピューティングノードに存在する場合、複数のノードで分散モデルを実行すると（ NetApp ONTAP のハイパフォーマンス分散ストレージとは異なり）ストレージがボトルネックになることがあります。
- トレーニング実験の次の反復は、キューの競合や優先順位のために別のコンピューティングインスタンスで実行される場合もあります。これも、データセットから計算場所へのネットワーク距離が大幅に大きくなります。
- 同じコンピューティングクラスタ上でトレーニング実験を実行する他のチームメンバーは、このデータセットを共有できません。各チームメンバーは、任意の場所からデータセットの（高価な）ダウンロードを実行します。

- 後続のトレーニングジョブで同じデータセットの他のデータセットまたはバージョンが必要な場合、データサイエンティストは、 training.NetApp および cnvrg.io を実行しているコンピューティングインスタンスにデータセットの（高価な）ダウンロードを再度実行する必要があります。これにより、これらの障害を解消する新しいデータセットキャッシング解決策が作成されます。解決策は、ホットデータセットを ONTAP ハイパフォーマンスストレージシステムにキャッシュすることで、 ML パイプラインの実行を高速化します。ONTAP NFS では、ネットアップが提供するデータファブリック（ AFF A800 など）にデータセットが 1 回だけ（一度だけ）キャッシュされ、コンピューティングと一緒に配置されます。NetApp ONTAP NFS 高速ストレージが複数の ML コンピューティングノードに対応できるようになるため、トレーニングモデルのパフォーマンスが最適化され、コスト削減、生産性、運用効率が向上します。

解決策アーキテクチャ

この解決策は、次の図に示すように、ネットアップおよび cnvrg.io から提供されます。データセットのキャッシングにより、データサイエンティストは必要なデータセットまたはデータセットのバージョンを選択し、ML コンピューティングクラスタのすぐ近くにある ONTAP NFS キャッシュに移動できます。データサイエンティストは、遅延やダウンロードを発生させることなく、複数の実験を実行できるようになりました。さらに、コラボレーションするすべてのエンジニアは、データレイクから追加のダウンロードを行うことなく、接続されたコンピューティングクラスタ（任意のノードを自由に選択できる）で同じデータセットを使用できます。データサイエンティストは、すべてのデータセットとバージョンを追跡および監視するダッシュボードを提供し、キャッシュされたデータセットを確認します。

cnvrg.io プラットフォームは、一定の期間使用されていない古いデータセットを自動検出し、キャッシュから削除します。これにより、使用頻度の高いデータセット用に NFS キャッシュの空きスペースが維持されます。ONTAP を使用したデータセットのキャッシングは、クラウドとオンプレミスで機能するため、最大限の柔軟性が得られることに注意してください。



このセクションでは、 ML ワークフローのデータキャッシングに関連する概念とコンポーネントについて説明します。

機械学習

ML は、世界中の多くの企業や組織にとって急速に不可欠になっています。そのため、 IT チームと DevOps チームは、 ML ワークロードの標準化や、 ML のジョブやパイプラインで求められる動的で負荷の高いワークフローをサポートするクラウド、オンプレミス、ハイブリッドコンピューティングリソースのプロビジョニングという課題に直面しています。

コンテナベースの機械学習と **Kubernetes**

コンテナは、共有ホストオペレーティングシステムカーネル上で実行される独立したユーザスペースインスタンスです。コンテナの採用が急速に増加しています。コンテナは、仮想マシン（ VM ）が提供するものと同じアプリケーションのサンドボックス化のメリットの多くを提供します。ただし、 VM が依存するハイパーバイザーレイヤとゲストオペレーティングシステムレイヤが排除されているため、コンテナの軽量化が大幅に向上しています。

コンテナを使用すると、アプリケーションの依存関係や実行時間などをアプリケーションで直接効率的にパッケージングできます。最も一般的に使用されるコンテナパッケージ形式は Docker コンテナです。Docker コンテナ形式でコンテナ化されたアプリケーションは、 Docker コンテナを実行できる任意のマシンで実行できます。これは、アプリケーションの依存関係がマシンに存在しない場合でも当てはまります。これは、すべての依存関係がコンテナ自体にパッケージ化されているためです。詳細については、を参照してください "Docker Web サイト"。

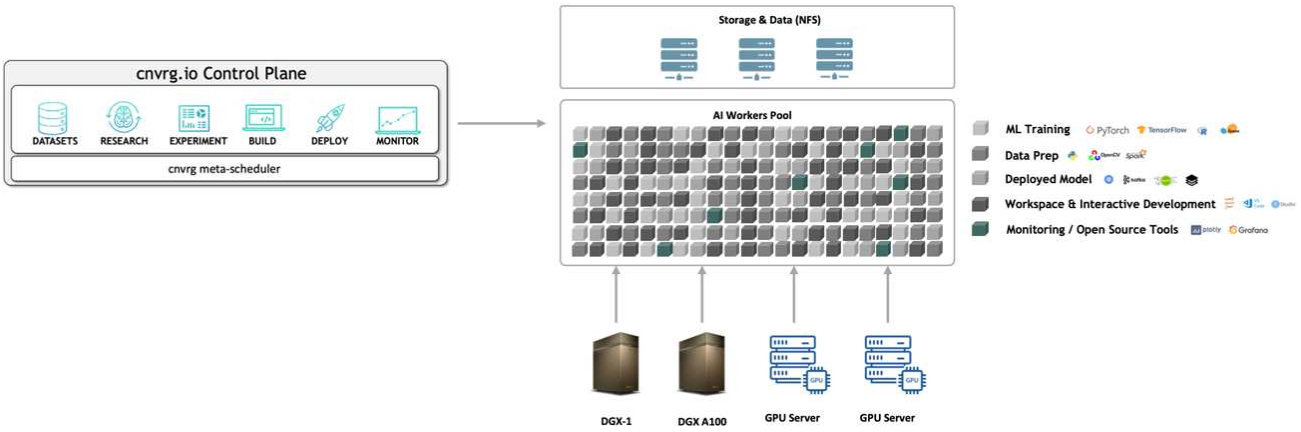
人気のあるコンテナオーケストレーションツールである Kubernetes を使用すると、データサイエンティストは柔軟なコンテナベースのジョブとパイプラインを開始できます。また、インフラチームは、管理された単一のクラウドネイティブ環境で ML ワークロードを管理および監視できます。詳細については、を参照してください "Kubernetes Web サイト"。

cnvrg.io

cnvrg.io は、企業が AI やデータサイエンスの開発を研究から生産に至るまで管理、拡張、高速化する方法を変革する AI オペレーティングシステムです。コードファーストのプラットフォームは、データサイエンティストがデータサイエンティストのために構築し、オンプレミスとクラウドのどちらでも実行できる柔軟性を提供します。モデル管理、 MLOps 、継続的な ML ソリューションを備えた cnvrg.io は、データサイエンスチームに最先端のテクノロジを提供します。そのため、 DevOps に費やす時間を短縮し、真の魔法のアルゴリズムに集中できます。cnvrg.io を使用して以来、さまざまな業界のチームが生産モデルを増やし、ビジネス価値を高めてきました。

cnvrg.io メタスケジューラ

cnvrg.IO には独自のアーキテクチャがあり、 IT エンジニアは異なるコンピューティングリソースを同じコントロールプレーンに接続し、すべてのリソースにわたって cnvrg.io で ML ジョブを管理できます。つまり、次の図に示すように、複数のオンプレミス Kubernetes クラスタ、 VM サーバ、クラウドアカウントを接続し、すべてのリソースで ML ワークロードを実行できます。



cnvrg.io データキャッシング

cnvrg.io を使用すると、データサイエンティストは、データキャッシングテクノロジを使用して、ホットデータセットとコールドデータセットのバージョンを定義できます。デフォルトでは、データセットは一元化されたオブジェクトストレージデータベースに格納されます。データサイエンティストは、選択したコンピューティングリソースに特定のデータバージョンをキャッシュして、ダウンロード時間を節約し、 ML の開発と生産性を向上させることができます。数日間キャッシュされていないデータセットは、選択した NFS から自動的に消去されます。キャッシュのキャッシュとクリアはワンクリックで実行でき、コーディング、 IT 、 DevOps の作業は必要ありません。

cnvrg.io フローと ML パイプライン

cnvrg.io フローは、本番 ML パイプラインを構築するためのツールです。フロー内の各コンポーネントは、ベースとなる Docker イメージを使用して選択したコンピューティング上で実行されるスクリプト / コードです。この設計により、データサイエンティストとエンジニアは、オンプレミスとクラウドの両方で実行できる単一のパイプラインを構築できます。cnvrg.io は、データ、パラメータ、およびアーティファクトが異なるコンポーネント間で移動していることを確認します。さらに、各フローを監視して追跡することで、再現性の高い 100% のデータサイエンスを実現します。

cnvrg.io コア

cnvrg.io コアは、データサイエンスコミュニティが DevOps よりもデータサイエンスに集中できるようにするための無償プラットフォームです。コアの柔軟なインフラストラクチャにより、データサイエンティストは、オンプレミスでもクラウドでも、あらゆる言語、 AI フレームワーク、コンピューティング環境を使用することができます。これにより、最適な処理を実行し、アルゴリズムを構築できます。cnvrg.io コアは、任意の Kubernetes クラスタ上で 1 つのコマンドを使用して簡単にインストールできます。

NetApp ONTAP AI

ONTAP AI は、 ML ワークロードとディープラーニング（ DL ）ワークロード向けのデータセンターリファレンスアーキテクチャであり、 Tesla V100 GPU を搭載した NetApp AFF ストレージシステムと NVIDIA DGX システムを使用します。ONTAP AI は、業界標準の NFS ファイルプロトコルである 100Gb イーサネットを基盤としており、標準的なデータセンターテクノロジを使用して実装や管理のオーバーヘッドを軽減する、ハイパフォーマンスな ML / DL インフラを提供します。標準化されたネットワークとプロトコルを使用することで、 ONTAP AI をハイブリッドクラウド環境に統合しながら、運用の一貫性と簡易性を維持できます。解決策 AI は、事前検証済みのインフラ ONTAP として、導入にかかる時間とリスクを削減し、管理オーバーヘッドを大幅に削減することで、お客様はより短期間で価値を実現できます。

NVIDIA DeepOps のことです

DeepOps は NVIDIA が開発したオープンソースプロジェクトです。 Ansible を使用することで、ベストプラクティスに従って GPU サーバクラスタの導入を自動化できます。DeepOps はモジュール方式であり、さまざまな導入タスクに使用できます。このドキュメントとこの検証の演習では、 DeepOps を使用して、 GPU サーバワーカーノードで構成される Kubernetes クラスタを導入します。詳細については、を参照してください "DeepOps の Web サイト"。

NetApp Trident

Trident は、ネットアップが開発および管理しているオープンソースのストレージオーケストレーションツールで、 Kubernetes ワークロード向けの永続的ストレージの作成、管理、使用を大幅に簡易化します。Trident 自体は Kubernetes ネイティブのアプリケーションであり、 Kubernetes クラスタ内で直接実行されます。Trident を使用すると、 Kubernetes のユーザ（開発者、データサイエンティスト、 Kubernetes 管理者など）は、使い慣れた標準的な Kubernetes 形式で永続ストレージボリュームを作成、管理、操作できます。同時に、ネットアップの高度なデータ管理機能と、ネットアップテクノロジを基盤とするデータファブリックを活用できます。Trident は、複雑な永続的ストレージを抽象化して、消費を簡易化します。詳細については、を参照してください "Trident の Web サイト"。

NetApp StorageGRID

NetApp StorageGRID は、ユーザが S3 プロトコルを使用してアクセスできるシンプルなクラウド型ストレージを提供することで、これらのニーズを満たすように設計された Software-Defined オブジェクトストレージプラットフォームです。StorageGRID は、距離に関係なく、インターネットに接続されたサイト全体で複数のノードをサポートするように設計されたスケールアウトシステムです。StorageGRID のインテリジェントポリシーエンジンを使用すると、サイト間でイレイジャーコーディングオブジェクトを選択して地理的な耐障害性を確保したり、リモートサイト間でオブジェクトレプリケーションを行ったりすることで、 WAN アクセスのレイテンシを最小限に抑えることができます。StorageGRID は、この解決策にある優れたプライベートクラウドプライマリオブジェクトストレージデータレイクを提供します。

NetApp Cloud Volumes ONTAP の略

NetApp Cloud Volumes ONTAP データ管理ソフトウェアは、 AWS 、 Google Cloud Platform 、 Microsoft Azure などのパブリッククラウドプロバイダの柔軟性を活かして、ユーザデータの制御、保護、効率化を実現します。Cloud Volumes ONTAP は、 NetApp ONTAP ストレージソフトウェアを基盤としたクラウドネイティブなデータ管理ソフトウェアで、クラウドデータのニーズに対応する、汎用性に優れた優れたストレージプラットフォームをユーザに提供します。クラウドとオンプレミスで同じストレージソフトウェアを使用することで、ユーザはデータファブリックの価値を活用できます。まったく新しいデータ管理方法について IT 担当者をトレーニングする必要はありません。

ハイブリッドクラウドの導入モデルに関心があるお客様は、 Cloud Volumes ONTAP を使用することで、ほとんどのパブリッククラウドで同じ機能とクラス最高のパフォーマンスを実現し、一貫したシームレスなユーザエクスペリエンスをあらゆる環境で実現できます。

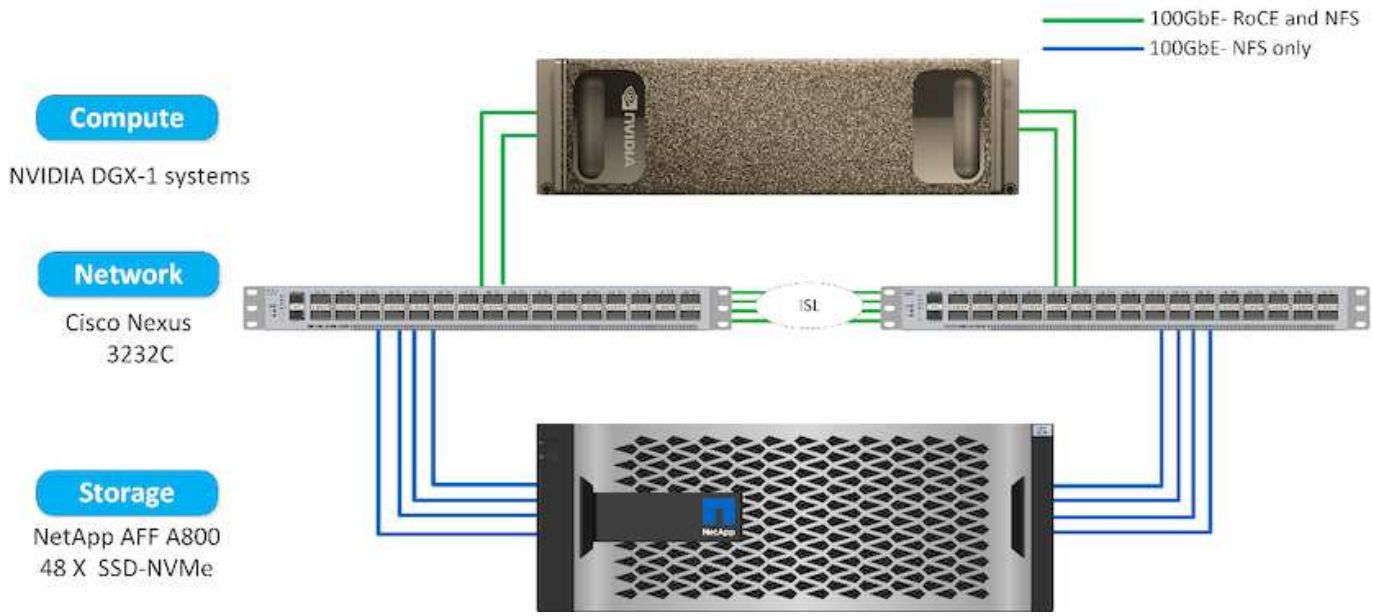
ハードウェアとソフトウェアの要件

このセクションでは、 ONTAP AI 解決策のテクノロジ要件について説明します。

ハードウェア要件

ハードウェア要件はお客様のワークロードによって異なりますが、 ONTAP AI は、大規模な ML/DL 運用向けに、単一の GPU からラックスケール構成まで、あらゆる規模のデータエンジニアリング、モデルトレーニング、本番環境推論に導入できます。ONTAP AI の詳細については、を参照してください "ONTAP AI Web サイト"。

この解決策は、コンピューティングには DGX-1 システム、ネットワーク接続には NetApp AFF A800 ストレージシステム、 Cisco Nexus 3232C を使用して検証しました。この検証で使用される AFF A800 は、ほとんどの ML/DL ワークロードで最大 10 台の DGX-1 システムをサポートできます。次の図に、この検証でモデルのトレーニングに使用する ONTAP AI トポロジを示します。



この解決策をパブリッククラウドに拡張するために、 Cloud Volumes ONTAP をクラウド GPU コンピューティングリソースと一緒に導入し、ハイブリッドクラウドデータファブリックに統合することで、お客様は特定のワークロードに適したリソースを自由に使用できます。

ソフトウェア要件

次の表に、この解決策検証で使用されるソフトウェアのバージョンを示します。

コンポーネント	バージョン
Ubuntu	18.04.4 LTS
NVIDIA DGX OS	4.4.0
NVIDIA DeepOps のことです	20.02.1
Kubernetes	1.15
Helm	3.1.0
cnvrg.io	3.0.0
NetApp ONTAP	9.6P4

この解決策検証では、 Kubernetes を DGX-1 システム上にシングルノードクラスタとして導入しました。大規模な導入の場合は、管理サービスの高可用性を実現し、 ML ワークロードと DL ワークロードに貴重な DGX リソースを確保するために、独立した Kubernetes マスターノードを導入する必要があります。

解決策の導入と検証の詳細

以降のセクションでは、解決策の導入と検証の詳細について説明します。

ONTAP AI の導入

ONTAP AI を導入するには、ネットワーク、コンピューティング、ストレージハードウェアのインストールと設定が必要です。ONTAP AI インフラの導入手順については、本ドキュメントでは説明していません。導入の詳細については、を参照してください "NVA-1121-deploy ： NetApp ONTAP AI 、 Powered by NVIDIA"。

この解決策検証では、 1 つのボリュームを作成して DGX-1 システムにマウントしました。その後、そのマウントポイントをコンテナにマウントして、トレーニング用のデータにアクセスできるようにしました。大規模な環境では、 NetApp Trident によってボリュームの作成とマウントが自動化されるため、管理上のオーバーヘッドが発生しないとともに、エンドユーザによるリソースの管理が可能になります。

Kubernetes の導入

NVIDIA DeepOps で Kubernetes クラスタを導入および設定するには、導入ジャンプホストから次のタスクを実行します。

1. の手順に従って、 NVIDIA DeepOps をダウンロードします "「はじめに」ページ" NVIDIA DeepOps GitHub サイトで入手できます。
2. の手順に従って、クラスタに Kubernetes を導入します 。 "Kubernetes 導入ガイド" NVIDIA DeepOps GitHub サイトで入手できます。

> DeepOps Kubernetes 環境を使用するには、 Kubernetes マスターノードとワーカーノードがすべて同じユーザである必要があります。

配備に失敗した場合は 'kubectl_localhost' の値を 'd eepops/config/group_vars/k8s-cluster.yml' で false に変更し ' 手順 2 を繰り返します'Copy kubectl binary to Ansible host' タスクは 'kubectl_localhost' の値が true の場合にのみ実行され ' メモリ使用に関する既知の問題がある FETCH Ansible モジュールに依存しますこのようなメモリ使用の問題により、原因がタスクを失敗させることがあります。メモリ問題が原因でタスクが失敗した場合は、以降の導入処理は正常に完了しません。

「 kubectl_localhost 」の値を「 false 」に変更した後で展開が正常に完了した場合、「 kubectl binary 」を Kubernetes マスターノードから配備ジャンプホストに手動でコピーする必要があります。特定のマスター・ノード上で 'kubectl binary' の場所を確認するには 'which kubectl' コマンドをそのノード上で直接実行します

cnvrg.io の展開

Helm を使用して cnvrg コアを導入します

Helm は、任意のクラスタ、オンプレミス、 Minikube 、または任意のクラウドクラスタ（ AKS 、 EKS 、 GKE など）を使用して、 cnvrg を迅速に導入する最も簡単な方法です。このセクションでは、 Kubernetes がインストールされたオンプレミス（ DGX-1 ）インスタンスに cnvrg がインストールされた方法について説明します。

前提条件

インストールを完了する前に、ローカルマシンに次の依存関係をインストールして準備する必要があります。

- Kubectl のように入力する
- Helm 3.x

- Kubernetes クラスタ 1.15 以降

Helm を使用して展開します

1. 最新の cnvrg Helm チャートをダウンロードするには、次のコマンドを実行します。

```
helm repo add cnvrg https://helm.cnvrg.io
helm repo update
```

2. cnvrg を導入する前に、クラスタの外部 IP アドレス、および cnvrg を導入するノードの名前が必要です。オンプレミスの Kubernetes クラスタに cnvrg を導入するには、次のコマンドを実行します。

```
helm install cnvrg cnvrg/cnvrg --timeout 1500s  --wait \ --set
global.external_ip=<ip_of_cluster> \ --set global.node=<name_of_node>
```

3. 「 helm install 」コマンドを実行します。すべてのサービスとシステムがクラスタに自動的にインストールされます。この処理には最大 15 分かかることがあります。
4. 「 helm install 」コマンドの所要時間は最大 10 分です。展開が完了したら、新しく展開した cnvrg の URL に移動するか、新しいクラスタを組織内のリソースとして追加します。「 helm' 」コマンドは正しい URL を通知します。

```
Thank you for installing cnvrg.io!
Your installation of cnvrg.io is now available, and can be reached via:
Talk to our team via email at
```

5. すべてのコンテナのステータスが「 Running 」または「 Complete 」の場合、 cnvrg は正常に展開されています。次のような出力が表示されます。

```
NAME                              READY   STATUS      RESTARTS   AGE
cnvrg-app-69fbb9df98-6xrgf                1/1     Running     0          2m
cnvrg-sidekiq-b9d54d889-5x4fc             1/1     Running     0          2m
controller-65895b47d4-s96v6               1/1     Running     0          2m
init-app-vs-config-wv9c4                  0/1     Completed   0          9m
init-gateway-vs-config-2zbpp              0/1     Completed   0          9m
init-minio-vs-config-cd2rg                0/1     Completed   0          9m
minio-0                                   1/1     Running     0          2m
postgres-0                                1/1     Running     0          2m
redis-695c49c986-kcbt9                    1/1     Running     0          2m
seeder-wh655                              0/1     Completed   0          2m
speaker-5sghr                             1/1     Running     0          2m
```

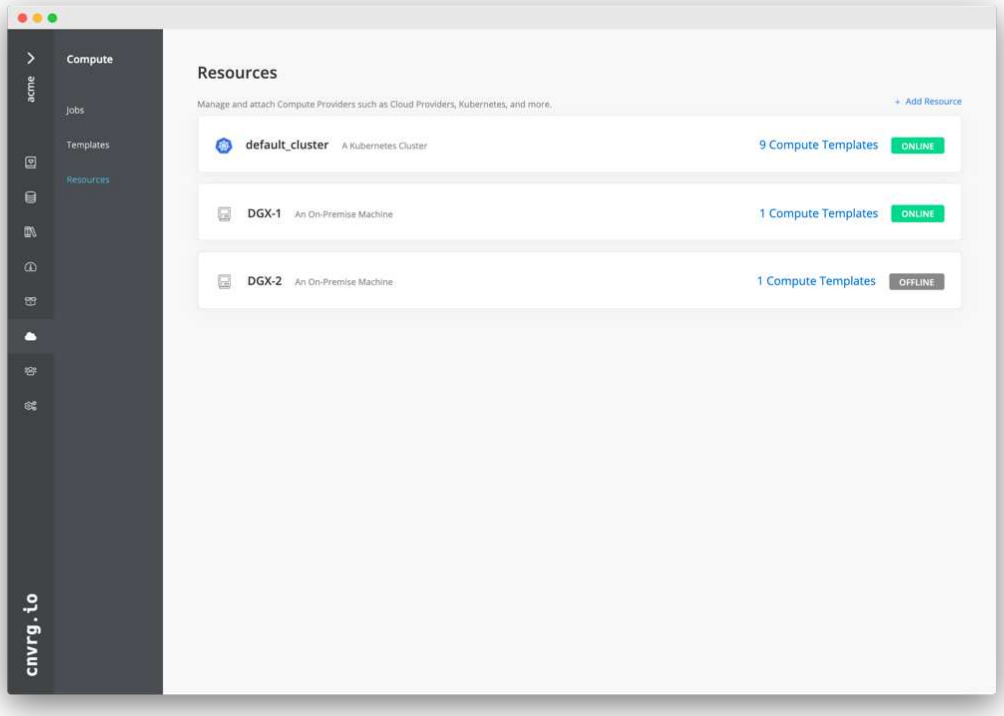
ResNet50 および胸部 **X** 線を使用したコンピュータビジョンモデルトレーニング データセット

NVIDIA DGX システムを基盤とする NetApp ONTAP AI アーキテクチャ上の Kubernetes セットアップに、cnvrg.io AI OS が導入されました。検証には、胸部 X 線の匿名画像からなる NIH 胸部 X 線データセットを使用しました。画像は PNG 形式でした。このデータは NIH クリニカルセンタおよびによって提供された は、から使用できます "NIH ダウンロードサイト"。250 GB のサンプルデータを 15 クラスの 627 、 615 イメージで使用しました。

データセットは cnvrg プラットフォームにアップロードされ、 NetApp AFF A800 ストレージシステムからの NFS エクスポートにキャッシュされました。

コンピューティングリソースをセットアップする

cnvrg アーキテクチャおよびメタスケジューリング機能により、エンジニアおよび IT プロフェッショナルは、異なるコンピューティングリソースを 1 つのプラットフォームに接続できます。今回のセットアップでは、ディープラーニングワークロードの実行用に導入されたクラスタ cnvrg を使用しました。追加のクラスタを接続する必要がある場合は、次のスクリーンショットに示すように、 GUI を使用してください。



データをロードします

cnvrg プラットフォームにデータをアップロードするには、 GUI または cnvrg CLI を使用します。大規模なデータセットの場合は、 CLI の使用を推奨します。 CLI は、多数のファイルを処理できる、拡張性と信頼性に優れた強力なツールです。

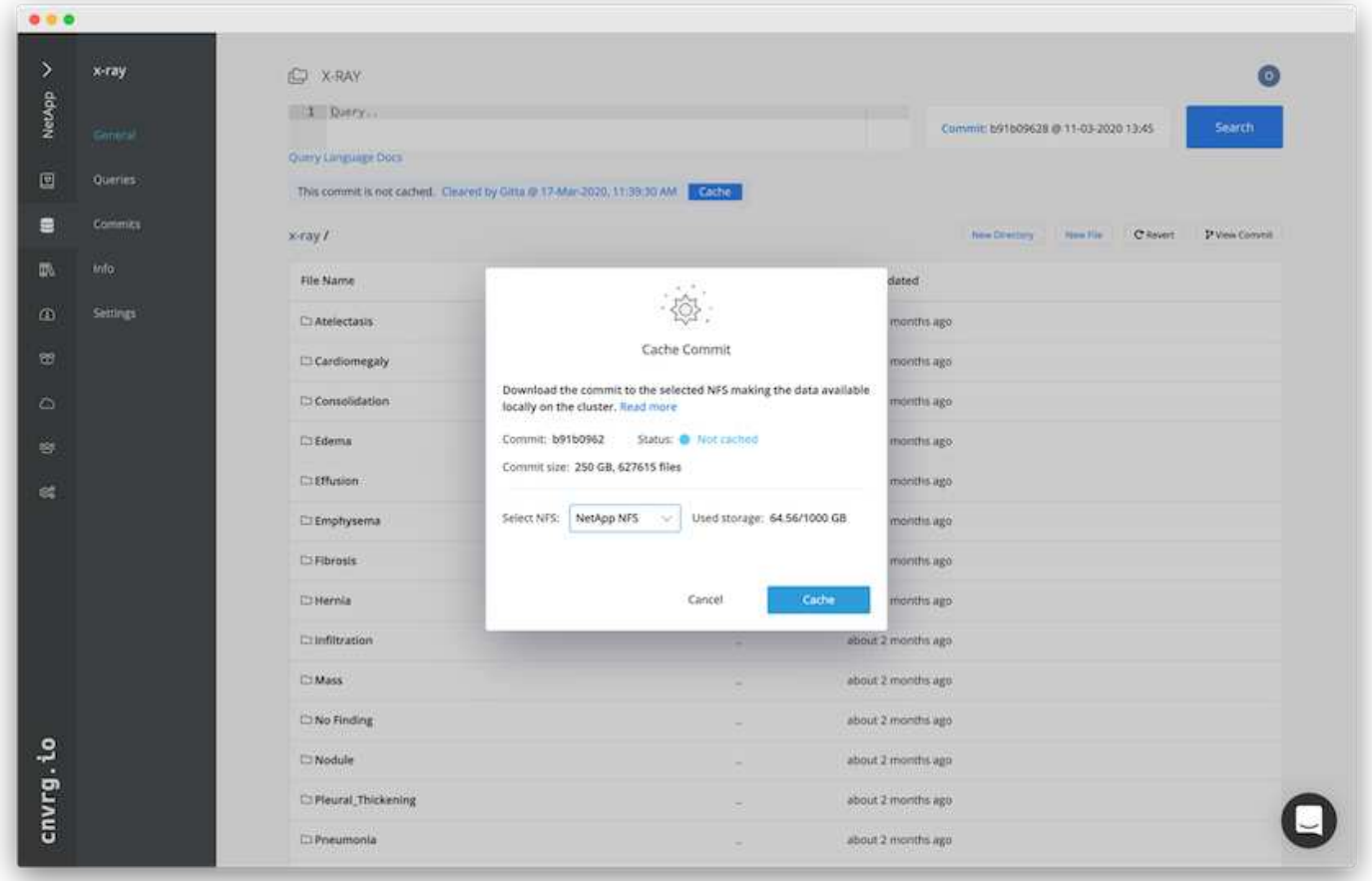
データをアップロードするには、次の手順を実行します。

1. をダウンロードします "cnvrg CLI"。
2. X 線ディレクトリに移動します。
3. 「 cnvrg data init 」コマンドを使用して、プラットフォーム内のデータセットを初期化します。

4. 「 cnvrg data sync 」コマンドを使用して、ディレクトリのすべての内容を中央のデータレイクにアップロードします。データが中央のオブジェクトストア（ StorageGRID 、 S3 、またはその他）にアップロードされたら、 GUI で参照できます。次の図は、ロードされた胸部 X 線線維症画像 PNG ファイルを示しています。さらに、 cnvrg は、ビルドしたすべてのモデルをデータバージョンに複製できるように、データをバージョン化します。

マッハデータ

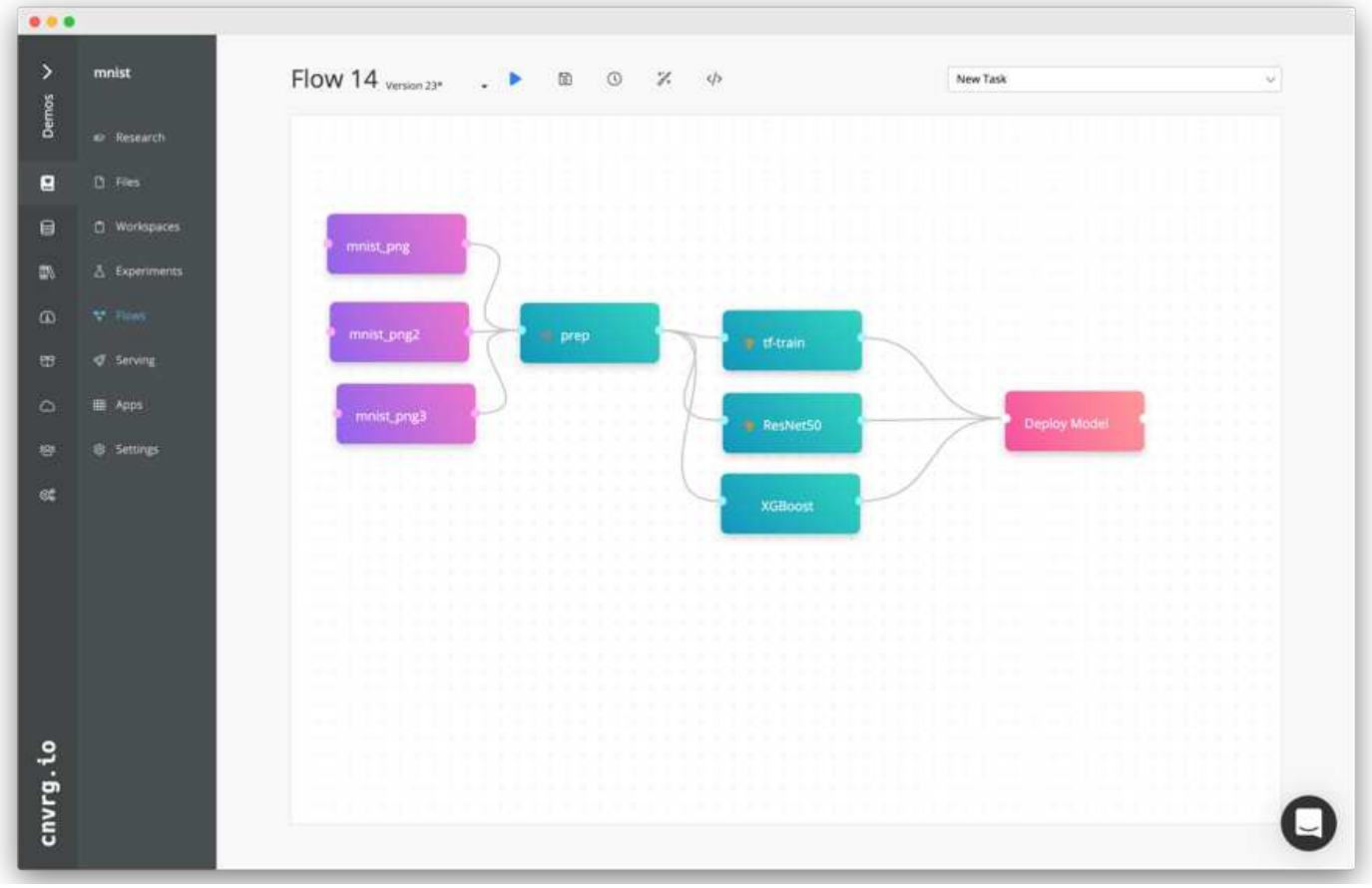
トレーニングを高速化し、モデルのトレーニングや実験ごとに 600k 以上のファイルをダウンロードしないようにするために、データを最初に中央のデータレイクオブジェクトストアにアップロードしたあとにデータキャッシュ機能を使用しました。



ユーザーが Cache をクリックすると、 cnvrg はリモートオブジェクトストアから特定のコミットでデータをダウンロードし、 ONTAP NFS ボリュームにキャッシュします。完了すると、データをすぐにトレーニングに利用できるようになります。さらに、データが数日間使用されていない場合（たとえば、モデルのトレーニングや探索など）、 cnvrg は自動的にキャッシュをクリアします。

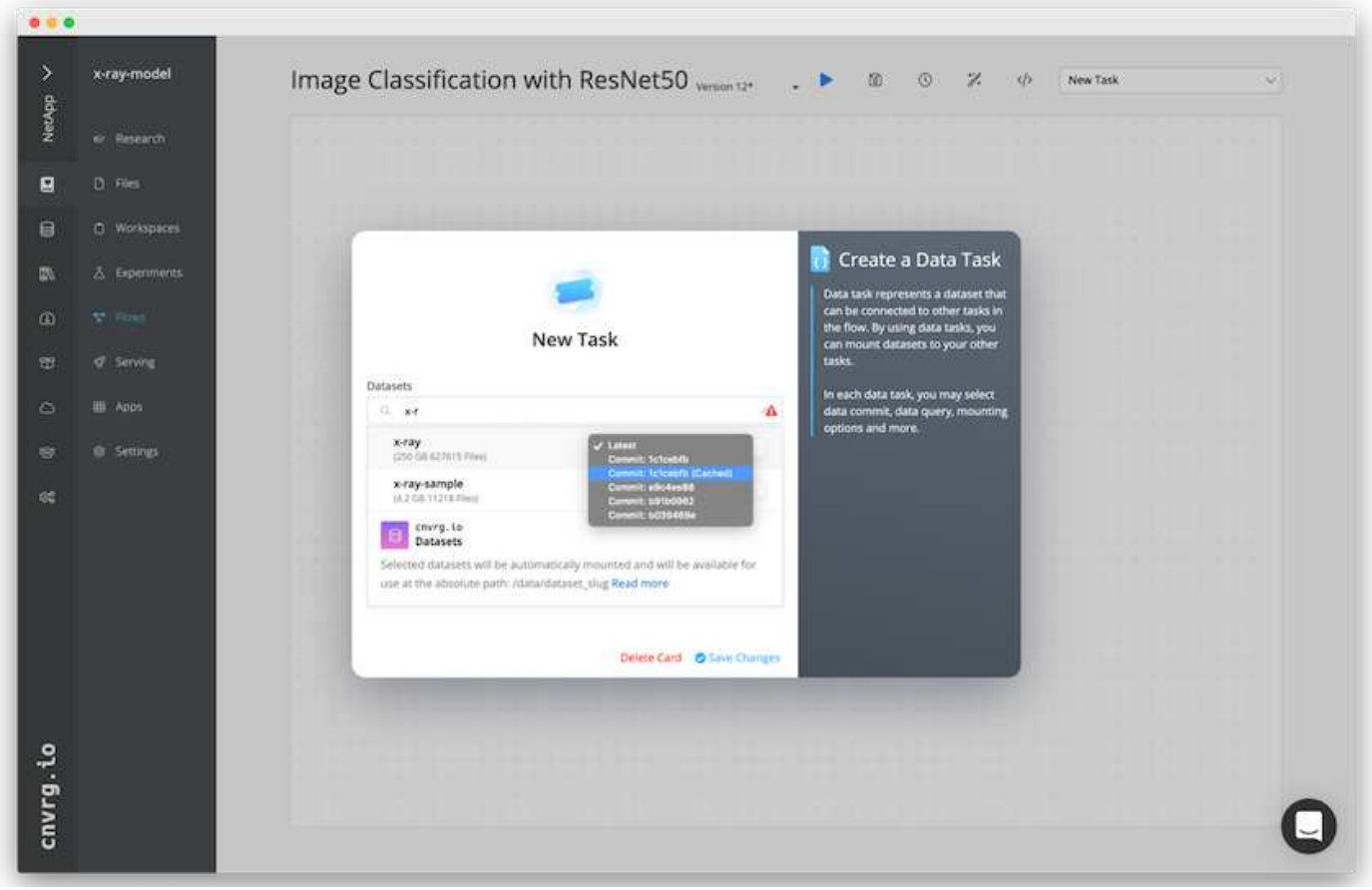
キャッシュデータで **ML** パイプラインを構築

cnvrg フローを使用すると、本番 ML パイプラインを簡単に構築できます。フローは柔軟性が高く、あらゆる種類の ML ユースケースに対応し、 GUI またはコードを使用して作成できます。フロー内の各コンポーネントは、異なる Docker イメージを使用して異なるコンピューティングリソース上で実行できるため、ハイブリッドクラウドを構築し、 ML パイプラインを最適化できます。



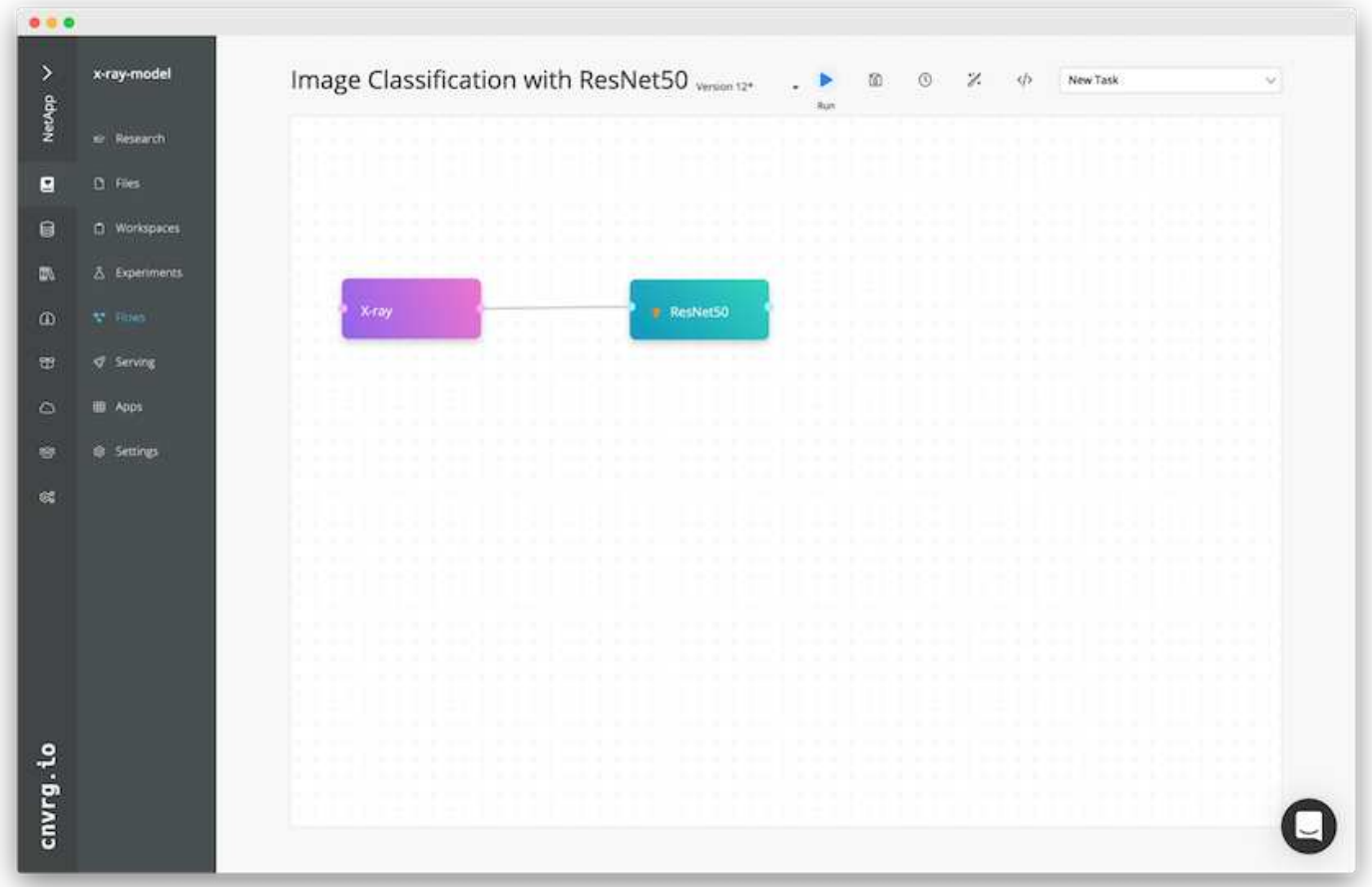
胸部 **X** 線フローの構築：データの設定

新しく作成したフローにデータセットを追加しました。データセットを追加する際には、特定のバージョン（commit ）を選択し、キャッシュされたバージョンが必要かどうかを指定できます。この例では、キャッシュされたコミットを選択しました。



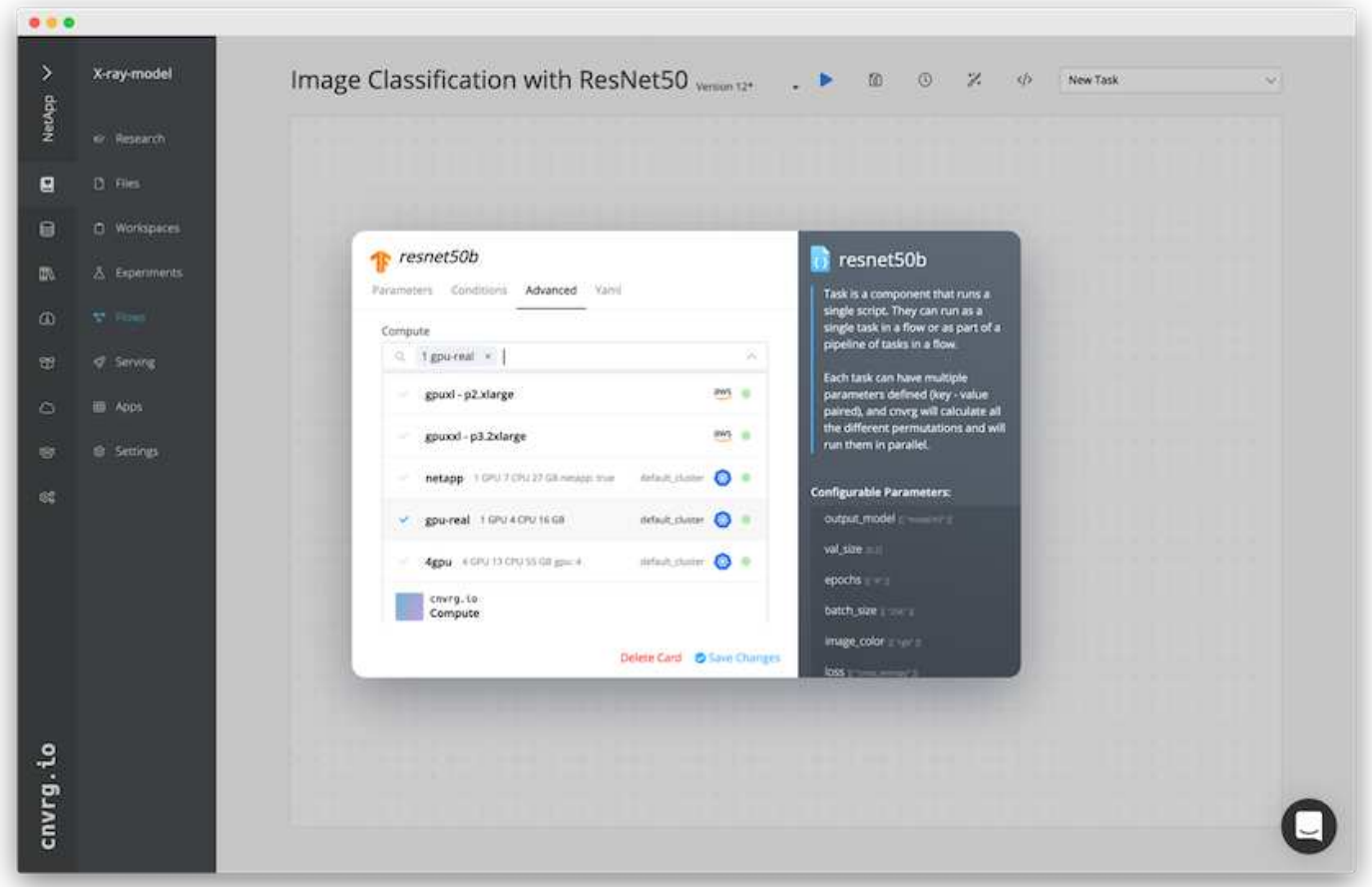
胸部 **X** 線フローの構築：トレーニングモデルの設定： **ResNet50**

パイプラインでは、任意の種類のカスタムコードを追加できます。cnvrg には、再利用可能な ML コンポーネントコレクションである AI ライブラリもあります。AI ライブラリには、アルゴリズム、スクリプト、データソースなど、あらゆる ML やディープラーニングフローで使用できるソリューションがあります。この例では、 ResNet50 の事前ビルドモジュールを選択しました。batch_size ： 128 、 epochs ： 10 などのデフォルトパラメータを使用しました。これらのパラメータは AI ライブラリのドキュメントで確認できます。次のスクリーンショットは、 X 線データセットが ResNet50 に接続された新しいフローを示しています。



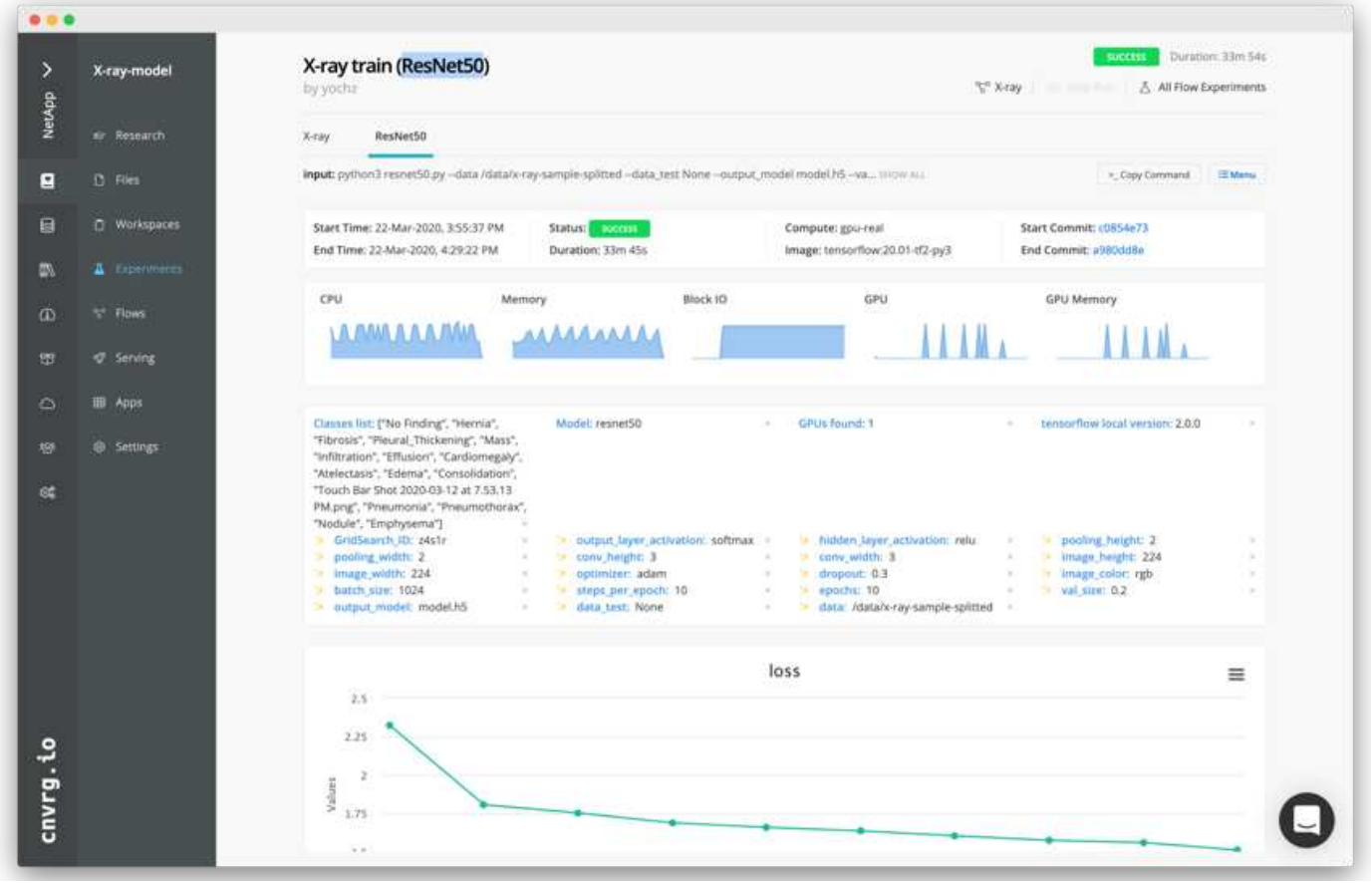
ResNet50 の計算リソースを定義します

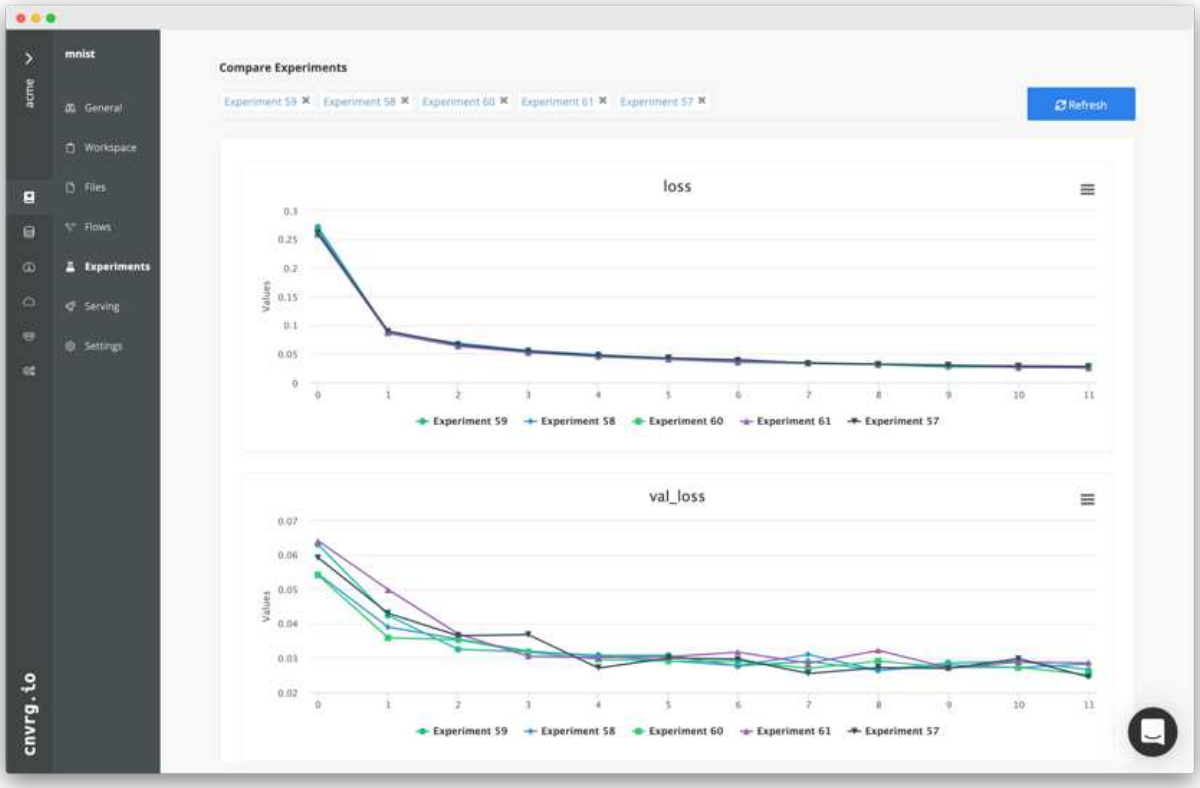
cnvrg フロー内の各アルゴリズムまたはコンポーネントは、異なる Docker イメージを使用して、異なるコンピューティングインスタンス上で実行できます。セットアップでは、 NetApp ONTAP AI アーキテクチャを採用した NVIDIA DGX システムでトレーニングアルゴリズムを実行したいと考えていました。次の図では、「 GPU - REAL 」を選択しました。これは、オンプレミスクラスタのコンピューティングテンプレートであり、仕様です。また、テンプレートのキューを作成し、複数のテンプレートを選択しました。このようにして 'GPU 実数のリソースを割り当てることができない場合 (たとえば ' 他のデータ・サイエンティストがリソースを使用している場合) は ' クラウド・プロバイダ・テンプレートを追加して ' 自動クラウド・バーストを有効にできます次のスクリーンショットは、 ResNet50 のコンピューティングノードとしての GPU 実数の使用を示しています。



結果の追跡と監視

フローが実行されると、 cnvrg はトラッキングおよびモニタリングエンジンをトリガーします。フローの各実行は自動的に文書化され、リアルタイムで更新されます。ハイパーパラメータ、指標、リソース使用率（ GPU 利用率など）、コードバージョン、アーティファクト、ログ また、次の 2 つのスクリーンショットに示すように、 ［テスト］ セクションで自動的に使用できるようになります。


X-ray-model
X-ray train (ResNet50)
by yochz
Research
Files
Workspaces
Experiments
Flows
Serving
Apps
Settings
cnvrg.io
Start Time: 22-Mar-2020, 3:55:37 PM
End Time: 22-Mar-2020, 4:29:22 PM
Status:
Duration: 33m 45s
Compute: gpu-real
Image: tensorflow:20.01-tf2-py3
Start Commit: c0854e73
End Commit: a980dd8e
CPU
Memory
Block IO
GPU
GPU Memory
loss
Values


mnist
Compare Experiments
Refresh
General
Workspace
Files
Flows
Experiments
Serving
Settings
cnvrg.io
loss
val_loss
Values
Experiment 59
Experiment 58
Experiment 60
Experiment 61
Experiment 57

まとめ

NetApp と cnvrg.io はパートナーとして提携し、 ML および DL ソフトウェア開発向けの包括的なデータ管理解決策をお客様に提供しています。ONTAP AI は、あらゆる規模の運用に対応できる高性能なコンピューティングとストレージを提供します。 cnvrg.io ソフトウェアは、データサイエンスのワークフローを合理化し、リソース利用率を向上させます。

謝辞

- ネットアップテクニカルマーケティングエンジニア、 Mike Oglesby 氏
- ネットアップシニアテクニカルディレクター Santosh Rao 氏

追加情報の検索場所

このドキュメントに記載されている情報の詳細については、次のリソースを参照してください。

- Cnvrg.io （ "https://cnvrg.io"）：
  - Cnvrg コア（無償の ML プラットフォーム）

    https://cnvrg.io/platform/core
  - Cnvrg のドキュメント

    "https://app.cnvrg.io/docs"
- NVIDIA DGX-1 サーバ：
  - NVIDIA DGX-1 サーバ

    https://www.nvidia.com/en-us/data-center/dgx-1/
  - NVIDIA Tesla V100 Tensor コア GPU

    https://www.nvidia.com/en-us/data-center/tesla-v100/
  - NVIDIA GPU Cloud （ NGC ）

    https://www.nvidia.com/en-us/gpu-cloud/
- NetApp AFF システム：
  - AFF データシート

    https://www.netapp.com/us/media/d-3582.pdf
  - AFF 向け NetApp FlashAdvantage プログラム

    https://www.netapp.com/us/media/ds-3733.pdf
  - ONTAP 9.x のドキュメント

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

  - NetApp FlexGroup テクニカルレポート

    https://www.netapp.com/us/media/tr-4557.pdf

- コンテナ向けのネットアップの永続的ストレージ：
  - NetApp Trident

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

- NetApp Interoperability Matrix を参照してください
  - NetApp Interoperability Matrix Tool で確認できます

    http://support.netapp.com/matrix

- ONTAP AI ネットワーク：
  - Cisco Nexus 3232C スイッチ

    https://www.cisco.com/c/en/us/products/switches/nexus-3232c-switch/index.html

  - Mellanox Spectrum 2000 シリーズスイッチ

    http://www.mellanox.com/page/products_dyn?product_family=251&mtag=sn2000

- ML フレームワークとツール：
  - 大理

    https://github.com/NVIDIA/DALI

  - TensorFlow ：あらゆる環境に対応するオープンソースの機械学習フレームワーク

    https://www.tensorflow.org/

  - Horovod ： Uber が開発したオープンソースの TensorFlow 用分散学習フレームワーク

    https://eng.uber.com/horovod/

  - コンテナランタイムエコシステムでの GPU の有効化

    https://devblogs.nvidia.com/gpu-containers-runtime/

  - Docker です

    https://docs.docker.com

  - Kubernetes

    https://kubernetes.io/docs/home/

  - NVIDIA DeepOps のことです

https://github.com/NVIDIA/deepops

- クビフロー

  http://www.kubeflow.org/

- Jupyter Notebook Server の 2 つのツールを使用

  http://www.jupyter.org/

- データセットとベンチマーク：
  - NIH 胸部 X 線データセット

    https://nihcc.app.box.com/v/ChestXray-NIHCC

  - Xiaosong Wang 、 Yifan Peng 、 Le Lu 、 Zhiyong Lu 、 Mohammadhadi Bagheri 、 ロナルド・サマーズ、 ChestX-Ray8 ：『 Hospital scale Chest X-ray Database and Benchmarks on weakly Supervised Classification and Localization of Common Thorax Diseases 、 IEEE CVR 、 pp3462-3471 、 2017TR-4841-0620

Lenovo ThinkSystem-解決策 Design を使用したエッジネットアップでの AI 推論

TR-4886 ： Lenovo ThinkSystem-解決策 Design を使用したエッジネットアップでの AI 推論

Lenovo 、 Miroslav Hodak 、 Sathish Thyagarajan 氏

まとめ

先進的なドライバーアシスタンスシステム（ADAS ）、インダストリー 4.0 、スマートシティ、モノのインターネット（ IoT ）など、いくつかの新しいアプリケーションシナリオでは、ほぼゼロのレイテンシで継続的なデータストリームを処理する必要があります。このドキュメントでは、こうした要件を満たすエッジ環境のネットアップストレージコントローラと Lenovo ThinkSystem サーバに GPU ベースの人工知能（ AI ）推論を導入するためのコンピューティングとストレージのアーキテクチャについて説明します。また、このドキュメントでは、業界標準の MLPerf Inference ベンチマークのパフォーマンスデータも提供し、 NVIDIA T4 GPU を搭載したエッジサーバ上のさまざまな推論タスクを評価します。オフライン、単一ストリーム、マルチストリームの推論のシナリオのパフォーマンスを調査し、コスト効率の高い共有ネットワークストレージシステムを使用したアーキテクチャはハイパフォーマンスであり、複数のエッジサーバのデータとモデルを一元的に管理できることを示します。

はじめに

企業は、ネットワークエッジで大量のデータを生成するケースが増えています。スマートセンサーや IoT データから最大限の価値を引き出すために、企業はエッジコンピューティングを可能にするリアルタイムのイベントストリーミング解決策を求めています。そのため、データセンターの外部にあるエッジでは、処理能力の高い作業がますます実行されるようになっています。AI 推論は、この傾向の推進要因の 1 つです。特にアクセラレータを使用する場合は、エッジサーバがこれらのワークロードに十分な処理能力を発揮しますが、ストレージが制限されることが多いのは、特にマルチサーバ環境では問題です。このドキュメントでは、共有ストレージシステムをエッジ環境に導入する方法と、パフォーマンスに影響を与えずに AI 推論ワークロードにどのようなメリットがあるかを説明します。

このドキュメントでは、エッジでの AI 推論向けのリファレンスアーキテクチャについて説明します。複数の Lenovo ThinkSystem エッジサーバとネットアップストレージシステムを組み合わせることで、導入と管理が容易な解決策を構築これは、複数のカメラと産業用センサーを備えた工場フロア、小売取引における POS シ

ステム、自律走行車の視覚的な異常を識別するフル・セルフ・ドライビング（ FSD ）システムなど、さまざまな状況での実践的な展開のための基本ガイドとなることを目的としています。

本ドキュメントでは、 Lenovo ThinkSystem SE350 Edge Server とエントリレベルの NetApp AFF および EF シリーズストレージシステムで構成された、コンピューティングとストレージの構成のテストと検証について説明します。リファレンスアーキテクチャは、 AI 導入向けの効率的でコスト効率に優れた解決策を提供すると同時に、 NetApp ONTAP と NetApp SANtricity データ管理ソフトウェアを使用して、包括的なデータサービス、統合データプロテクション、シームレスな拡張性、クラウド対応データストレージを提供します。

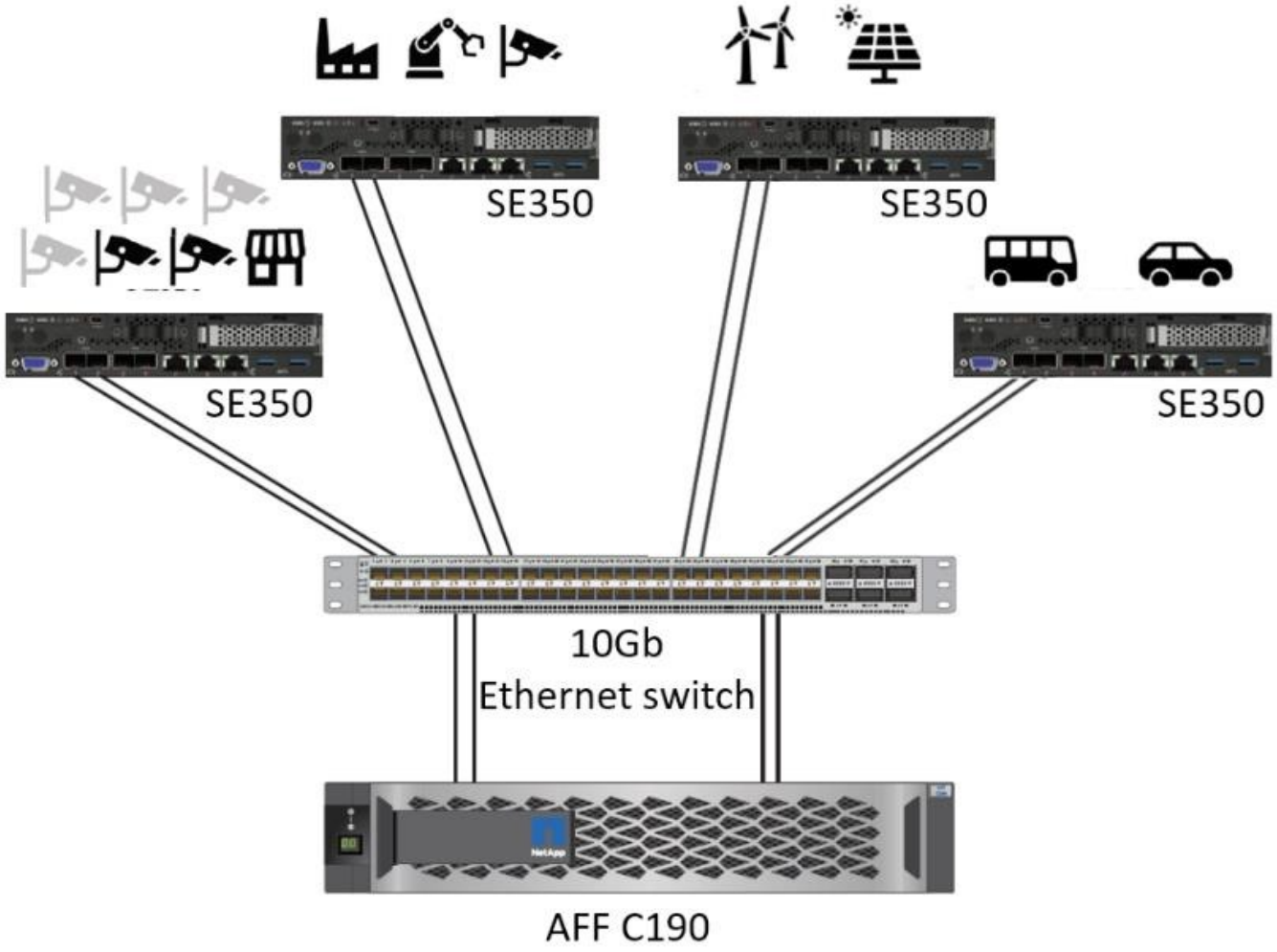
対象読者

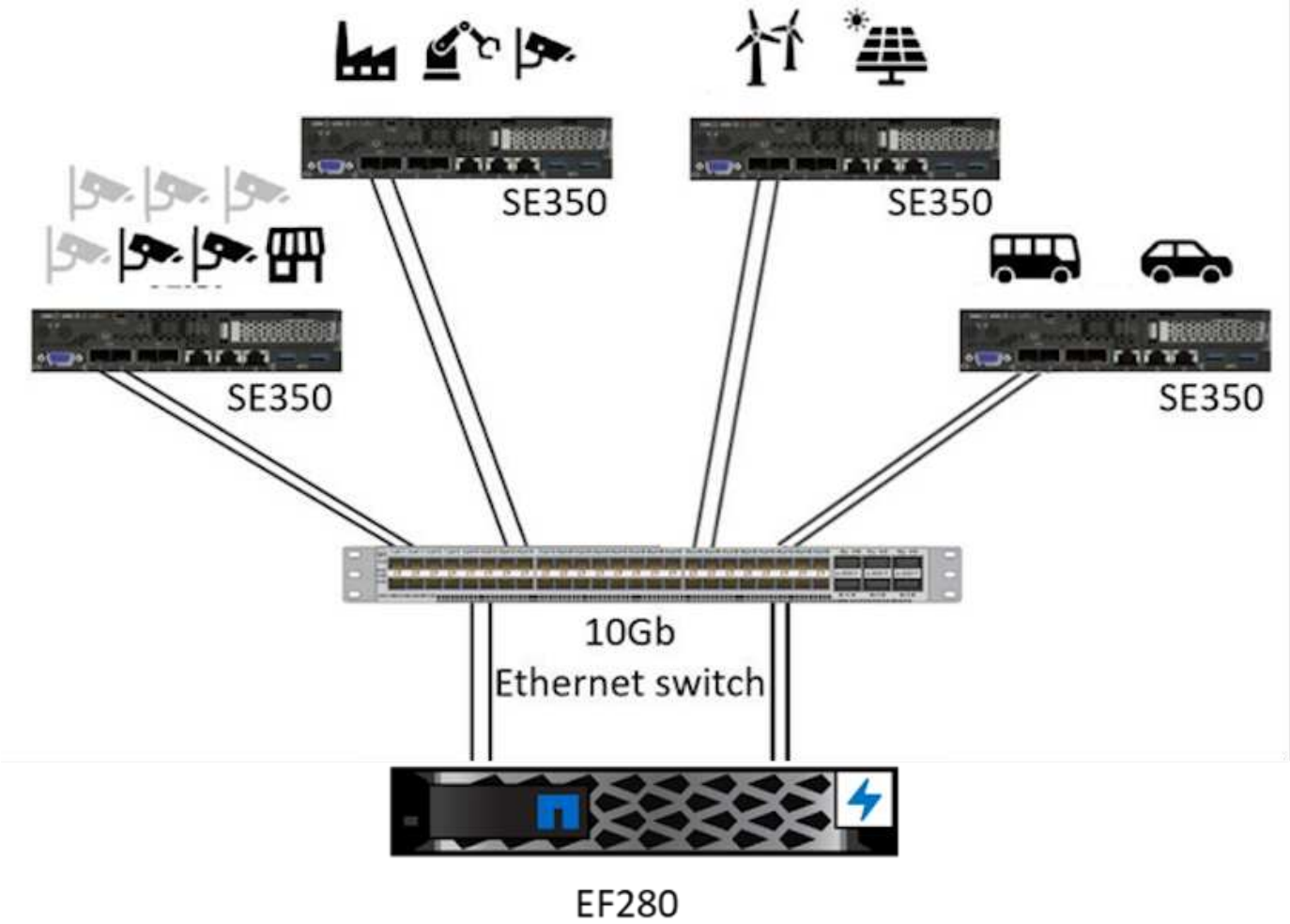
本ドキュメントは、次のような方を対象としています。

- エッジで AI を生産するビジネスリーダーやエンタープライズアーキテクト。
- データサイエンティスト、データエンジニア、 AI / 機械学習（ ML ）研究者、 AI システムの開発者
- AI / ML モデルとアプリケーションの開発のためのソリューションを設計するエンタープライズアーキテクト。
- ディープラーニング（ DL ）モデルや ML モデルを効率的に導入する方法を探しているデータサイエンティストと AI エンジニア。
- エッジ推論モデルの導入と管理を担当するエッジデバイスマネージャとエッジサーバ管理者。

解決策アーキテクチャ

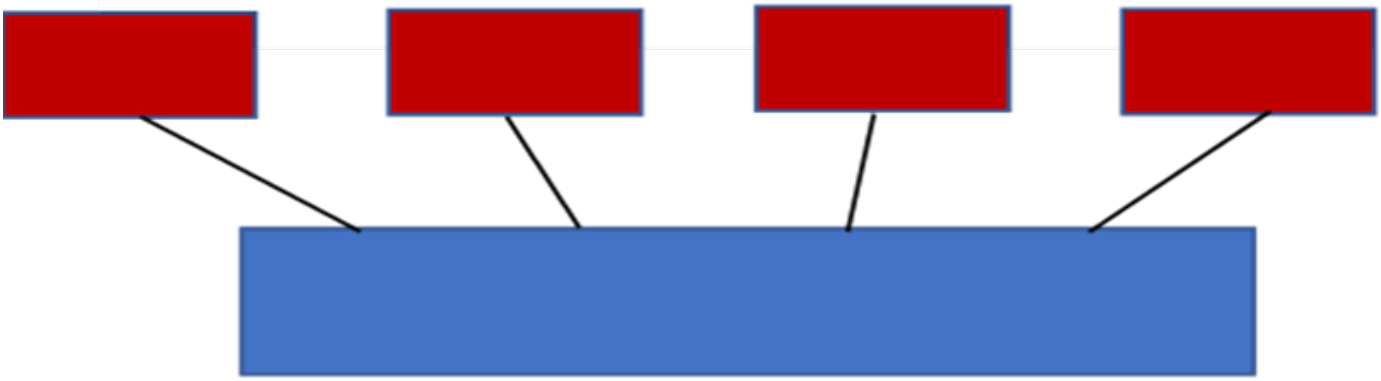
この Lenovo ThinkSystem サーバと NetApp ONTAP または NetApp SANtricity ストレージ解決策は、従来の CPU に加えて GPU の処理能力を使用して大規模なデータセットで AI 推論を処理するように設計されています。この検証では、次の 2 つの図に示すように、単一または複数の Lenovo SR350 エッジサーバを 1 つの NetApp AFF ストレージシステムと相互接続したアーキテクチャを使用して、パフォーマンスと最適なデータ管理を実現します。


SE350
SE350
SE350
SE350
10Gb
Ethernet switch
AFF C190



次の図の論理アーキテクチャの概要は、このアーキテクチャのコンピューティング要素とストレージ要素の役割を示しています。具体的には、次の情報が表示されます。

- カメラやセンサーなどから受信したデータを推論しているエッジコンピューティングデバイス。
- 複数の用途に使用できる共有ストレージ要素：
  - 推論モデルや推論の実行に必要なその他のデータを一元的に格納できます。コンピューティングサーバはストレージに直接アクセスし、ネットワーク全体で推論モデルを使用します。ローカルにコピーする必要はありません。
  - 更新されたモデルはここにプッシュされます。
  - エッジサーバーが受信する入力データをアーカイブして、後で分析します。たとえば、エッジデバイスがカメラに接続されている場合、ストレージエレメントはカメラでキャプチャされたビデオを保持します。



赤	青
Lenovo コンピューティングシステム	NetApp AFF ストレージシステム
カメラやセンサーなどからの入力で推論を実行するエッジデバイス。	推論モデルとエッジデバイスからのデータを保持する共有ストレージを使用して、後から分析することができます。

このネットアップと Lenovo 解決策は、主に次のようなメリットをもたらします。

- エッジでの GPU アクセラレーションコンピューティング。
- 共有ストレージからバックアップおよび管理される複数のエッジサーバの導入。
- 堅牢なデータ保護により、データ損失ゼロで目標復旧時点（ RPO ）と目標復旧時間（ RTO ）を達成
- NetApp Snapshot コピーとクローンでデータ管理を最適化し、開発ワークフローを合理化

このアーキテクチャの使用方法

本ドキュメントでは、提案アーキテクチャの設計とパフォーマンスを検証します。ただし、ネットアップでは、コンテナ、ワークロード、モデル管理、クラウドやデータセンターとのデータ同期など、特定のソフトウェアレベルの要素は導入シナリオに固有のものであるため、テストは実施していません。ここには複数の選択肢があります。

コンテナ管理レベルでは、 Kubernetes コンテナ管理が最適な選択肢であり、エンタープライズ環境に適した完全アップストリームバージョン（ Canonical ）または変更バージョン（ Red Hat ）のどちらでも十分にサポートされています。。 "NetApp AI コントロールプレーン" NetApp Trident と新たに追加された Trident を使用しています "NetApp DataOps ツールキット" トレーサビリティ、データ管理機能、インターフェイス、ツールが組み込まれており、データサイエンティストやデータエンジニアはネットアップストレージと統合できます。Kubernetes 向け ML ツールキットである Kubeflow は、 TensorFlow サービスや NVIDIA Triton Inference Server などの複数のプラットフォームで、モデルのバージョン管理と KFServing をサポートするほか、 AI 機能も追加します。もう 1 つの選択肢は NVIDIA EGX プラットフォームです。このプラットフォームは、 GPU 対応 AI 推論コンテナのカタログにアクセスしながら、ワークロード管理を提供します。ただし、これらのオプションを使用するには、本番環境に移行するための多大な労力と専門知識が必要になる場合があります。また、サードパーティの独立系ソフトウェアベンダー（ ISV ）やコンサルタントの支援が必要になる場合もあります。

解決策エリア

AI 推論とエッジコンピューティングの主なメリットは、デバイスがレイテンシなしで高品質のデータを計算、処理、分析できることです。このドキュメントで説明するエッジコンピューティングのユースケースの例は非常に多くありますが、ここではいくつかの重要な例を示します。

自動車：自律走行車

従来のエッジコンピューティングの図は、自律走行車（ AV ）の先進ドライバーアシスタンスシステム（ ADAS ）にあります。ドライバーのいない自動車の AI は、カメラやセンサーからの大量のデータを迅速に処理して、安全性を強化する必要があります。物体と人間の間を解釈するのに時間がかかりすぎると、生命や死亡を意味することがあります。そのため、可能な限り車両の近くでそのデータを処理できることが重要です。この場合、 1 つ以上のエッジコンピュートサーバがカメラ、レーダー、 LiDAR などのセンサーからの入力を処理し、共有ストレージには推論モデルが保持されてセンサーからの入力データが格納されます。

ヘルスケア：患者のモニタリング

AI とエッジコンピューティングがもたらす最大の影響の 1 つは、在宅ケアと集中治療ユニット（ ICU ）の両方において、慢性疾患の患者の継続的なモニタリングを強化できることです。インスリンレベル、呼吸、神経学的活性、心リズム、および消化管機能をモニターするエッジデバイスからのデータは、患者の生命を救うための時間が限られているため、ただちに作用する必要のあるデータを瞬時に分析する必要があります。

小売：現金払い

エッジコンピューティングは AI と ML を強化することで、小売企業はチェックアウト時間を短縮し、足のトラフィックを増加させることができます。キャッシュレスシステムは、次のようなさまざまなコンポーネントをサポートします。

- 認証とアクセス：物理的な買い物客を検証済みのアカウントに接続し、小売店のスペースへのアクセスを許可する。
- インベントリの監視：センサー、 RFID タグ、コンピューター・ビジョン・システムを使用して、買い物客による商品の選択や選択解除を確認できます。

  ここで ' 各エッジ・サーバが各チェックアウト・カウンタを処理し ' 共有ストレージ・システムが中央の同期ポイントとして機能します

金融サービス：キオスクでの人間の安全と不正防止

銀行業界では、 AI とエッジコンピューティングを活用して、パーソナライズされた銀行業務を革新し、創出しています。リアルタイムのデータ分析と AI 推論を使用したインタラクティブなキオスクにより、 ATM は顧客がお金を引き出すのを支援できるだけでなく、カメラからキャプチャされた画像を介してキオスクをプロアクティブに監視し、人間の安全や不正行為に対するリスクを特定できるようになりました。このシナリオでは、エッジコンピューティングサーバと共有ストレージシステムが対話型のキオスクやカメラに接続されて、銀行が AI 推論モデルでデータを収集して処理できるようにします。

製造： **Industry 4.0**

産業革命の 4 つ目（インダストリー 4.0 ）は、スマートファクトリーや 3D プリントなどの新たなトレンドとともに始まっています。データ主導の未来に備えるために、大規模な機械間（ M2M ）通信と IoT が統合されており、人間の介入なしに自動化を強化します。製造はすでに高度に自動化されており、 AI 機能の追加は長期的なトレンドの自然な流れを続けています。AI により、コンピュータビジョンやその他の AI 機能を活用して自動化できる運用を自動化できます。品質管理や、人間のビジョンや意思決定に依存するタスクを自動化して、工場の現場で組み立てライン上の材料を迅速に分析し、製造工場が必要とする ISO 規格の安全性と品質管理に適合できるようにすることができます。ここでは、各コンピュートエッジサーバが、製造プロセスを監視する一連のセンサーと、更新された推論モデルに必要に応じて共有ストレージにプッシュされます。

通信：地殻検出、タワー検査、およびネットワーク最適化

電気通信業界は、コンピュータビジョンと AI 技術を使用して、錆を自動的に検出し、腐食を含む基地局を特定する画像を処理しているため、さらなる検査が必要です。最近では、ドローン画像と AI モデルを使用して、塔の異なる領域を特定し、錆、表面の亀裂、腐食を分析しています。通信インフラやセルタワーを効率的に検査し、定期的に劣化を評価し、必要に応じて迅速に修復できる AI テクノロジの需要は高まり続けています。

さらに、通信業界で新たに登場したユースケースとして、 AI と ML のアルゴリズムを使用して、データトラフィックパターンの予測、 5G 対応デバイスの検出、 MIMO （複数入力 / 複数出力）エネルギー管理の自動化と強化が挙げられます。MIMO ハードウェアは、ネットワーク容量を増やすために無線タワーで使用されていますが、これには追加のエネルギーコストが伴います。セルサイトに導入された「 MIMO スリープモード」用の ML モデルは、無線機の効率的な使用を予測し、モバイルネットワークオペレータ（ MNO ）のエネルギー消費コストを削減するのに役立ちます。AI 推論とエッジコンピューティングのソリューションは、 MNO がデータセンターにやり取りするデータ量を削減し、 TCO を削減し、ネットワーク運用を最適化し、エンドユーザの全体的なパフォーマンスを向上させるのに役立ちます。

テクノロジの概要

このセクションでは、この AI 解決策の技術基盤について説明します。

NetApp AFF システム

最先端の NetApp AFF ストレージシステムにより、 AI 推論をエッジで導入することで、業界をリードするパフォーマンス、卓越した柔軟性、クラウド統合、業界最高クラスのデータ管理機能を備えたエンタープライズストレージの要件を満たすことができます。ネットアップの AFF システムはフラッシュに特化して設計されており、ビジネスクリティカルなデータの高速化、管理、保護に役立ちます。

- エントリレベルの NetApp AFF ストレージシステムは、 FAS2750 のハードウェアと SSD フラッシュメディアに基づいています
- HA 構成の場合は 2 台のコントローラ

ネットアップのエントリレベルの AFF C190 ストレージシステムは、次の機能をサポートしています。

- 960GB SSD を最大で 24 本搭載できます

- 次の 2 つの構成が可能です
  - イーサネット（ 10GbE ）： 10GBASE-T （ RJ-45 ）ポート × 4
  - ユニファイド（ 16Gb FC または 10GbE ）：ユニファイドターゲットアダプタ 2 （ UTA2 ）ポート × 4
- 最大 50.5TB の実効容量

> NAS ワークロードの場合、エントリレベルの AFF C190 システム 1 台で、シーケンシャルリードの場合は 4.4GBps 、スモールランダムリードの場合は 230K IOPS が 1 ミリ秒以下のレイテンシでサポートされます。

NetApp AFF A220

ネットアップは、他のエントリレベルストレージシステムも提供しています。このシステムは、大規模な環境にも対応できる優れたパフォーマンスと拡張性を提供します。NAS ワークロードの場合、 1 つのエントリレベルの AFF A220 システムで次のことがサポートされます。

- シーケンシャルリードのスループットは 6.2GBps です
- 1 ミリ秒以下のレイテンシでスモールランダムリードの IOPS 値 375K
- 960GB 、 3.8TB 、 7.6TB の SSD の最大ドライブ数： 144x
- AFF A220 は、 1PB を超える実効容量にまで拡張できます

NetApp AFF A250

- 最大実効容量は 35PB で、最大スケールアウト構成は 2~24 ノード（ HA ペア × 12 ）
- AFF A220 と比較して、パフォーマンスが 45% 以上向上します
- 440 万 IOPS のランダムリード： 1 ミリ秒
- 最新のネットアップ ONTAP リリース ONTAP 9.8 を基盤としています
- HA とクラスタインターコネクトに 25GB のイーサネットを利用しています

NetApp E シリーズ EF システム

EF シリーズは、エントリレベルとミッドレンジのオールフラッシュ SAN ストレージアレイファミリーです。データへのアクセスを高速化し、 NetApp SANtricity ソフトウェアを使用してデータから迅速に価値を引き出すことができます。SAS と NVMe の両方のフラッシュストレージを搭載し、低コストで卓越した IOPS 、 100 マイクロ秒未満の応答時間、最大 44GBps の帯域幅を実現します。これらのシステムは、混在ワークロードや、 AI 推論やハイパフォーマンスコンピューティング（ HPC ）などの要件の厳しいアプリケーションに最適です。

次の図は、 NetApp EF280 ストレージシステムを示しています。

NetApp EF280

- 32Gb / 16Gb FC 、 25Gb / 10Gb iSCSI 、 12Gb SAS に対応しています
- 最大実効容量は 96 本のドライブで合計 1.5PB です
- 10Gbps のスループット（シーケンシャルリード）
- 30 万 IOPS （ランダムリード）
- NetApp EF280 は、ネットアップポートフォリオの中で最も低コストのオールフラッシュアレイ（ AFA ）です

NetApp EF300

- 合計容量 367TB の NVMe SSD を 24 本搭載
- 拡張オプションは合計で 240x NL-SAS HDD 、 96x SAS SSD 、またはその組み合わせです
- 100Gb NVMe/IB 、 NVMe/RoCE 、 iSER/IB 、および SRP/IB
- 32Gb NVMe/FC 、 FCP
- 25Gb iSCSI です
- 20GBps （シーケンシャルリード）
- 670K IOPS （ランダムリード）

詳細については、を参照してください "NetApp EF シリーズ NetApp EF シリーズオールフラッシュアレイ EF600 、 F300 、 EF570 、 EF280 のデータシート"。

NetApp ONTAP 9.

ONTAP 9.8.1 は、ネットアップの最新世代のストレージ管理ソフトウェアです。インフラを最新化し、クラウド対応データセンターに移行することができます。ONTAP は、業界をリードするデータ管理機能を活用して、データの格納場所に関係なく、単一のツールセットでデータの管理と保護を実現します。エッジ、コア、クラウドなど、必要な場所に自由にデータを移動することもできます。ONTAP 9.8.1 には、データ管理を簡易化し、重要なデータの高速化と保護を実現し、ハイブリッドクラウドアーキテクチャ全体で次世代インフラ機能を実現する、多数の機能が搭載されています。

データ管理を簡易化

データ管理は、アプリケーションやデータセットに適切なリソースを使用できるようにするために、エンタープライズ IT 運用にとって非常に重要です。ONTAP には、運用を合理化および簡易化し、総運用コストを削減するための次の機能が含まれています。

- * インラインデータコンパクションと重複排除の強化。 * データコンパクションはストレージブロック内の無駄なスペースを削減し、重複排除は実効容量を大幅に増やします。この環境データはローカルに格納され、データはクラウドに階層化されます。
- * 最小、最大、アダプティブの Quality of Service （ AQoS ）。 * きめ細かいサービス品質（ QoS ）管理機能により、高度に共有された環境で重要なアプリケーションのパフォーマンスレベルを維持できます。
- * NetApp FabricPool 。 * この機能は、 Amazon Web Services （ AWS ）、 Azure 、 NetApp StorageGRID ストレージ解決策などのパブリックおよびプライベートクラウドストレージオプションへのコールドデータの自動階層化を提供します。FabricPool の詳細については、を参照してください "TR-4598"。

データの高速化と保護

ONTAP 9 は、卓越したパフォーマンスとデータ保護を実現し、以下の方法でこれらの機能を拡張します。

- * パフォーマンスと低レイテンシ。 * ONTAP は、可能な限り低いレイテンシで最高のスループットを提供します。
- * データ保護。 * ONTAP は、組み込みのデータ保護機能を提供し、すべてのプラットフォームで共通の管理を実現します。
- * NetApp Volume Encryption （ NVE ）。 * ONTAP は、オンボードと外部キー管理の両方をサポートし、ボリュームレベルでのネイティブな暗号化を実現します。
- * マルチテナンシーと多要素認証。 * ONTAP により、インフラリソースを最高レベルのセキュリティで共有できます。

将来のニーズにも対応できるインフラ

ONTAP 9 には次の機能が搭載されており、要件が厳しく、絶えず変化するビジネスニーズに対応できます。

- * シームレスな拡張とノンストップオペレーション。 * ONTAP は、既存のコントローラとスケールアウトクラスタに無停止で容量を追加できます。NVMe や 32Gb FC などの最新テクノロジへのアップグレードも、コストのかかるデータ移行やシステム停止を行わずに実行できます。
- * クラウドへの接続。 * ONTAP は、すべてのパブリッククラウドで Software-Defined Storage （ ONTAP Select ）とクラウドネイティブインスタンス（ NetApp Cloud Volumes Service ）を選択できる、最もクラウドに接続されたストレージ管理ソフトウェアです。
- * 新しいアプリケーションとの統合。 * ONTAP は、既存のエンタープライズアプリケーションをサポートする同じインフラストラクチャを使用して、自律走行車、スマートシティ、インダストリー 4.0 などの次世代プラットフォームやアプリケーションにエンタープライズクラスのデータサービスを提供します。

NetApp SANtricity

NetApp SANtricity は、 E シリーズハイブリッドフラッシュと EF シリーズオールフラッシュアレイに業界をリードするパフォーマンス、信頼性、シンプルさを提供するように設計されています。E シリーズハイブリッドフラッシュアレイと EF シリーズオールフラッシュアレイのパフォーマンスと利用率を最大限に高め、データ分析、ビデオ監視、バックアップとリカバリなどの高負荷のアプリケーションに対応します。SANtricity を使用すると、ストレージをオンラインにしたまま、設定の調整、メンテナンス、容量の拡張などのタスクを実

行できます。SANtricity は、優れたデータ保護、プロアクティブな監視、認定済みのセキュリティも提供します。いずれも使いやすい標準搭載の System Manager インターフェイスからアクセスできます。詳細については、を参照してください "NetApp E シリーズ SANtricity ソフトウェアのデータシート"。

パフォーマンスの最適化

パフォーマンスが最適化された SANtricity ソフトウェアは、データ分析、ビデオ監視、バックアップのすべてのアプリケーションに、高い IOPS 、高いスループット、低レイテンシを実現します。高 IOPS 、低レイテンシのアプリケーション、広帯域幅、高スループットのアプリケーションのパフォーマンスを向上

アップタイムを最大限に向上

ストレージをオンラインにしたまま、すべての管理タスクを実行できます。構成の調整、メンテナンス、容量の拡張を、 I/O を中断せずに実行できます自動化機能、オンライン構成、最先端の Dynamic Disk Pools （ DPP ）テクノロジなどにより、業界最高の信頼性を実現します。

お休みください

SANtricity ソフトウェアは、使いやすい標準搭載の System Manager インターフェイスを通じて、優れたデータ保護、プロアクティブな監視、認定済みのセキュリティを実現します。ストレージ管理業務を簡易化E シリーズストレージシステムの高度な調整に必要な柔軟性を実現します。NetApp E シリーズシステムをいつでも、どこからでも管理可能標準搭載されている Web ベースのインターフェイスにより、管理ワークフローが合理化されます。

NetApp Trident

"Trident" ネットアップは、 Docker と Kubernetes 向けのオープンソースの動的ストレージオーケストレーションツールであり、永続的ストレージの作成、管理、使用を簡易化します。Kubernetes ネイティブアプリケーションである Trident は、 Kubernetes クラスタ内で直接実行されます。Trident を使用すると、 DL コンテナイメージをネットアップストレージにシームレスに導入し、エンタープライズクラスの AI コンテナ環境を実現できます。Kubernetes ユーザ（ ML 開発者やデータサイエンティストなど）は、オーケストレーションとクローニングを作成、管理、自動化し、ネットアップテクノロジを基盤とするネットアップの高度なデータ管理機能を活用できます。

NetApp BlueXPのコピーと同期

"BlueXPのコピーと同期" 迅速かつセキュアなデータ同期を実現するネットアップのサービスです。オンプレミスのNFSまたはSMBファイル共有間でファイルを転送する必要があるかどうかにかかわらず、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、 IBM Cloud Object StorageのBlueXP Copy and Syncなら、必要な場所に迅速かつセキュアにファイルを移動できます。転送されたデータは、ソースとターゲットの両方で完全に使用できます。BlueXPのCopy and Syncは、事前定義されたスケジュールに基づいて継続的にデータを同期し、差分のみを移動するため、データレプリケーションにかかる時間とコストを最小限に抑えることができます。BlueXPのCopy and Syncは、セットアップと使用が非常に簡単なソフトウェアサービス（SaaS）ツールです。BlueXPのCopyとSyncによってトリガーされるデータ転送は、データブローカーによって実行されます。BlueXPのCopy and Syncデータブローカーは、AWS、Azure、Google Cloud Platform、オンプレミスに導入できます。

Lenovo ThinkSystem サーバ

Lenovo ThinkSystem サーバは、革新的なハードウェア、ソフトウェア、サービスを搭載しており、お客様の現在の課題を解決し、将来の課題に対処するための、進化した、用途に合わせたモジュラー設計アプローチを提供します。これらのサーバは、クラス最高の業界標準テクノロジーと、差別化された Lenovo の革新技術を

組み合わせて、 x86 サーバで可能な限り高い柔軟性を提供します。

Lenovo ThinkSystem サーバを導入する主なメリットは次のとおりです。

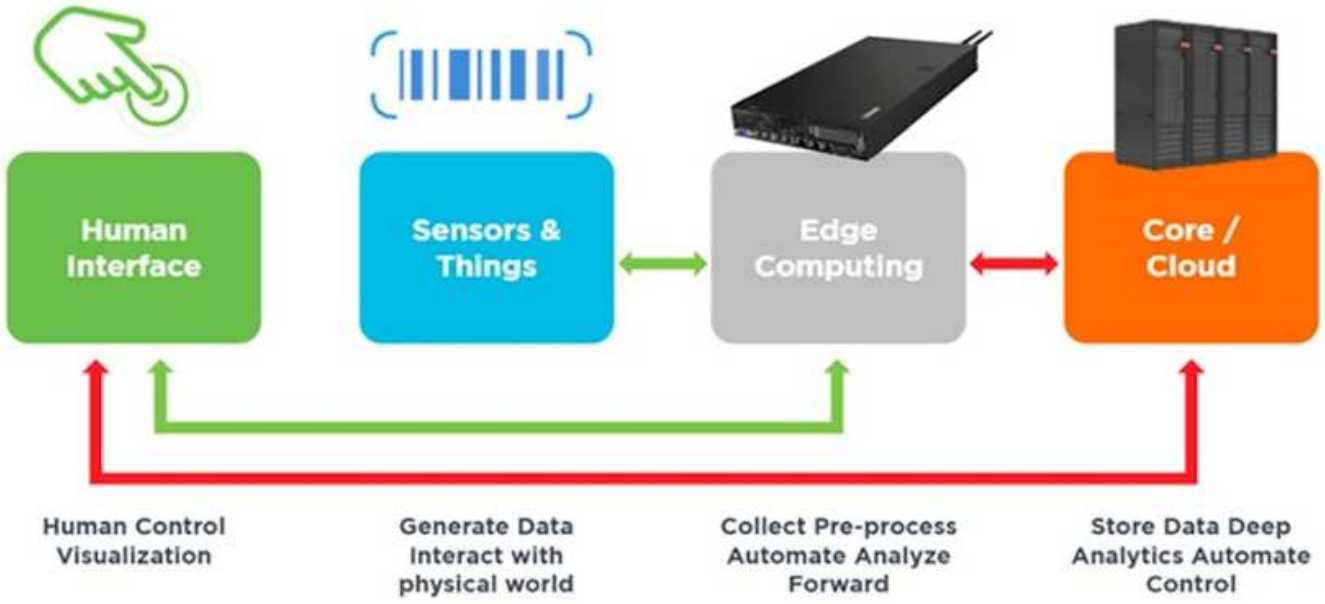
- ビジネスの成長に合わせて拡張性に優れたモジュラ設計
- 業界をリードする耐障害性により、計画外停止にかかるコストを時間単位で削減します
- 高速フラッシュテクノロジにより、レイテンシを低減し、応答時間を短縮し、リアルタイムでのデータ管理をスマートに実現します

Lenovo は、 AI 分野において、企業がワークロードに ML と AI のメリットを理解し、採用できるようにするための実践的なアプローチをとっています。Lenovo のお客様は、 Lenovo AI Innovation Center で Lenovo AI 製品を調査および評価し、特定のユースケースの価値を十分に理解することができます。価値実現までの時間を短縮するために、このお客様中心のアプローチでは、 AI に最適化された、すぐに使用できる解決策開発プラットフォームのコンセプトの実証をお客様に提供しています。

Lenovo ThinkSystem SE350 Edge Server

エッジコンピューティングにより、 IoT デバイスからのデータをネットワークのエッジで分析してから、データセンターやクラウドに送信できます。下の図に示す Lenovo ThinkSystem SE350 は、柔軟性、接続性、セキュリティ、およびリモート管理性を重視した、耐久性と環境を強化したコンパクトなフォームファクタのエッジでの導入に固有の要件を満たすように設計されています。

SE350 は、エッジ AI ワークロードの高速化をサポートする柔軟性を備えたインテル Xeon D プロセッサーを搭載しており、データセンター外のさまざまな環境でのサーバー導入の課題に対応できるように設計されています。



MLPerf

MLPerf は、 AI のパフォーマンスを評価するための業界をリードするベンチマークスイートです。画像分類、オブジェクト検出、医療画像処理、自然言語処理（ NLP ）など、応用 AI の多くの分野をカバーしています。この検証では、推論 v0.7 ワークロードを使用しました。これは、この検証の完了時に MLPerf 推論の最新の反復処理です。。 "MLPerf 推論 v0.7" Suite には、データセンターとエッジシステムのための 4 つの新しいベンチマークが含まれています。

- * BERT * Transformers （ BERT ）の双方向エンコーダリプレゼンテーションは、チームデータセットを使用して質問に答えるように微調整されています。
- * DLRM.* ディープラーニング・レコメンド・モデル（ DLRM ）は、クリックスルー・レート（ CTR ）を最適化するためのトレーニングを受けた、パーソナライズされた推奨モデルです。
- *3D U-Net. * 3D U-Net アーキテクチャは、 Brain Tumor Segmentation （ BRT ）データセットについてトレーニングされています。
- **RNN-T** 再帰型ニューラルネットワークトランスデューサ (RNN-T) は、 LibriSpeech のサブセットについてトレーニングを受けた自動音声認識 (ASR) モデルです。MLPerf 推論の結果とコードは、 Apache ライセンスに基づいて公開およびリリースされます。MLPerf Inference にはエッジがあり、次のシナリオをサポートします。
- * 単一ストリーム * このシナリオは、スマートフォンで実行されるオフライン AI クエリなど、応答性が重要な要因となるシステムを模倣しています。個々のクエリがシステムに送信され、応答時間が記録されます。すべての応答の 90 パーセンタイルレイテンシが結果として報告されます。
- * マルチストリーム * このベンチマークは、複数のセンサーからの入力を処理するシステム用です。テスト中は、一定の間隔でクエリが送信されます。QoS の制約（許容される最大レイテンシ）が発生する。テストでは、 QoS の制約を満たしている間にシステムが処理できるストリーム数が報告されます。
- * オフライン。 * これはバッチ処理アプリケーションを対象とした最も簡単なシナリオで、メトリックは 1 秒あたりのサンプル数でスループットです。すべてのデータをシステムで使用でき、ベンチマークはすべてのサンプルの処理にかかる時間を測定します。

Lenovo は、本ドキュメントで使用されているサーバである T4 で SE350 の MLPerf Inference スコアを発表しました。の結果を参照してください "https://mlperf.org/inference-results-0-7/" エントリ #0.7~145 の「 Edge

、 Closed Division 」セクションに記載されています。

テスト計画

このドキュメントは MLPerf 推論 v0.7 に準拠しています "コード"、 MLPerf Inference v1.1 "コード"および "ルール"。次の表に示すように、エッジでの推論向けに設計された MLPerf ベンチマークを実行しました。

面積（ **Area** ）	タスク	モデル	データセット	**QSL** サイズ	品質	マルチストリーム遅延制約
ビジョン	画像分類	Resnet50v1.5	ImageNet (224x224)	1024	FP32 の 99%	50 ミリ秒
ビジョン	物体検出（大）	SSD リネット 34	ココ (1200x1200)	64	FP32 の 99%	66 ミリ秒
ビジョン	物体検出（小）	ssd - MobileNetsv1 を参照してください	ココ (300 x 300)	256	FP32 の 99%	50 ミリ秒
ビジョン	医療画像のセグメンテーション	3D UNET	2019 年 BRT （ 224x224x160 ）	16	FP32 の 99% および 99.9%	該当なし
スピーチ	音声テキスト	RNNT	ライブラリキーテック開発 - クリーン	2513	FP32 の 99%	該当なし
言語	言語処理	BERT	分隊 v1.1	10833	FP32 の 99%	該当なし

次の表に、 Edge ベンチマークのシナリオを示します。

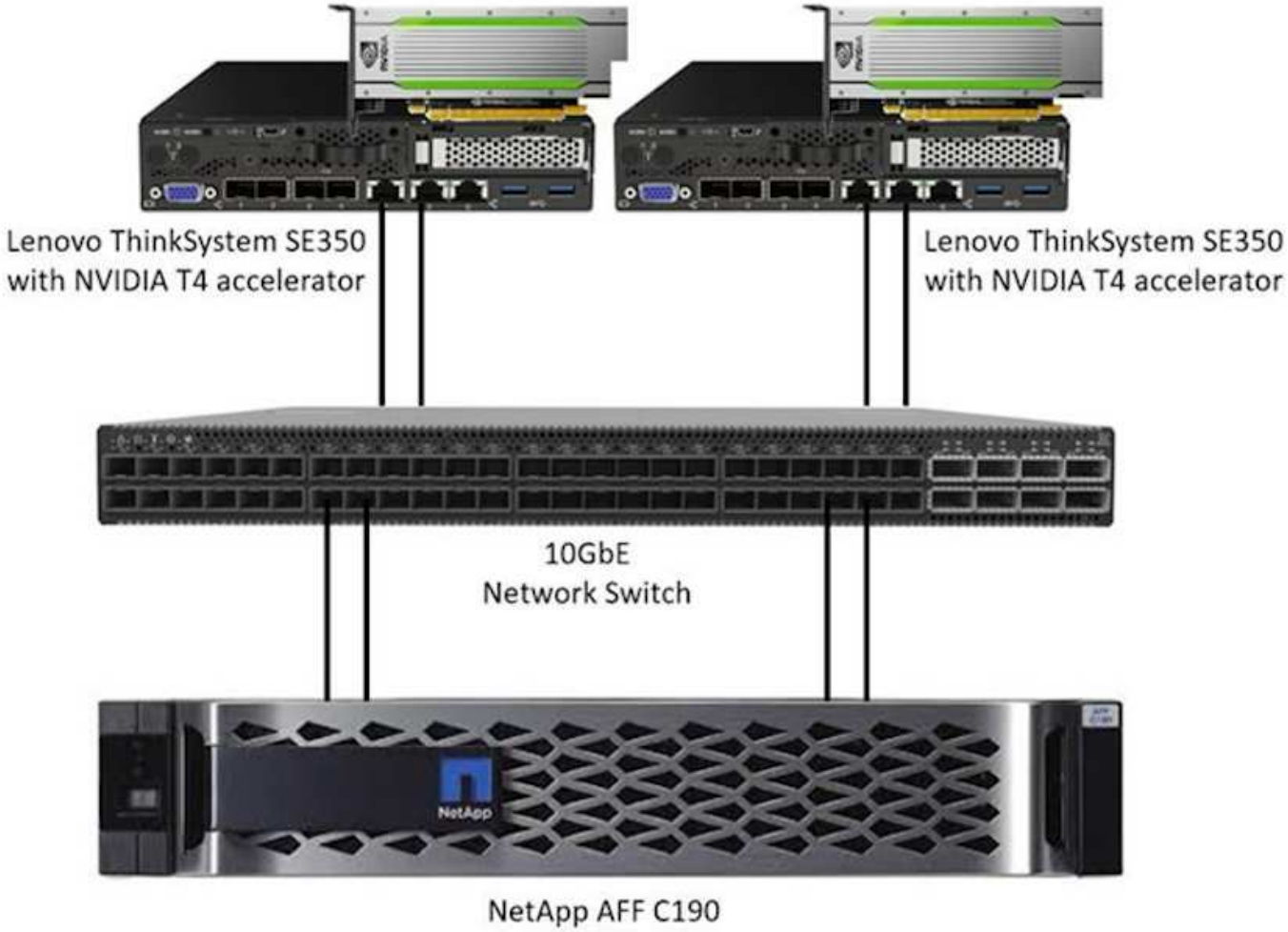
面積（ **Area** ）	タスク	シナリオ
ビジョン	画像分類	シングルストリーム、オフライン、マルチストリーム
ビジョン	物体検出（大）	シングルストリーム、オフライン、マルチストリーム
ビジョン	物体検出（小）	シングルストリーム、オフライン、マルチストリーム
ビジョン	医療画像のセグメンテーション	単一ストリーム、オフライン
スピーチ	音声テキスト	単一ストリーム、オフライン
言語	言語処理	単一ストリーム、オフライン

この検証で開発されたネットワーク・ストレージ・アーキテクチャを使用してこれらのベンチマークを実行し、 MLPerf に送信されたエッジ・サーバ上のローカル実行の結果と比較しました。この比較は、共有ストレージが推論パフォーマンスに与える影響を判断するためのものです。

次の図に、テスト構成を示します。NetApp AFF C190 ストレージシステムと、 Lenovo ThinkSystem SE350 サーバを 2 台（それぞれ NVIDIA T4 アクセラレータを 1 台搭載）使用しました。これらのコンポーネントは、 10GbE ネットワークスイッチを介して接続されます。ネットワークストレージには、検証 / テスト用のデータセットと事前トレーニング済みのモデルが格納されます。サーバはコンピューティング機能を提供し、ストレージに NFS プロトコル経由でアクセスします。

このセクションでは、テスト構成、ネットワークインフラストラクチャ、 SE350 サーバ、およびストレージプロビジョニングの詳細について説明します。次の表に、解決策アーキテクチャの基本コンポーネントを示します。

解決策コンポーネント	詳細
Lenovo ThinkSystem サーバ	• SE350 サーバ x 2 （それぞれ NVIDIA T4 GPU カード 1 枚）
	• 各サーバには Intel Xeon D-2123IT CPU が 1 つ搭載され、物理コアは 2.20GHz と 128GB の RAM で動作します
エントリレベルの NetApp AFF ストレージシステム（HA ペア）	• NetApp ONTAP 9 ソフトウェア • 960GB SSD × 24 • NFS プロトコル • コントローラごとに 1 つのインターフェイスグループ。マウントポイント用に 4 つの論理 IP アドレスが割り当てられます



次の表に、 AFF C190 と 2RU 、 24 ドライブスロットのストレージ構成を示します。

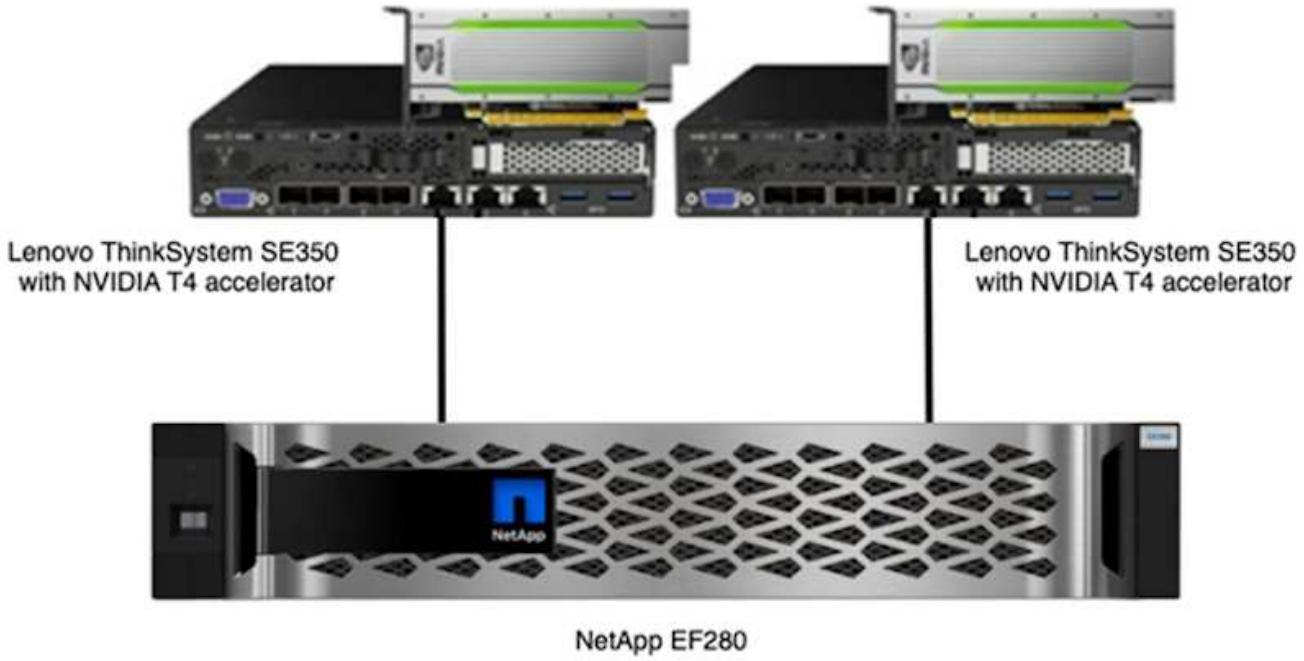
コントローラ	アグリゲート	**FlexGroup** ボリューム	**Aggregatesize** を実行します	ボリュームサイズ	オペレーティングシステムのマウントポイント
コントローラ 1	aggr1	/netappleenov_AI_fg	8.42TiB	15TB	/NetApp_Lenovo_ fg
コントローラ 2	aggr2		8.42TiB		

/netappLenovo_AI_fg フォルダには、モデルの検証に使用するデータセットが含まれています。

次の図は、テスト構成を示しています。NetApp EF280 ストレージシステムを 2 台、 Lenovo ThinkSystem SE350 サーバを 2 台（それぞれ NVIDIA T4 アクセラレータを 1 台搭載）使用しました。これらのコンポーネントは、 10GbE ネットワークスイッチを介して接続されます。ネットワークストレージには、検証 / テスト用のデータセットと事前トレーニング済みのモデルが格納されます。サーバはコンピューティング機能を提供し、ストレージに NFS プロトコル経由でアクセスします。

次の表に、 EF280 のストレージ構成を示します。

コントローラ	ボリュームグループ	ボリューム	ボリュームサイズ	**DDPsize**	接続方法
コントローラ 1	DDP1	ボリューム 1	8.42TiB	16TB	SE350-1 から iSCSI LUN 0
コントローラ 2		ボリューム 2	8.42TiB		SE350-2 から iSCSI LUN 1 へ



手順をテストします

このセクションでは、この解決策の検証に使用するテスト手順について説明します。

オペレーティングシステムと **AI** 推論のセットアップ

AFF C190 には、 NVIDIA ドライバと Docker を搭載した Ubuntu 18.04 を使用し、 NVIDIA GPU をサポートし、 MLPerf を使用しました "コード" Lenovo から MLPerf Inference v0.7 への提出書類の一部として提供されます。

EF280 には、 NVIDIA ドライバと Docker を搭載した Ubuntu 20.04 を使用し、 NVIDIA GPU と MLPerf をサポートしました "コード" Lenovo から MLPerf Inference v1.1 への提出の一部として提供されています。

AI 推論をセットアップするには、次の手順を実行します。

1. 登録が必要なデータセット、 ImageNet 2012 Validation set 、 Crito Terabyte データセット、および BRT 2019 Training セットをダウンロードし、ファイルを解凍します。
2. 1TB 以上の作業ディレクトリを作成し、ディレクトリを参照する環境変数「 M LPERF_scratch_path 」を定義します。

   このディレクトリは、ネットワークストレージのユースケース用に共有ストレージ上で共有するか、また

はローカルデータでテストする際にローカルディスク上で共有する必要があります。

3. make 「 prebuild 」コマンドを実行します。このコマンドは、必要な推論タスク用の Docker コンテナを構築して起動します。

> 実行中の Docker コンテナ内から次のコマンドがすべて実行されます。

- MLPerf Inference タスク用のトレーニング済み AI モデル「 make download_model 」をダウンロードしてください
- 無料でダウンロードできる追加のデータセット「 make download_data 」をダウンロードしてください
- データをプリプロセスします。「 preprocess_data 」にします
- 「 make build 」を実行します。
- コンピューティングサーバの GPU に最適化された推論エンジン「 generate_engines 」を構築します
- 推論ワークロードを実行するには、次のコマンドを実行します（ 1 つのコマンド）。

```
make run_harness RUN_ARGS="--benchmarks=<BENCHMARKS>
--scenarios=<SCENARIOS>"
```

AI 推論の実行

実行された実行のタイプは次の 3 つです。

- ローカルストレージを使用した単一サーバの AI 推論
- ネットワークストレージを使用した単一サーバの AI 推論
- ネットワークストレージを使用したマルチサーバ AI 推論

テスト結果

提案するアーキテクチャのパフォーマンスを評価するために、多数のテストを実施しました。

6 種類のワークロードがあります（画像分類、オブジェクト検出 [小規模] 、オブジェクト検出 [大規模] 、医療画像処理、音声テキスト変換、 また、ナチュラル言語処理（ NLP ）もサポートされており、オフライン、シングルストリーム、マルチストリームの 3 つのシナリオで実行できます。

> 最後のシナリオは、画像分類とオブジェクト検出の場合にのみ実装されます。

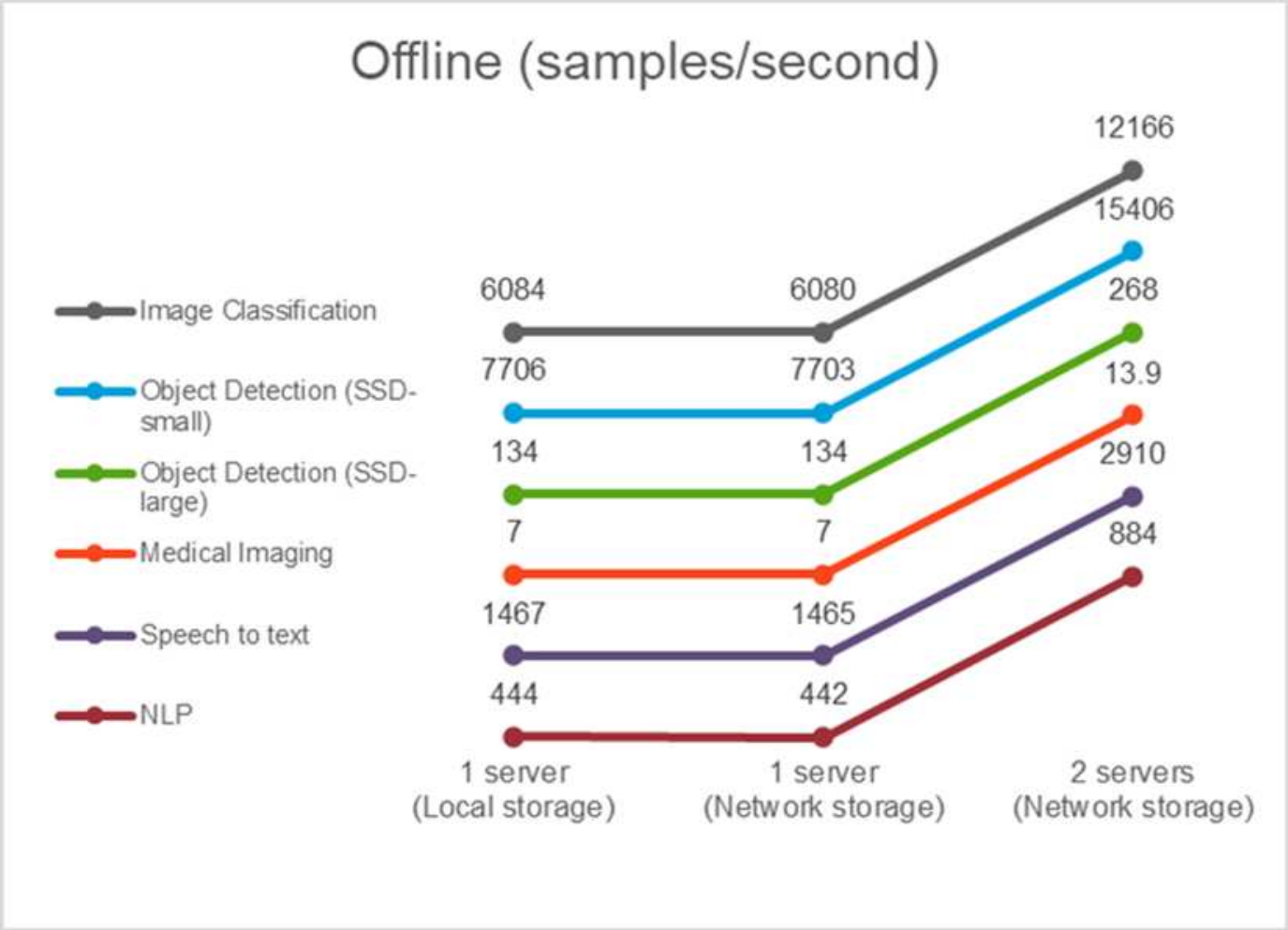
その結果、次の 3 種類のセットアップですべてテストされた 15 のワークロードが生成されます。

- 単一のサーバ / ローカルストレージ
- 単一のサーバ / ネットワークストレージ
- マルチサーバ / ネットワークストレージ

結果については、以降のセクションで説明します。

AFF のオフラインシナリオにおける **AI** 推論

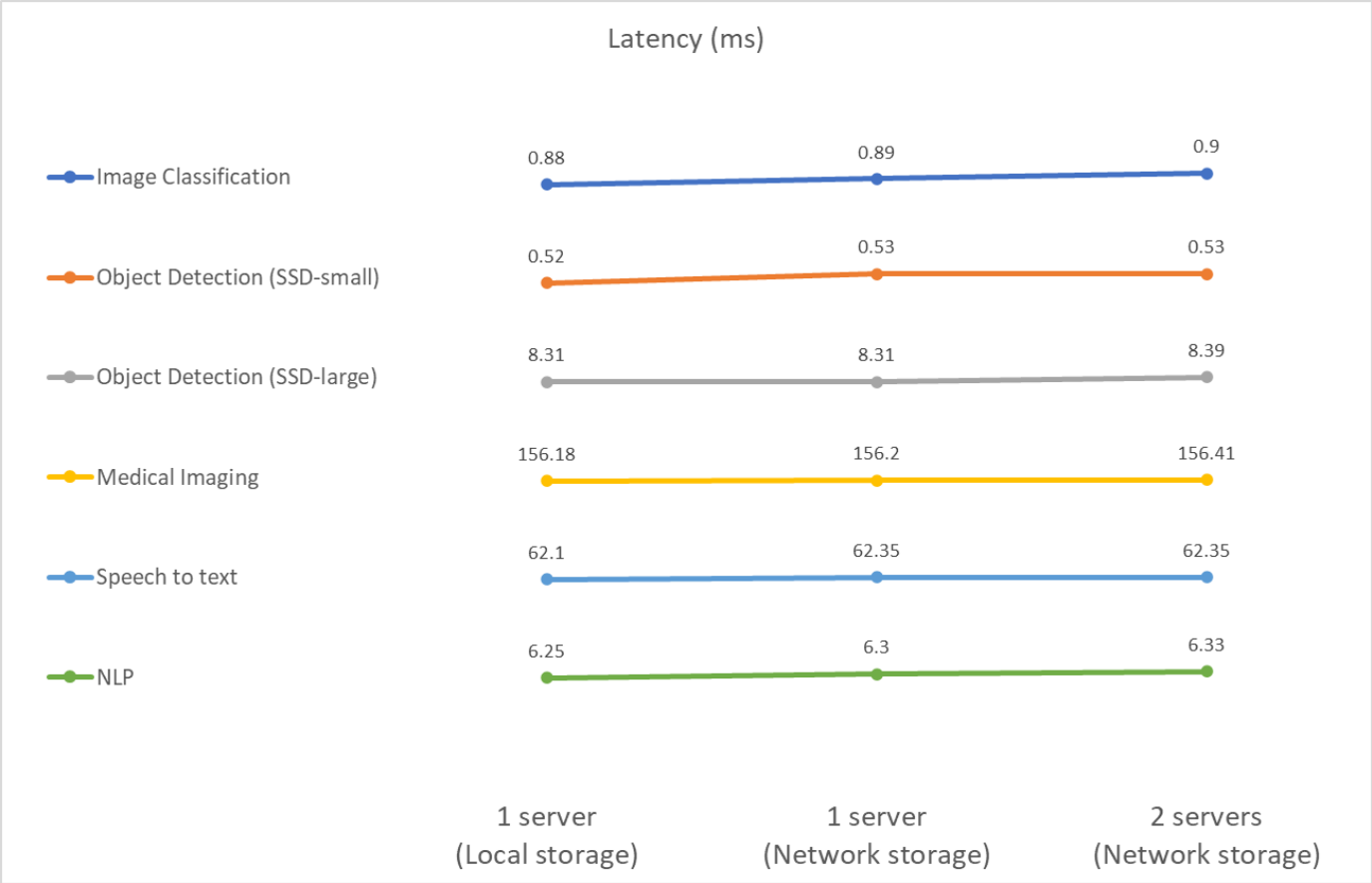
このシナリオでは、すべてのデータがサーバで使用可能であり、すべてのサンプルの処理にかかった時間が測定されました。テストの結果として、帯域幅が 1 秒あたりのサンプル数で報告されます。複数のコンピューティングサーバを使用した場合、すべてのサーバの合計帯域幅がレポートされます。3 つのユースケースすべての結果を次の図に示します。2 サーバの場合は、両方のサーバからの帯域幅の合計を報告します。



結果から、ネットワークストレージがパフォーマンスに悪影響を与えていないことがわかります。変更は最小限で、一部のタスクでは何も検出されません。2 台目のサーバを追加する場合、合計帯域幅は正確に 2 倍になるか、最悪の場合は 1 % 未満になります。

AFF 向けの単一ストリームのシナリオでの **AI** 推論

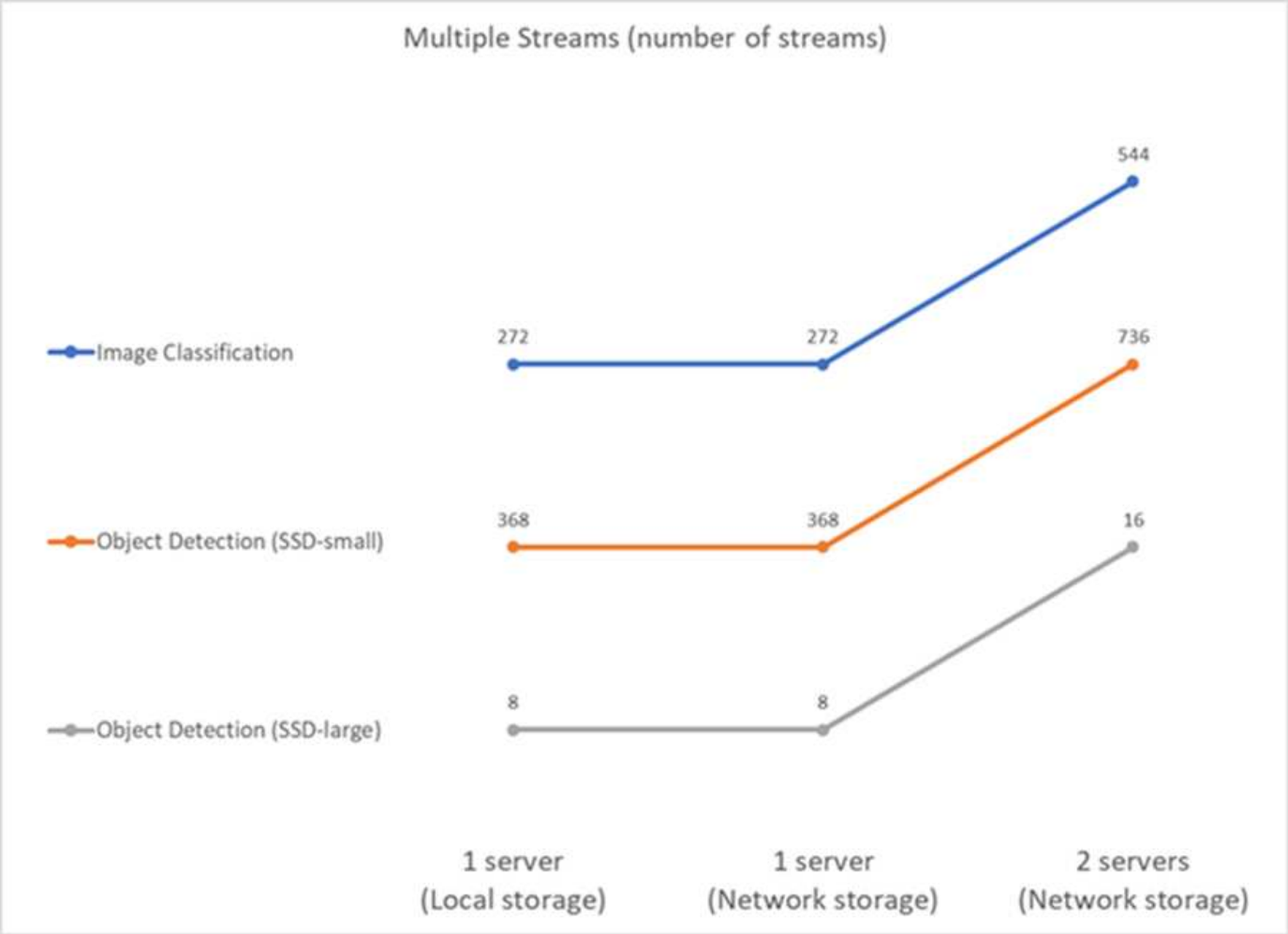
このベンチマークではレイテンシを測定します。複数のコンピューティングサーバの場合は、平均レイテンシが報告されます。一連のタスクの結果を次の図に示します。2 台のサーバの場合は、両方のサーバの平均レイテンシが報告されます。



結果からも、タスクを処理するのに十分なネットワークストレージがあることがわかります。1 つのサーバケースにおけるローカルストレージとネットワークストレージの違いは、最小またはなしです。同様に、 2 台のサーバが同じストレージを使用している場合、両方のサーバの遅延は同じままであるか、非常に小さい値で変化します。

AFF のマルチストリームシナリオにおける **AI** 推論

この場合、 QoS の制約を満たしながらシステムで処理可能なストリーム数が返されます。したがって、結果は常に整数になります。複数のサーバについて ' すべてのサーバの合計ストリーム数を報告しますすべてのワークロードがこのシナリオをサポートしているわけではありませんが、そのシナリオを実行してきました。テストの結果を次の図にまとめます。2 サーバの場合は、両方のサーバからのストリームの合計数を報告します。



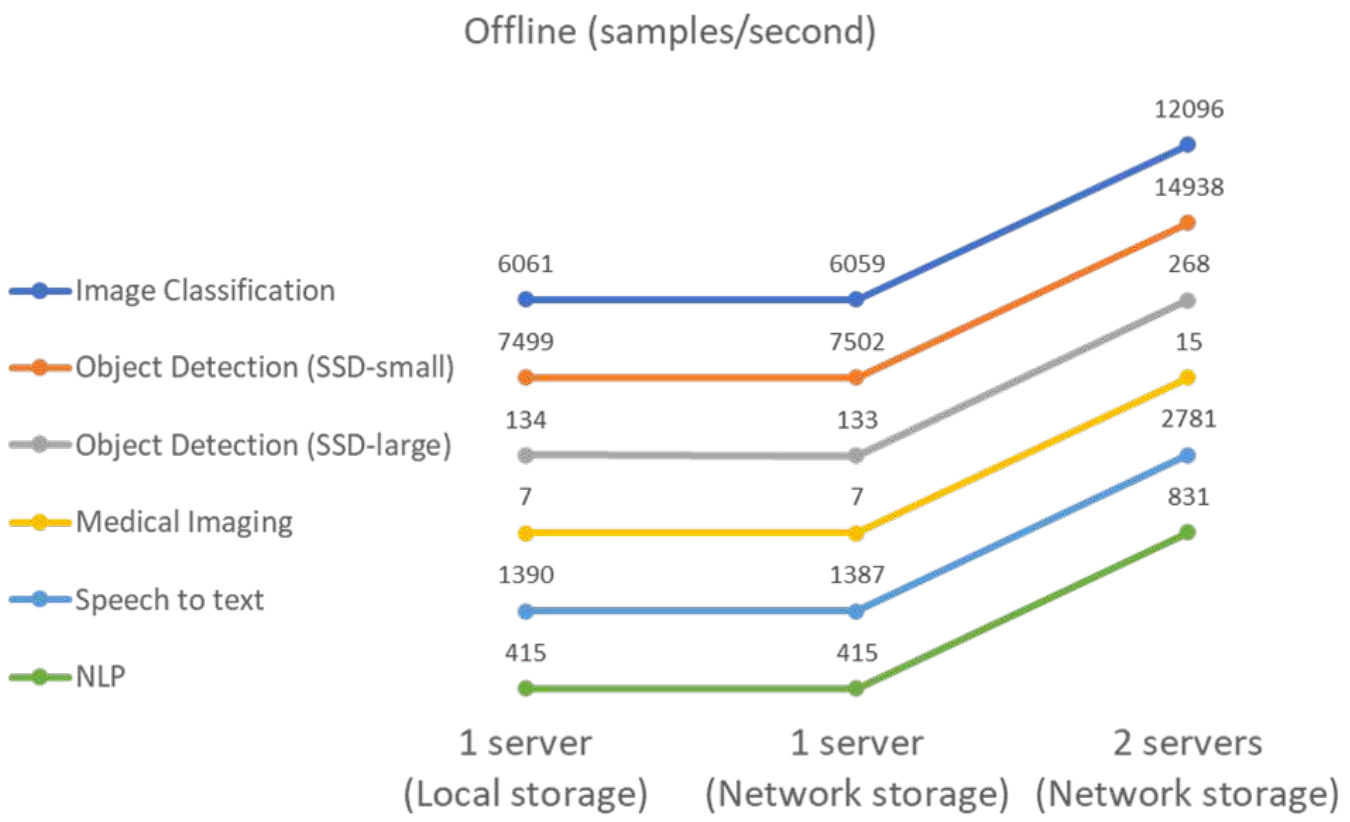
この結果は、ローカルストレージとネットワーキングストレージでセットアップのパフォーマンスが完璧に向上し、 2 台目のサーバを追加すると、提案されたセットアップで処理できるストリーム数が 2 倍になります。

EF シリーズのテスト結果

提案するアーキテクチャのパフォーマンスを評価するために、多数のテストを実施しました。6 種類のワークロードがあります（画像分類、オブジェクト検出 [小規模] 、オブジェクト検出 [大規模] 、医療画像処理、音声テキスト変換、 とナチュラル言語処理（ NLP ）は、オフラインとシングルストリームの 2 つの異なるシナリオで実行されました。結果については、以降のセクションで説明します。

EF 向けのオフラインシナリオでの AI 推論

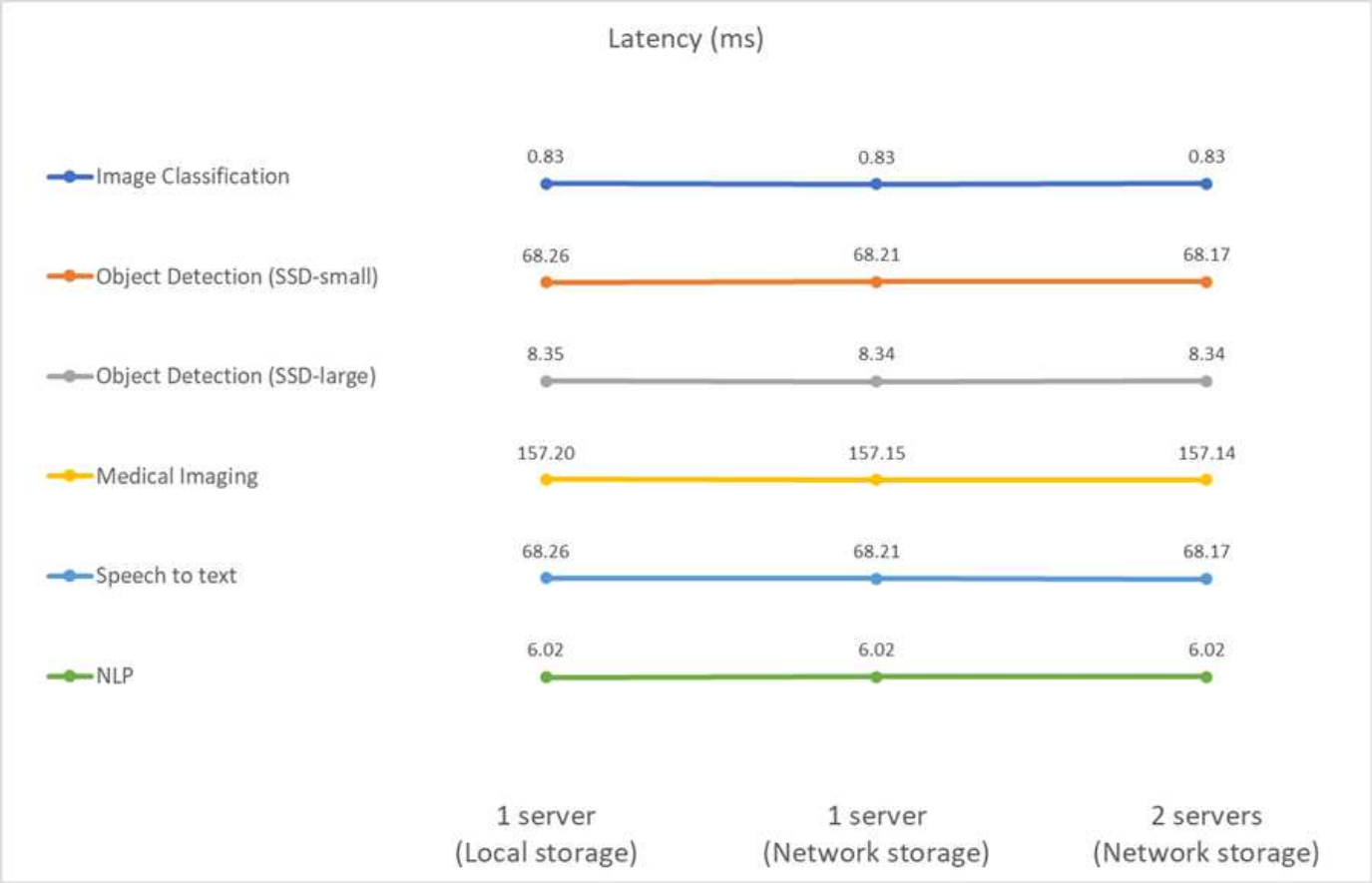
このシナリオでは、すべてのデータがサーバで使用可能であり、すべてのサンプルの処理にかかった時間が測定されました。テストの結果として、帯域幅が 1 秒あたりのサンプル数で報告されます。1 つのノードの実行については両方のサーバからの平均をレポートし、 2 つのサーバの実行については、すべてのサーバに合計された合計帯域幅をレポートします。ユースケースの結果を次の図に示します。



結果から、ネットワークストレージがパフォーマンスに悪影響を与えていないことがわかります。変更は最小限で、一部のタスクでは何も検出されません。2 台目のサーバを追加する場合、合計帯域幅は正確に 2 倍になるか、最悪の場合は 1 % 未満になります。

EF 向けの単一ストリームのシナリオでの **AI** 推論

このベンチマークではレイテンシを測定します。いずれの場合も、実行に関連するすべてのサーバの平均レイテンシを報告します。一連のタスクの結果が表示されます。



結果から、このタスクを処理するのに十分なネットワークストレージがあることが再びわかります。1 つのサーバケースにおけるローカルストレージとネットワークストレージの違いは、最小またはなしです。同様に、2 台のサーバが同じストレージを使用している場合、両方のサーバの遅延は同じままであるか、非常に小さい値で変化します。

アーキテクチャのサイジングオプション

検証に使用する設定は、他のユースケースに合わせて調整できます。

コンピューティングサーバ

SE350 でサポートされている最小レベルの CPU である Intel Xeon D-2123IT CPU を使用し、 4 つの物理コアと 60W TDP を使用しました。サーバは CPU の交換をサポートしていませんが、より強力な CPU で発注することもできます。サポートされている CPU の上位は、 16 コアを搭載した Intel Xeon D-2183IT 、 2.20GHz で動作する 100W です。これにより、 CPU の計算能力が大幅に向上します。CPU は推論ワークロード自体を実行するためのボトルネックではありませんでしたが、データ処理や推論に関連するその他のタスクに役立ちます。現時点では、 NVIDIA T4 がエッジで唯一の GPU です。そのため、 GPU のアップグレードやダウングレードは行えません。

共有ストレージ

テストと検証には、ストレージ容量が最大 50.5TB の NetApp AFF C190 システムが使用されています。シーケンシャルリードの場合は 4.4GBps のスループット、スモールランダムリードの場合は 230K の IOPS が、このドキュメントではエッジ推論ワークロードに適していることが実証されています。

ただし、より多くのストレージ容量を必要としたり、より高速なネットワーク速度を必要とする場合は、NetApp AFF A220 またはを使用してください "NetApp AFF A250" ストレージシステムまた、最大容量が

1.5PB の NetApp EF280 システムでは、この解決策検証に、帯域幅も 10Gbps 使用しました。より多くのストレージ容量をより多くの帯域幅で使用する場合は、 "NetApp EF300" を使用できます。

まとめ

AI 主導の自動化とエッジコンピューティングは、ビジネス組織がデジタル変革を実現し、運用効率と安全性を最大限に高めるための、業界をリードするアプローチです。エッジコンピューティングでは、データセンターとの間を移動する必要がないため、データの処理速度が大幅に向上します。そのため、データセンターやクラウドへのデータの送受信に関連するコストが削減されます。エッジに導入された AI 推論モデルを使用してほぼリアルタイムで意思決定を行う必要がある場合は、レイテンシの低減とスピードの向上が効果的です。

ネットアップのストレージシステムは、ローカル SSD ストレージと同等以上のパフォーマンスを発揮し、データサイエンティスト、データエンジニア、 AI / ML 開発者、ビジネスや IT の意思決定者に次のようなメリットをもたらします。

- AI システム、分析などの重要なビジネスシステム間でデータを容易に共有できます。このようなデータ共有により、インフラのオーバーヘッドを削減し、パフォーマンスを向上させ、企業全体のデータ管理を合理化できます。
- 個別に拡張可能なコンピューティングとストレージにより、コストを最小限に抑え、リソース使用率を向上させます。
- 統合された Snapshot コピーとクローンを使用して開発と導入のワークフローを合理化し、ユーザのワークスペースを瞬時にスペース効率よく利用できるほか、バージョン管理機能も統合され、導入も自動化されています。
- ディザスタリカバリとビジネス継続性を実現するエンタープライズクラスのデータ保護本ドキュメントで紹介するネットアップと Lenovo の解決策は、柔軟性に優れたスケールアウトアーキテクチャを備えており、エッジでのエンタープライズクラスの AI 推論導入に最適です。

謝辞

- J. J. J.Falkanger 、 Sr.Lenovo 、 HPC & AI ソリューション担当マネージャー
- ネットアップ、テクニカルマーケティングエンジニア、 Dave Arnette 氏
- Joey Parnell 、 Tech Lead E シリーズ AI Solutions 、ネットアップ
- ネットアップ、 QA エンジニア、 Cody Harryman 氏

追加情報の参照先

このドキュメントに記載されている情報の詳細については、以下のドキュメントや Web サイトを参照してください。

- NetApp AFF A シリーズアレイの製品ページ

  "https://www.netapp.com/data-storage/aff-a-series/"

- NetApp ONTAP データ管理ソフトウェア— ONTAP 9 情報ライブラリ

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

- TR-4727 ：『 NetApp EF Series Introduction 』

  https://www.netapp.com/pdf.html?item=/media/17179-tr4727pdf.pdf

- NetApp E シリーズ SANtricity ソフトウェアのデータシート

  https://www.netapp.com/pdf.html?item=/media/19775-ds-3171-66862.pdf

- コンテナ向け NetApp 永続的ストレージ— NetApp Trident

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

- MLPerf
  - "https://mlcommons.org/en/"
  - "http://www.image-net.org/"
  - "https://mlcommons.org/en/news/mlperf-inference-v11/"
- NetApp BlueXPのコピーと同期

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

- TensorFlow ベンチマーク

  "https://github.com/tensorflow/benchmarks"

- Lenovo ThinkSystem SE350 Edge Server

  "https://lenovopress.com/lp1168"

- Lenovo ThinkSystem DM5100F ユニファイドフラッシュストレージアレイ

  "https://lenovopress.com/lp1365-thinksystem-dm5100f-unified-flash-storage-array"

WP-7328 ：『 NetApp Conversational AI Using NVIDIA Jarvis 』

ネットアップ、 Rick Huang 、 Sung-Han Lin 、 NVIDIA 、 Davide Onofrio

NVIDIA DGX システムファミリーは、エンタープライズ AI に特化した世界初の人工知能（ AI ）ベースシステムで構成されます。NetApp AFF ストレージシステムは、卓越したパフォーマンスと業界をリードするハイブリッドクラウドデータ管理機能を提供します。ネットアップと NVIDIA は提携を通じて、 NetApp ONTAP AI リファレンスアーキテクチャを構築しました。このリファレンスアーキテクチャは、 AI と機械学習（ ML ）のワークロード向けのターンキー解決策であり、エンタープライズクラスのパフォーマンス、信頼性、サポートを提供します。

このホワイトペーパーでは、さまざまな業種のさまざまなユースケースに対応した会話型 AI システムを構築するお客様にガイダンスを提供します。これには、 NVIDIA Jarvis を使用したシステムの導入に関する情報が含まれています。テストは NVIDIA DGX ステーションと NetApp AFF A220 ストレージシステムを使用して実施しました。

解決策の対象となるグループは次のとおりです。

- AI 開発のためのソリューションを設計するエンタープライズアーキテクト などの会話型 AI のユースケースに適したモデルとソフトウェア バーチャル・リテール・アシスタント
- データサイエンティストは、言語モデリングを効率的に実現する方法を探しています 能力開発の目標
- テキストデータの保守と処理を担当するデータエンジニア お客様からの質問や会話の記録など
- エグゼクティブや IT の意思決定者、および関心のあるビジネスリーダー 会話型 AI のエクスペリエンスを変革し、最速の時間を実現できます AI への取り組みから市場に投入

解決策の概要

NetApp ONTAP AIとBlueXPのコピーと同期

ネットアップと NVIDIA は、 NVIDIA DGX システムとネットアップのクラウド対応ストレージシステムを基盤とする NetApp ONTAP AI アーキテクチャを開発、検証しました。このリファレンスアーキテクチャには、IT 組織に次のようなメリットがあります。

- 設計の複雑さを解消
- コンピューティングとストレージを個別に拡張できます
- 小規模構成から始めて、シームレスに拡張できます
- さまざまなパフォーマンスとコストの観点から、幅広いストレージオプションを提供 NetApp ONTAP AI は、 DGX システムと NetApp AFF A220 ストレージシステムを最先端のネットワーク機能と緊密に統合します。NetApp ONTAP AI システムと DGX システムでは、設計の複雑さと推測に頼らず、 AI 導入を簡易化できます。お客様は小規模構成から始めて、システムを中断なく拡張できます。同時に、エッジ、コア、クラウドにわたってデータをインテリジェントに管理できます。

NetApp BlueXPのコピーと同期機能を使用すると、2つのNFS共有、2つのCIFS共有、1つのファイル共有とAmazon S3、Amazon Elastic File System（EFS）、Azure Blob Storageの間など、さまざまなプロトコルを使用してデータを簡単に移動できます。アクティブ / アクティブ処理とは、ソースとターゲットの両方と同時に作業を継続し、必要に応じてデータの変更を段階的に同期することを意味します。BlueXPのCopy and Syncは、オンプレミスでもクラウドベースでも、あらゆるソースシステムとデスティネーションシステムの間でデータを移動して差分同期できるため、データの利用方法が多様化します。オンプレミスのシステム間でのデータ移行、クラウドへのオンボーディングやクラウドへの移行、コラボレーションとデータ分析などのすべての作業を容易に実現できます。次の図は、使用可能なソースとデスティネーションを示しています。

会話型AIシステムでは、開発者はBlueXPのコピーと同期を活用して会話履歴をクラウドからデータセンターにアーカイブし、自然言語処理（NLP）モデルのオフライントレーニングを実現できます。より多くのインテントを認識するためのトレーニングモデルによって、会話型 AI システムは、エンドユーザーからのより複雑な質問にも対応できるようになります。

NVIDIA Jarvis マルチモーダルフレームワーク

"NVIDIA Jarvis" 会話型 AI サービスを構築するためのエンドツーエンドのフレームワークです。GPU 向けに最適化された次のサービスが含まれています。

- 自動音声認識（ASR）
- 自然言語理解（NLU）
- ドメイン固有のフルフィルメントサービスとの統合
- テキスト / スピーチ（TTS）
- コンピュータビジョン（CV）ジャービスベースのサービスは、最先端のディープラーニングモデルを使用して、リアルタイムの会話型 AI の複雑で困難なタスクに対処します。エンドユーザーとのリアルタイムかつ自然な対話を可能にするには、モデルが 300 ミリ秒未満で計算を完了する必要があります。自然な相互作用は困難であり、マルチモーダル感覚を統合する必要があります。モデルパイプラインも複雑で、上記のサービス全体で調整が必要です。

Jarvis は、エンドツーエンドのディープラーニングパイプラインを使用する、マルチモーダル会話型 AI サービスを構築するための、完全に高速化されたアプリケーションフレームワークです。Jarvis フレームワークには、音声、ビジョン、および NLU タスク向けに、事前にトレーニングされた会話型 AI モデル、ツール、最適化されたエンドツーエンドサービスが含まれます。AI サービスに加えて、 Jarvis ではビジョン、オーディオ、およびその他のセンサー入力を同時に融合し、仮想アシスタント、マルチユーザーのディアゼーション、コールセンターアシスタントなどのアプリケーションでマルチコンテキスト会話などの機能を提供できます。

NVIDIA Nemo

"NVIDIA Nemo" は、使いやすいアプリケーションプログラミングインターフェイス（API）を使用して、GPU によって高速化された最先端の会話型 AI モデルを構築、トレーニング、微調整するためのオープンソース Python ツールキットです。Nemo は、 NVIDIA GPU で Tensor コアを使用して精度の高いコンピューティングを実行し、複数の GPU に簡単にスケールアップして、トレーニングのパフォーマンスを最大限に高めることができます。Nemo は、医療、金融、小売、通信など、さまざまな業界のさまざまな業界で、ビデオ通話の文字変換、インテリジェントビデオアシスタント、自動コールセンターサポートなどのリアルタイム ASR 、 NLP 、 TTS アプリケーションのモデルを構築するために使用されます。

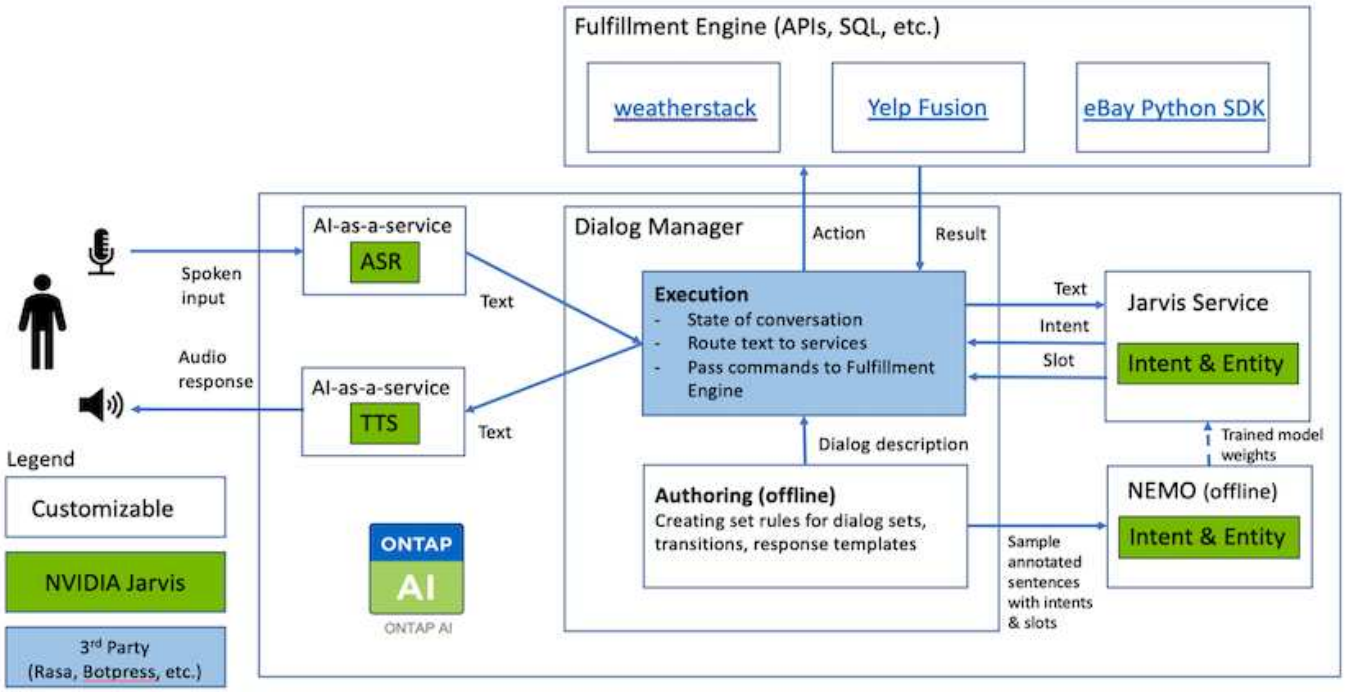
Nemo を使用して、アーカイブされた会話履歴のユーザ質問から複雑なインテントを認識するモデルをトレーニングしました。このトレーニングは、 Jarvis が提供したもの以外にも、小売バーチャルアシスタントの機能を拡張します。

小売業のユースケースの概要

NVIDIA Jarvis を使用して、スピーチやテキスト入力を受け付け、天気、関心のあるポイント、在庫価格に関する質問に回答できる仮想小売アシスタントを構築しました。会話型 AI システムでは、たとえば、天気や関心のある場所を指定していない場合は、フォローアップの質問をして会話の流れを記憶することができます。また、「タイ料理」や「ノートパソコンのメモリ」などの複雑なエンティティも認識します。 「ロサンゼルスで来週雨が降るだろうか？」など、自然言語の質問を理解しています。 小売バーチャルアシスタントのデモンストレーションは、にあります "小売ユースケースの状態とフローをカスタマイズします"。

解決策テクノロジ

次の図は、提案された会話型 AI システムアーキテクチャを示しています。システムは、音声信号またはテキスト入力で操作できます。音声入力が検出された場合、 Jarvis As-as-a-Service （ AIaaS ）は ASR を実行して、 Dialog Manager 用のテキストを生成します。Dialog Manager は、会話の状態を記憶し、テキストを対応するサービスにルーティングし、 Fulfillment Engine にコマンドを渡します。Jarvis NLP サービスは、テキストを受け取り、インテントとエンティティを認識し、それらのインテントとエンティティスロットをダイアログマネージャーに出力します。ダイアログマネージャーは、アクションをフルフィルメントエンジンに送信します。フルフィルメントエンジンは、回答ユーザーが照会するサードパーティ API または SQL データベースで構成されています。フルフィルメントエンジンから結果を受け取った後、 Dialog Manager はテキストを Jarvis TTS AIaaS にルーティングして、エンドユーザーの音声応答を生成します。NLP サービスがシステムとの対話をより多くのユーザが行うように改善されるように、会話履歴をアーカイブし、インテントや Nemo トレーニング用のスロットに注釈を付けることができます。



ハードウェア要件

この解決策は、 1 台の DGX ステーションと 1 台の AFF A220 ストレージシステムで検証済みです。Jarvis では、ディープニューラルネットワークの計算に T4 GPU または V100 GPU のいずれかが必要です。

次の表に、テストで解決策を実装するために必要なハードウェアコンポーネントを示します。

ハードウェア	数量
T4 または V100 GPU	1.
NVIDIA DGX ステーション	1.

ソフトウェア要件

次の表に、テストで解決策を実装するために必要なソフトウェアコンポーネントを示します。

ソフトウェア	バージョンまたはその他の情報
NetApp ONTAP データ管理ソフトウェア	9.6
Cisco NX-OS スイッチのファームウェア	7.0 （3） I6 （1）
NVIDIA DGX OS	4.0.4 - Ubuntu 18.04 LTS
NVIDIA Jarvis フレームワーク	EA v0.2 の 1 つです
NVIDIA Nemo	nvcr.io/nvidia / Nemo ： v0.10
Docker コンテナプラットフォーム	18.06.1-CE [e68fc7a]

概要

このセクションでは、仮想小売アシスタントの実装について詳しく説明します。

にサインアップできます "Jarvis Early Access プログラム" NVIDIA GPU Cloud （ NGC ）のジャービスコンテナにアクセスするため。NVIDIA から資格情報を受け取った後、以下の手順を使用して Jarvis を展開できます。

1. NGC へのサインオン
2. NGC に組織を設定します :'ea -2- ジャービス '.
3. Jarvis EA v0.2 資産の検索 : Jarvis コンテナは「プライベートレジストリ」＞「組織コンテナ」にあります。
4. 「 Jarvis 」を選択します。「 Model Scripts 」に移動し、「 Jarvis Quick Start 」をクリックします
5. すべてのアセットが正しく動作していることを確認します。
6. 独自のアプリケーションを構築するためのドキュメントを検索します。 PDF は「 M odel Scripts 」 > 「 Jarvis Documentation 」 > 「ファイルブラウザ」にあります。

小売ユースケースの状態とフローをカスタマイズします

特定のユースケースに合わせてダイアログマネージャーの状態とフローをカスタマイズできます。小売業の例では、次の 4 つの YAML ファイルを使用して、異なるインテントに従って会話を誘導します。

各ファイルについて、次のファイル名と概要のリストを用意します。

- `main_flow.yml`: 主な会話の流れと状態を定義し、必要に応じて他の 3 つの YAML ファイルにフローを指示します。
- `retail_flow.yml`: 小売業または関心のある点に関する質問に関連する状態を含みます。システムは最も近い店の情報、または与えられた項目の価格を提供する。
- `weater_flow.yml`: 天気に関する質問に関連する州を含みます。場所を特定できない場合は、フォローアップの質問をして明確にします。
- `error_flow.yml`: ユーザのインテントが上記の 3 つの YAML ファイルに入っていないケースを処理します。エラーメッセージを表示した後、システムはユーザーの質問を受け入れるように再経路化します。次のセクションでは、これらの YAML ファイルの詳細な定義を示します。

main_flow.yml

```
name: JarvisRetail
intent_transitions:
  jarvis_error: error
  price_check: retail_price_check
  inventory_check: retail_inventory_check
  store_location: retail_store_location
  weather.weather: weather
  weather.temperature: temperature
  weather.sunny: sunny
  weather.cloudy: cloudy
```

```
  weather.snow: snow
  weather.rainfall: rain
  weather.snow_yes_no: snowfall
  weather.rainfall_yes_no: rainfall
  weather.temperature_yes_no: tempyesno
  weather.humidity: humidity
  weather.humidity_yes_no: humidity
  navigation.startnavigationpoi: retail # Transitions should be context
and slot based. Redirecting for now.
  navigation.geteta: retail
  navigation.showdirection: retail
  navigation.showmappoi: idk_what_you_talkin_about
  nomatch.none: idk_what_you_talkin_about
states:
  init:
    type: message_text
    properties:
      text: "Hi, welcome to NARA retail and weather service. How can I
help you?"
  input_intent:
    type: input_context
    properties:
      nlp_type: jarvis
      entities:
        intent: dontcare
# This state is executed if the intent was not understood
  dont_get_the intent:
    type: message_text_random
    properties:
      responses:
        - "Sorry I didn't get that! Please come again."
        - "I beg your pardon! Say that again?"
        - "Are we talking about weather? What would you like to know?"
        - "Sorry I know only about the weather"
        - "You can ask me about the weather, the rainfall, the
temperature, I don't know much more"
      delay: 0
    transitions:
      next_state: input_intent
  idk_what_you_talkin_about:
    type: message_text_random
    properties:
      responses:
        - "Sorry I didn't get that! Please come again."
        - "I beg your pardon! Say that again?"
        - "Are we talking about retail or weather? What would you like to
```

```
know?"
        - "Sorry I know only about retail and the weather"
        - "You can ask me about retail information or the weather, the
rainfall, the temperature. I don't know much more."
      delay: 0
    transitions:
      next_state: input_intent
  error:
    type: change_context
    properties:
        update_keys:
           intent: 'error'
    transitions:
        flow: error_flow
  retail_inventory_check:
    type: change_context
    properties:
        update_keys:
           intent: 'retail_inventory_check'
    transitions:
        flow: retail_flow
  retail_price_check:
    type: change_context
    properties:
        update_keys:
           intent: 'check_item_price'
    transitions:
        flow: retail_flow
  retail_store_location:
    type: change_context
    properties:
        update_keys:
           intent: 'find_the_store'
    transitions:
        flow: retail_flow
  weather:
    type: change_context
    properties:
        update_keys:
           intent: 'weather'
    transitions:
        flow: weather_flow
  temperature:
    type: change_context
    properties:
        update_keys:
```

```
            intent: 'temperature'
      transitions:
          flow: weather_flow
    rainfall:
      type: change_context
      properties:
          update_keys:
             intent: 'rainfall'
      transitions:
          flow: weather_flow
    sunny:
      type: change_context
      properties:
          update_keys:
             intent: 'sunny'
      transitions:
          flow: weather_flow
    cloudy:
      type: change_context
      properties:
          update_keys:
             intent: 'cloudy'
      transitions:
          flow: weather_flow
    snow:
      type: change_context
      properties:
          update_keys:
             intent: 'snow'
      transitions:
          flow: weather_flow
    rain:
      type: change_context
      properties:
          update_keys:
             intent: 'rain'
      transitions:
          flow: weather_flow
    snowfall:
        type: change_context
        properties:
            update_keys:
               intent: 'snowfall'
        transitions:
            flow: weather_flow
    tempyesno:
```

```
      type: change_context
      properties:
          update_keys:
             intent: 'tempyesno'
      transitions:
          flow: weather_flow
  humidity:
      type: change_context
      properties:
          update_keys:
             intent: 'humidity'
      transitions:
          flow: weather_flow
  end_state:
    type: reset
    transitions:
      next_state: init
```

retail _flow.yml

```
name: retail_flow
states:
  store_location:
    type: conditional_exists
    properties:
      key: '{{location}}'
    transitions:
      exists: retail_state
      notexists: ask_retail_location
  retail_state:
    type: Retail
    properties:
    transitions:
      next_state: output_retail
  output_retail:
      type: message_text
      properties:
        text: '{{retail_status}}'
      transitions:
        next_state: input_intent
  ask_retail_location:
    type: message_text
    properties:
      text: "For which location? I can find the closest store near you."
    transitions:
```

```
      next_state: input_retail_location
  input_retail_location:
    type: input_user
    properties:
      nlp_type: jarvis
      entities:
        slot: location
      require_match: true
    transitions:
      match: retail_state
      notmatch: check_retail_jarvis_error
  output_retail_acknowledge:
    type: message_text_random
    properties:
      responses:
        - 'ok in {{location}}'
        - 'the store in {{location}}'
        - 'I always wanted to shop in {{location}}'
      delay: 0
    transitions:
      next_state: retail_state
  output_retail_notlocation:
    type: message_text
    properties:
      text: "I did not understand the location. Can you please repeat?"
    transitions:
      next_state: input_intent
  check_rerail_jarvis_error:
    type: conditional_exists
    properties:
      key: '{{jarvis_error}}'
    transitions:
      exists: show_retail_jarvis_api_error
      notexists: output_retail_notlocation
  show_retail_jarvis_api_error:
    type: message_text
    properties:
      text: "I am having troubled understanding right now. Come again on
that?"
    transitions:
      next_state: input_intent
```

weater_flow.yml

```
name: weather_flow
```

```
states:
  check_weather_location:
    type: conditional_exists
    properties:
      key: '{{location}}'
    transitions:
      exists: weather_state
      notexists: ask_weather_location
  weather_state:
    type: Weather
    properties:
    transitions:
      next_state: output_weather
  output_weather:
      type: message_text
      properties:
        text: '{{weather_status}}'
      transitions:
        next_state: input_intent
  ask_weather_location:
    type: message_text
    properties:
      text: "For which location?"
    transitions:
      next_state: input_weather_location
  input_weather_location:
    type: input_user
    properties:
      nlp_type: jarvis
      entities:
        slot: location
      require_match: true
    transitions:
      match: weather_state
      notmatch: check_jarvis_error
  output_weather_acknowledge:
    type: message_text_random
    properties:
      responses:
        - 'ok in {{location}}'
        - 'the weather in {{location}}'
        - 'I always wanted to go in {{location}}'
      delay: 0
    transitions:
      next_state: weather_state
  output_weather_notlocation:
```

```
    type: message_text
    properties:
      text: "I did not understand the location, can you please repeat?"
    transitions:
      next_state: input_intent
  check_jarvis_error:
    type: conditional_exists
    properties:
      key: '{{jarvis_error}}'
    transitions:
      exists: show_jarvis_api_error
      notexists: output_weather_notlocation
  show_jarvis_api_error:
    type: message_text
    properties:
      text: "I am having troubled understanding right now. Come again on
that, else check jarvis services?"
    transitions:
      next_state: input_intent
```

ERROR_FLOW.yml

```
name: error_flow
states:
  error_state:
    type: message_text_random
    properties:
      responses:
        - "Sorry I didn't get that!"
        - "Are we talking about retail or weather? What would you like to
know?"
        - "Sorry I know only about retail information or the weather"
        - "You can ask me about retail information or the weather, the
rainfall, the temperature. I don't know much more"
        - "Let's talk about retail or the weather!"
      delay: 0
    transitions:
      next_state: input_intent
```

フルフィルメントエンジンとしてサードパーティ **API** に接続します

次のサードパーティ API を欠品補充エンジンとして回答の質問に関連付けました。

- "WeatherStack API": 指定された場所の天候、温度、降雨および雪を返す。

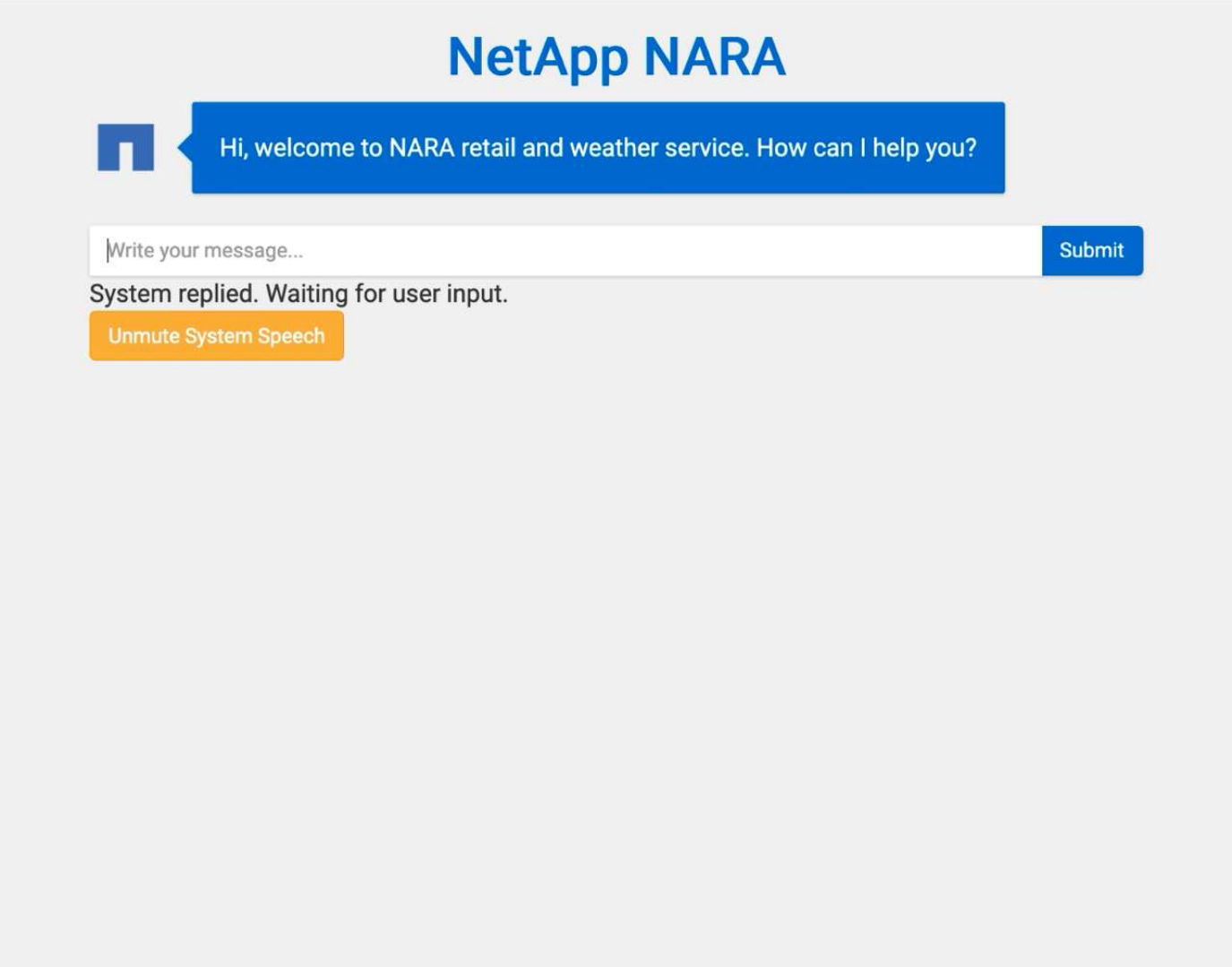
- "Yelp Fusion API": 指定された場所で最も近いストア情報を返します。
- "eBay Python SDK": 指定されたアイテムの価格を返します。

NetApp Retail Assistant のデモ

NetApp Retail Assistant （奈良）のデモビデオを録画しました。

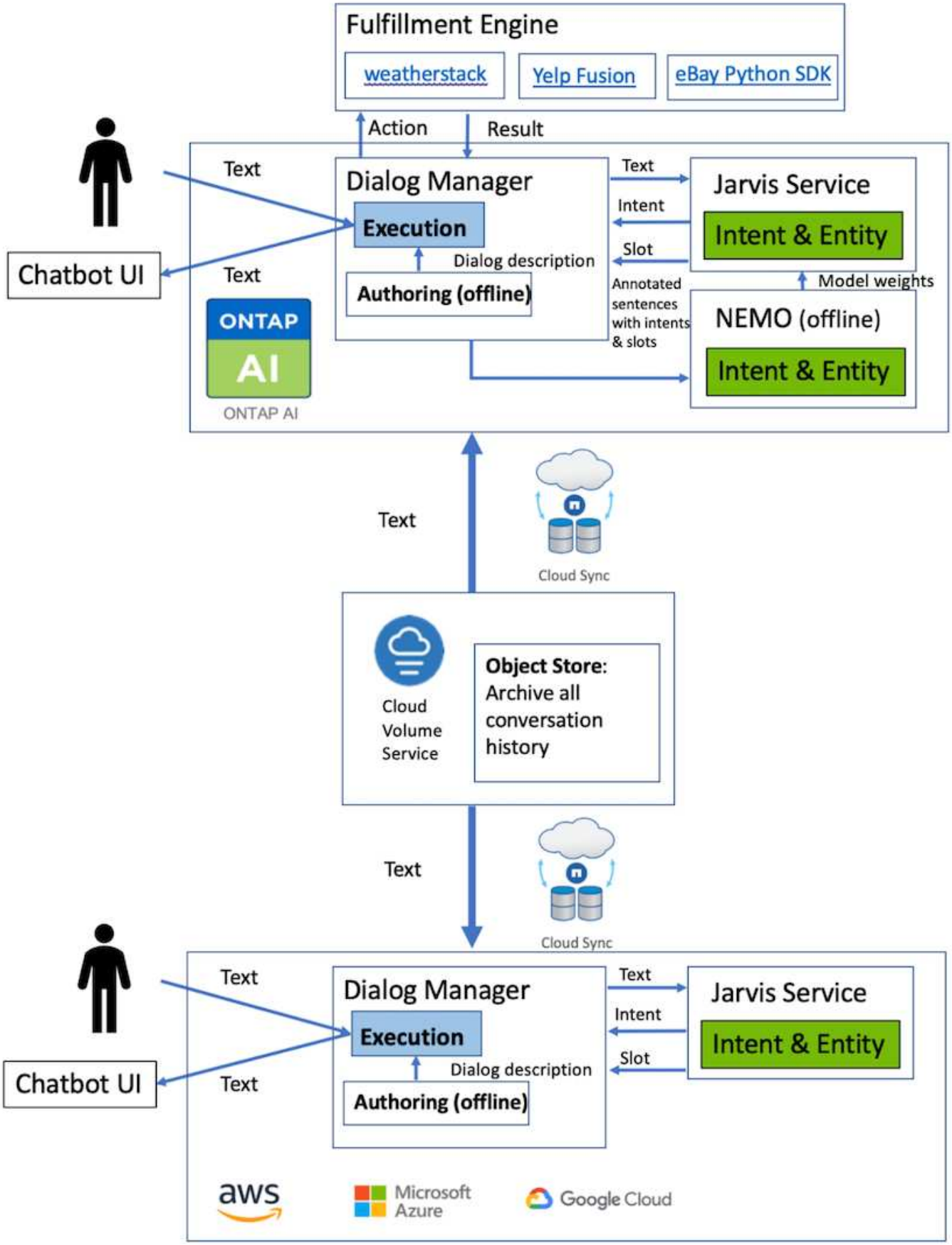
奈良のデモ映像

奈良のデモ映像



NetApp BlueXPのコピーと同期を使用して会話履歴をアーカイブ

会話履歴を1日1回CSVファイルにダンプすることで、BlueXPのコピーと同期を活用してローカルストレージにログファイルをダウンロードできます。次の図は、JarvisをオンプレミスとパブリッククラウドにBlueXPのCopy and Syncを使用してNemoトレーニングに関する会話履歴を送信するアーキテクチャを示しています。Nemo の訓練の詳細はセクションで見つけることができる "Nemo トレーニングを使用してインテントモデルを拡張する"。


Fulfillment Engine
weatherstack
Yelp Fusion
eBay Python SDK
Action
Result
Text
Dialog Manager
Execution
Dialog description
Authoring (offline)
Text
Intent
Slot
Jarvis Service
Intent & Entity
Model weights
Annotated sentences with intents & slots
NEMO (offline)
Intent & Entity
Chatbot UI
Text
ONTAP
AI
ONTAP AI
Text
Cloud Sync
Cloud Volume Service
Object Store: Archive all conversation history
Text
Cloud Sync
Text
Dialog Manager
Execution
Dialog description
Authoring (offline)
Text
Intent
Slot
Jarvis Service
Intent & Entity
Chatbot UI
Text
aws
Microsoft Azure
Google Cloud

NVIDIA Nemo は、会話型 AI アプリケーションを作成するために NVIDIA が開発したツールキットです。このツールキットには、 ASR 、 NLP 、 TTS 用のトレーニング済みモジュールのコレクションが含まれています。これにより、研究者やデータサイエンティストは複雑なニューラルネットワークアーキテクチャを簡単に構築し、独自のアプリケーションの設計に集中できるようになります。

前の例で示したように、奈良は限られたタイプの質問しか処理できない。これは、トレーニング済みの NLP モデルでは、これらのタイプの質問についてのみトレーニングを受けているためです。奈良がより幅広い質問に対応できるようにするには、独自のデータセットを使って再トレーニングする必要があります。ここでは、Nemo を使用して NLP モデルを拡張し、要件を満たす方法を示します。まず、奈良から収集したログを Nemo の形式に変換し、 NLP モデルを強化するためのデータセットを使って学習します。

モデル

私たちの目標は、ユーザーの好みに基づいてアイテムを並べ替えることです。たとえば、奈良に最高級の寿司レストランを提案したり、奈良に最も安い価格のジーンズを探してもらうようにしたりすることができます。このためには、 Nemo で提供されているインテント検出とスロット充填モデルをトレーニングモデルとして使用します。このモデルにより、奈良は検索の好みを理解することができる。

データの準備

モデルをトレーニングするには、このタイプの質問のデータセットを収集し、 Nemo 形式に変換します。ここでは、モデルのトレーニングに使用するファイルをリストしました。

dict.intents.csv

このファイルには、 Nemo に理解してもらいたいすべてのインテントが一覧表示されます。ここでは、主なインテントが 2 つあり、 1 つのインテントは、主なインテントのどれにも当てはまらない質問を分類するために使用されます。

```
price_check
find_the_store
unknown
```

dict.slots.csv

このファイルには、トレーニングの質問にラベルを付けることができるすべてのスロットが記載されています。

```
B-store.type
B-store.name
B-store.status
B-store.hour.start
B-store.hour.end
B-store.hour.day
B-item.type
```

```
B-item.name
B-item.color
B-item.size
B-item.quantity
B-location
B-cost.high
B-cost.average
B-cost.low
B-time.period_of_time
B-rating.high
B-rating.average
B-rating.low
B-interrogative.location
B-interrogative.manner
B-interrogative.time
B-interrogative.personal
B-interrogative
B-verb
B-article
I-store.type
I-store.name
I-store.status
I-store.hour.start
I-store.hour.end
I-store.hour.day
I-item.type
I-item.name
I-item.color
I-item.size
I-item.quantity
I-location
I-cost.high
I-cost.average
I-cost.low
I-time.period_of_time
I-rating.high
I-rating.average
I-rating.low
I-interrogative.location
I-interrogative.manner
I-interrogative.time
I-interrogative.personal
I-interrogative
I-verb
I-article
O
```

鉄道 **.tsv**

これが主なトレーニングデータセットです。各行は、 dict.intent.csv ファイルのインテントカテゴリのリストに続く質問から始まります。ラベルはゼロから列挙されます。

train slots.tsv

```
20 46 24 25 6 32 6
52 52 24 6
23 52 14 40 52 25 6 32 6
…
```

モデルのトレーニング

```
docker pull nvcr.io/nvidia/nemo:v0.10
```

次に、次のコマンドを使用してコンテナを起動します。このコマンドでは、軽量なトレーニング用の演習であるため、コンテナで使用する GPU は 1 つに制限されます（ GPU ID = 1 ）。また、ローカルワークスペース / ワークスペース /Nemo/ をコンテナ /Nemo 内のフォルダにマッピングします。

```
NV_GPU='1' docker run --runtime=nvidia -it --shm-size=16g \
                        --network=host --ulimit memlock=-1 --ulimit
stack=67108864 \
                        -v /workspace/nemo:/nemo\
                        --rm nvcr.io/nvidia/nemo:v0.10
```

コンテナ内では、事前にトレーニングされたオリジナルの BERT モデルから開始する場合、次のコマンドを使用してトレーニング手順を開始できます。data_dir は、トレーニングデータのパスを設定する引数です。work_dir では ' チェックポイント・ファイルを保存する場所を設定できます

```
cd examples/nlp/intent_detection_slot_tagging/
python joint_intent_slot_with_bert.py \
    --data_dir /nemo/training_data\
    --work_dir /nemo/log
```

新しいトレーニングデータセットがあり、以前のモデルを改善したい場合は、次のコマンドを使用して停止した時点から続行できます。checkpoint_dir は ' 前のチェックポイント・フォルダへのパスを取得します

```
cd examples/nlp/intent_detection_slot_tagging/
python joint_intent_slot_infer.py \
    --data_dir /nemo/training_data \
    --checkpoint_dir /nemo/log/2020-05-04_18-34-20/checkpoints/ \
    --eval_file_prefix test
```

モデルを推論します

トレーニング済みモデルのパフォーマンスは、一定の期間の経過後に検証する必要があります。次のコマンドを使用すると、１つずつクエリをテストできます。たとえば、このコマンドでは、モデルがクエリの意図を正しく識別できるかどうかを確認します。クエリの目的は、「ここで最高のパスタを取得できる」です。

```
cd examples/nlp/intent_detection_slot_tagging/
python joint_intent_slot_infer_b1.py \
--checkpoint_dir /nemo/log/2020-05-29_23-50-58/checkpoints/ \
--query "where can i get the best pasta" \
--data_dir /nemo/training_data/ \
--num_epochs=50
```

次に、推論からの出力を示します。出力では、トレーニング済みモデルが意図を正しく予測し、関心のあるキーワードを返すことができます。これらのキーワードを使うことで、奈良はユーザが欲しいものを検索し、より正確な検索を行うことができるようになります。

```
[NeMo I 2020-05-30 00:06:54 actions:728] Evaluating batch 0 out of 1
[NeMo I 2020-05-30 00:06:55 inference_utils:34] Query: where can i get the
best pasta
[NeMo I 2020-05-30 00:06:55 inference_utils:36] Predicted intent:       1
find_the_store
[NeMo I 2020-05-30 00:06:55 inference_utils:50] where   B-
interrogative.location
[NeMo I 2020-05-30 00:06:55 inference_utils:50] can     O
[NeMo I 2020-05-30 00:06:55 inference_utils:50] i       O
[NeMo I 2020-05-30 00:06:55 inference_utils:50] get     B-verb
[NeMo I 2020-05-30 00:06:55 inference_utils:50] the     B-article
[NeMo I 2020-05-30 00:06:55 inference_utils:50] best    B-rating.high
[NeMo I 2020-05-30 00:06:55 inference_utils:50] pasta   B-item.type
```

まとめ

真の会話型 AI システムは、人間のような対話を行い、文脈を理解し、インテリジェントな応答を提供します。このような AI モデルは、多くの場合、巨大で非常に複雑です。NVIDIA GPU とネットアップストレージを使用することで、最新の大規模な言語モデルをトレーニングし、最適化して推論を迅速に実行できます。これは、大規模で複雑

な AI モデルと比べて高速な AI モデルのトレードオフを終えるための大きなストライドです。GPU に最適化された言語理解モデルは、医療、小売、金融サービスなどの業界向け AI アプリケーションに統合でき、高度なデジタル音声アシスタントをスマートスピーカーやカスタマーサービスラインに搭載できます。これらの高品質な会話型 AI システムにより、さまざまな業種の企業が、お客様との取引においてこれまで達成できなかったパーソナライズされたサービスを提供できます。

Jarvis は、バーチャルアシスタント、デジタルアバター、マルチモーダルセンサー Fusion （ ASR/NLP/TTS とフュージョンされた CV ）、または ASR/NLP/TTS/CV スタンドアロンの使用例（転写など）の導入を可能にします。私たちは、天気、関心のあるポイント、在庫価格に関する回答の質問を行えるバーチャル小売アシスタントを開発しました。また、BlueXP CopyとSyncを使用して会話履歴をアーカイブし、新しいデータについてNemoモデルをトレーニングすることで、会話型AIシステムの自然言語理解機能を向上させる方法についてもデモンストレーションを行いました。

謝辞

このホワイトペーパーに貢献したことを、 NVIDIA の著名な同僚によって認められたことに感謝の意を表します。 Davide Onfrio 、 Alex Qi 、 Sicong Ji 、 Marty Jain 、 Robert Sohigian 。また、ネットアップの主要チームメンバーである Santosh Rao 、 David Arnette 、 Michael Oglesby 、 Brent Davis 、 Andy Sayare の協力も感謝しています。 Erik Mulder 氏、 Mike McNamara 氏。

本ホワイトペーパーの作成に大きく貢献した洞察と専門知識を提供したすべての方々に心から感謝します。

追加情報の検索場所

このドキュメントに記載されている情報の詳細については、次のリソースを参照してください。

- NVIDIA DGX Station 、 V100 GPU 、 GPU Cloud
  - NVIDIA DGX ステーションhttps://www.nvidia.com/en-us/data-center/dgx-station/["https://www.nvidia.com/en-us/data-center/dgx-station/"^]
  - NVIDIA V100 Tensor コア GPUhttps://www.nvidia.com/en-us/data-center/tesla-v100/["https://www.nvidia.com/en-us/data-center/tesla-v100/"^]
  - NVIDIA NGChttps://www.nvidia.com/en-us/gpu-cloud/["https://www.nvidia.com/en-us/gpu-cloud/"^]
- NVIDIA Jarvis マルチモーダルフレームワーク
  - NVIDIA Jarvishttps://developer.nvidia.com/nvidia-jarvis["https://developer.nvidia.com/nvidia-jarvis"^]
  - NVIDIA Jarvis Early Accesshttps://developer.nvidia.com/nvidia-jarvis-early-access["https://developer.nvidia.com/nvidia-jarvis-early-access"^]
- NVIDIA Nemo
  - NVIDIA Nemohttps://developer.nvidia.com/nvidia-nemo["https://developer.nvidia.com/nvidia-nemo"^]
  - 開発者ガイドhttps://nvidia.github.io/NeMo/["https://nvidia.github.io/NeMo/"^]
- NetApp AFF システム
  - NetApp AFF A シリーズのデータシートhttps://www.netapp.com/us/media/ds-3582.pdf["https://www.netapp.com/us/media/ds-3582.pdf"^]

- ネットアップの All Flash FAS 向けフラッシュソリューションの利点https://www.netapp.com/us/media/ds-3733.pdf["https://www.netapp.com/us/media/ds-3733.pdf"^]
- ONTAP 9 情報ライブラリhttp://mysupport.netapp.com/documentation/productlibrary/index.html?productID=62286["http://mysupport.netapp.com/documentation/productlibrary/index.html?productID=62286"^]
- NetApp ONTAP FlexGroup Volume テクニカルレポートhttps://www.netapp.com/us/media/tr-4557.pdf["https://www.netapp.com/us/media/tr-4557.pdf"^]

- NetApp ONTAP AI
  - DGX-1 と Cisco Networking Design Guide による ONTAP AIhttps://www.netapp.com/us/media/nva-1121-design.pdf["https://www.netapp.com/us/media/nva-1121-design.pdf"^]
  - DGX-1 と Cisco Networking Deployment Guide を使用した ONTAP AIhttps://www.netapp.com/us/media/nva-1121-deploy.pdf["https://www.netapp.com/us/media/nva-1121-deploy.pdf"^]
  - DGX-1 と Mellanox のネットワーキング設計ガイドで構成される ONTAP AIhttp://www.netapp.com/us/media/nva-1138-design.pdf["http://www.netapp.com/us/media/nva-1138-design.pdf"^]
  - DGX-2 を使用した ONTAP AI 設計ガイドhttps://www.netapp.com/us/media/nva-1135-design.pdf["https://www.netapp.com/us/media/nva-1135-design.pdf"^]

TR-4858 ：『 NetApp Orchestration 解決策 with Run ： AI 』

ネットアップの Yaron Goldberg 、 Run ： AI 、 David Arnette 、 Sung-Han Lin

NetApp AFF ストレージシステムは、卓越したパフォーマンスと業界をリードするハイブリッドクラウドデータ管理機能を提供します。ネットアップと Run ： AI は、 NetApp ONTAP AI 解決策の独自の機能である人工知能（ AI ）と機械学習（ ML ）のワークロードに対応し、エンタープライズクラスのパフォーマンス、信頼性、サポートを提供していることを実証するために提携しました。実行： AI ワークロードの AI オーケストレーションにより、 Kubernetes ベースのスケジュールとリソース利用プラットフォームが追加され、研究者が GPU 利用率を管理、最適化できるようになります。ネットアップ、 NVIDIA 、 Run ： AI の解決策を組み合わせた NVIDIA DGX システムは、エンタープライズ AI ワークロードに特化したインフラスタックを提供します。このテクニカルレポートでは、さまざまなユースケースや業種に対応した会話型 AI システムを構築するお客様向けのガイダンスを提供します。Run の導入に関する情報、 AI と NetApp AFF A800 ストレージシステムが記載されています。 AI イニシアチブを短期間で成功に導くためのリファレンスアーキテクチャとして機能します。

解決策の対象となるグループは次のとおりです。

- AI 開発のためのソリューションを設計するエンタープライズアーキテクト コンテナ化などの Kubernetes ベースのユースケース向けのモデルとソフトウェア マイクロサービス
- データサイエンティストは、効率的なモデルを実現するための効率的な方法を探しています 複数のチームとで構成されるクラスタ環境における開発目標 プロジェクト
- 本番モデルの保守と実行を担当するデータエンジニア
- エグゼクティブや IT の意思決定者、ビジネスリーダー Kubernetes クラスタのリソース利用率を最適化す

る方法をご確認ください AI 導入の市場投入までの時間を短縮できます

解決策の概要

NetApp ONTAP AI と AI のコントロールプレーン

ネットアップと NVIDIA が開発、検証した NetApp ONTAP AI アーキテクチャは、 NVIDIA DGX システムとネットアップのクラウド対応ストレージシステムを基盤としています。このリファレンスアーキテクチャには、IT 組織に次のようなメリットがあります。

- 設計の複雑さを解消
- コンピューティングとストレージを個別に拡張できます
- 小規模構成から始めて、シームレスに拡張できます
- は、さまざまなパフォーマンスとに対応するさまざまなストレージオプションを提供します コストポイント

NetApp ONTAP AI は、 DGX システムと NetApp AFF A800 ストレージシステムを最先端のネットワークと緊密に統合します。NetApp ONTAP AI システムと DGX システムでは、設計の複雑さと推測に頼らず、 AI 導入を簡易化できます。お客様は小規模構成から始めて、システムを中断なく拡張できます。同時に、エッジ、コア、クラウドにわたってデータをインテリジェントに管理できます。

ネットアップの AI コントロールプレーンは、データサイエンティストとデータエンジニアを対象とした、フルスタックの AI 、 ML 、ディープラーニング（ DL ）のデータと実験管理解決策です。AI の利用が拡大するにつれて、ワークロードの拡張性やデータの可用性など、さまざまな課題が生じています。NetApp AI コントロールプレーンは、データネームスペースの迅速なクローニングを Git レポジトリと同様に行う機能や、トレーサビリティとバージョン管理のためにデータやモデルベースラインをほぼ瞬時に作成する AI トレーニングワークフローを定義して実装するなど、これらの課題に対処します。NetApp AI コントロールプレーンを使用すると、サイトやリージョン間でデータをシームレスにレプリケートし、 Jupyter Notebook ワークスペースをすばやくプロビジョニングして、大規模なデータセットにアクセスできます。

実行： AI ワークロードのオーケストレーション向けの AI プラットフォーム

実行： AI は、世界初の AI インフラストラクチャのオーケストレーションおよび仮想化プラットフォームを構築しました。基盤となるハードウェアからワークロードを抽象化することで、 Run ： AI は、 GPU リソースの共有プールを作成します。この共有プールを動的にプロビジョニングし、 AI ワークロードの効率的なオーケストレーションと GPU の使用の最適化を実現します。データサイエンティストは、大量の GPU パワーをシームレスに消費して研究を向上させ、加速させることができます。一方、 IT チームは、リソースのプロビジョニング、キューイング、利用率に関する一元化されたクロスサイト管理とリアルタイムの可視性を維持します。Run ： AI プラットフォームは Kubernetes を基盤として構築されているため、既存の IT ワークフローやデータサイエンスワークフローと簡単に統合できます。

Run ： AI プラットフォームには、次のようなメリットがあります。

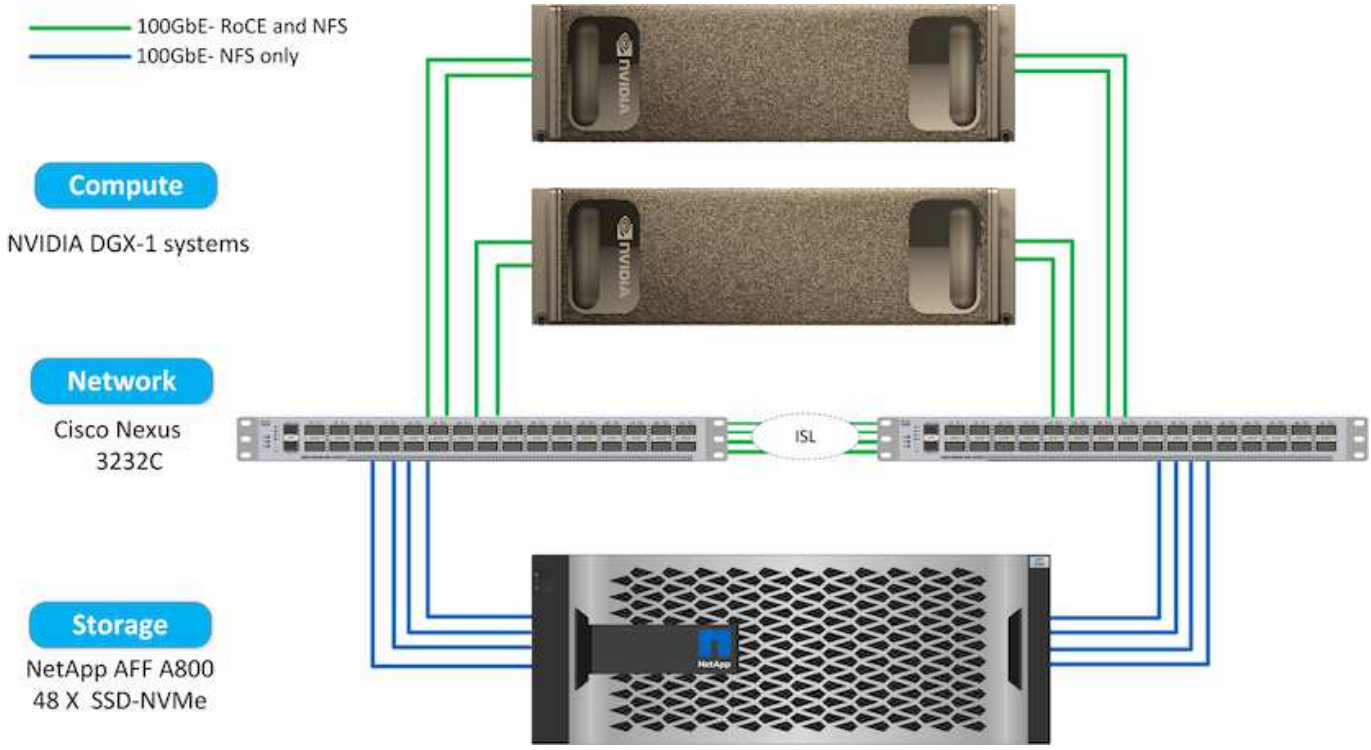
- * イノベーションにかかる時間を短縮 * Run ： AI リソースのプール化、キューイング、優先付けのメカニズムをネットアップストレージシステムと組み合わせることで、研究者たちはインフラ管理の苦労から取り取り除かれ、データサイエンスに専念することができます。実行： AI とネットアップのお客様は、コンピューティングやデータパイプラインのボトルネックを発生させることなく、必要な数のワークロードを実行することで生産性を向上できます。
- * チームの生産性向上。 * 実行： AI 公正性アルゴリズムにより、すべてのユーザーとチームが公平なリソースを共有できるようになります。優先度の高いプロジェクトを中心としたポリシーをあらかじめ設定しておくことで、あるユーザやチームから別のユーザやチームにリソースを動的に割り当てることができ、

誰もが切望した GPU リソースにタイムリーにアクセスできるようになります。

- * GPU 利用率の向上。 * 実行： AI スケジューラを使用すると、 Kubernetes での分散トレーニング用に、フラクショナルな GPU 、整数型 GPU 、複数の GPU ノードを簡単に利用できます。このように、 AI ワークロードは容量ではなくニーズに基づいて実行されます。データサイエンスチームは、 1 つのインフラでより多くの AI 実験を実行できるようになります。

解決策テクノロジ

この解決策は、 1 台の NetApp AFF A800 システム、 2 台の DGX-1 サーバ、 2 台の Cisco Nexus 3232C 100GbE スイッチで実装されました。DGX-1 サーバはそれぞれ 4 本の 100GbE 接続で Nexus スイッチに接続されます。この接続は、 Converged Ethernet （ RoCE ）を介したリモートダイレクトメモリアクセス（ RDMA ）を使用する GPU 間通信に使用されます。NFS ストレージアクセス用の従来の IP 通信も、これらのリンクで行われます。各ストレージコントローラは、 4 つの 100GbE リンクを使用してネットワークスイッチに接続されています。次の図に、このテクニカルレポートですべてのテストシナリオに使用した ONTAP AI 解決策アーキテクチャを示します。



この解決策で使用されているハードウェア

この解決策は、 ONTAP AI リファレンスアーキテクチャ DGX 1 ノードと AFF A800 ストレージシステム 1 台を使用して検証されました。を参照してください "NVA-1121." を参照してください。

次の表に、テストで解決策を実装するために必要なハードウェアコンポーネントを示します。

ハードウェア	数量
DGX-1 システム	2.
AFF A800	1.

ハードウェア	数量
Nexus 3232C スイッチ	2.

ソフトウェア要件

この解決策は、 Run ： AI オペレータがインストールされた基本的な Kubernetes 環境で検証されました。Kubernetes はを使用して導入されました "NVIDIA DeepOps のことです" 導入エンジン：本番環境に必要なすべてのコンポーネントを導入します。DeepOps は自動的に導入されます "NetApp Trident" Kubernetes 環境との永続的なストレージ統合のために、デフォルトのストレージクラスが作成されました。これにより、コンテナは AFF A800 ストレージシステムのストレージを活用できます。Trident と ONTAP AI の Kubernetes の詳細については、を参照してください "TR-4798"。

次の表に、テストで解決策を実装するために必要なソフトウェアコンポーネントを示します。

ソフトウェア	バージョンまたはその他の情報
NetApp ONTAP データ管理ソフトウェア	9.6p4
Cisco NX-OS スイッチのファームウェア	7.0 （ 3 ） I6 （ 1 ）
NVIDIA DGX OS	4.0.4 - Ubuntu 18.04 LTS
Kubernetes のバージョン	1.17
Trident のバージョン	20.04.0
AI CLI を実行	v2.1.13
実行： AI Orchestration Kubernetes Operator バージョン	1.0.39
Docker コンテナプラットフォーム	18.06.1-CE [e68fc7a]

実行に必要なその他のソフトウェア要件： AI は、から入手できます "AI GPU クラスタの前提条件を実行します"。

クラスタと **GPU** の利用率を最適化して実行： **AI**

以下のセクションでは、この検証で実行した AI のインストール、テストシナリオ、結果について詳しく説明します。

TensorFlow ベンチマークを含む業界標準のベンチマークツールを使用して、このシステムの動作とパフォーマンスを検証しました。ImageNet データセットを使用して ResNet-50 をトレーニングしました。これは、画像分類のための有名な Convolutional Neural Network （ CNN ；畳み込みニューラルネットワーク） DL モデルです。ResNet-50 は、処理時間を短縮して正確なトレーニング結果を提供します。これにより、ストレージに対する十分な需要を喚起することができました。

「 **AI Installation** 」を実行します

Run ： AI をインストールするには、次の手順を実行します。

1. DeepOps を使用して Kubernetes クラスタをインストールし、ネットアップのデフォルトストレージクラスを設定します。
2. GPU ノードの準備：

a. NVIDIA ドライバが GPU ノードにインストールされていることを確認します。

b. 「 nvidia - Docker 」がインストールされ、デフォルトの Docker ランタイムとして設定されていることを確認します。

3. インストール実行： AI ：

a. にログインします "AI 管理 UI を実行" クラスタを作成できます。

b. 作成した「 runai-operator-<clustername> .yaml 」ファイルをダウンロードします。

c. オペレータ設定を Kubernetes クラスタに適用します。

```
kubectl apply -f runai-operator-<clustername>.yaml
```

4. インストールを確認します。

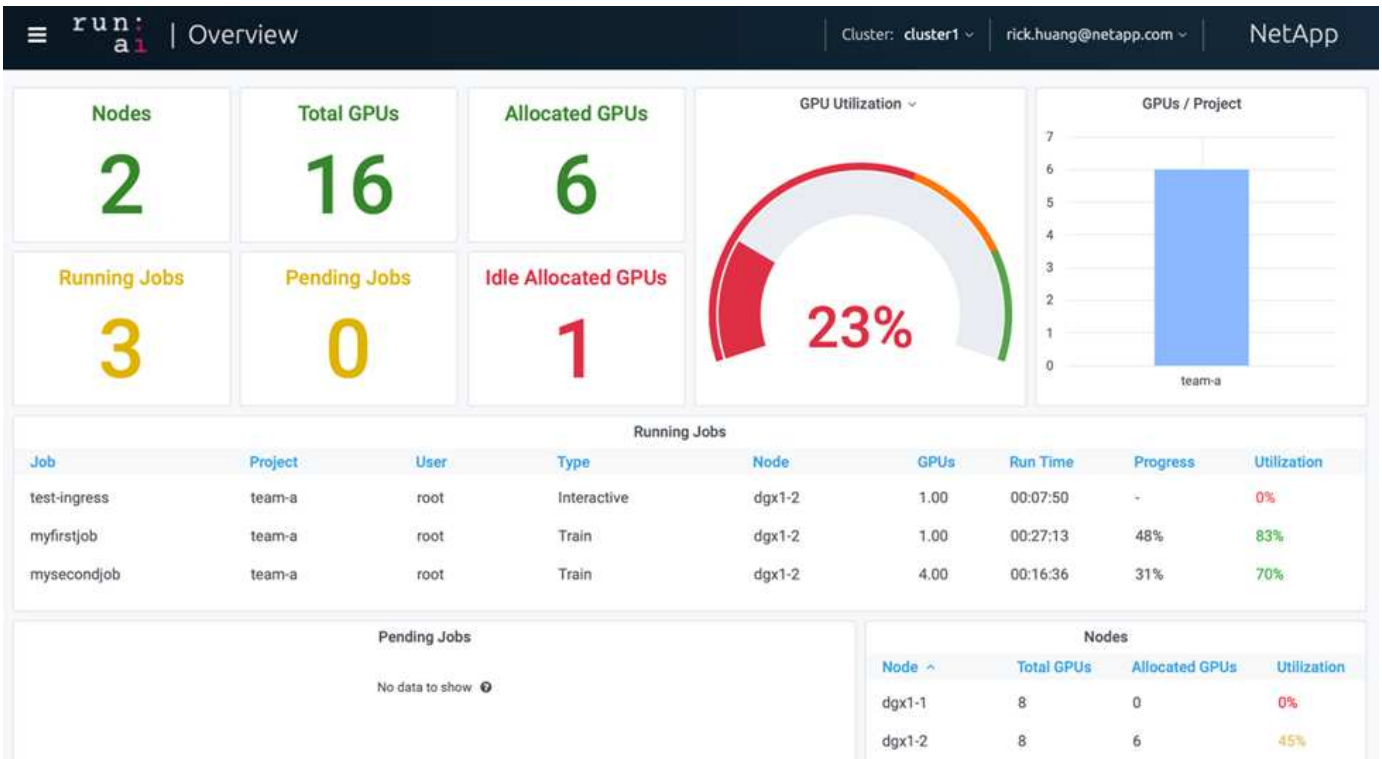
a. に進みます "https://app.run.ai/"。

b. 概要ダッシュボードに移動します。

c. 右上の GPU の数が、想定される GPU の数と GPU ノードの数をすべてサーバのリストに表示していることを確認します。実行： AI 導入の詳細については、を参照してください "Run ： AI をオンプレミスの Kubernetes クラスタにインストール" および "Run ： AI CLI のインストール"。

「 **AI Dashboards and Views** 」を実行します

Run ： AI を Kubernetes クラスタにインストールし、コンテナを正しく設定したら、次のダッシュボードとビューが表示されます "https://app.run.ai" 次の図に示すように、ブラウザで設定します。



2 つの DGX-1 ノードによってクラスタ内に提供される GPU は合計 16 個です。ノード数、使用可能な GPU

の総数、ワークロードに割り当てられている GPU の数、実行中のジョブの総数、保留中のジョブ、およびアイドル状態に割り当てられている GPU の数を確認できます。右側のバー図は、プロジェクトごとの GPU を示しています。これは、各チームがどのようにクラスタリソースを使用しているかをまとめたものです。中央には、ジョブ名、プロジェクト、ユーザー、ジョブタイプなど、現在実行中のジョブとジョブの詳細のリストが表示されます。 各ジョブが実行されているノード、そのジョブに割り当てられている GPU の数、ジョブの現在の実行時間、ジョブの進捗状況、およびそのジョブの GPU 利用率。1 つのチーム（「 TEAM 」）から送信された実行中のジョブは 3 つしかないため、クラスタの使用率が低いことに注意してください（ GPU 使用率は 23% ）。

次のセクションでは、「プロジェクト」タブで複数のチームを作成し、各チームに GPU を割り当てることで、クラスタあたりのユーザ数が多いときにクラスタの使用率を最大限に高め、リソースを管理する方法を説明します。テストシナリオは、トレーニング、推論、対話型のワークロードでメモリリソースと GPU リソースが共有されているエンタープライズ環境をシミュレートしたものです。

データサイエンスチームのプロジェクトを作成し、 **GPU** を割り当てる

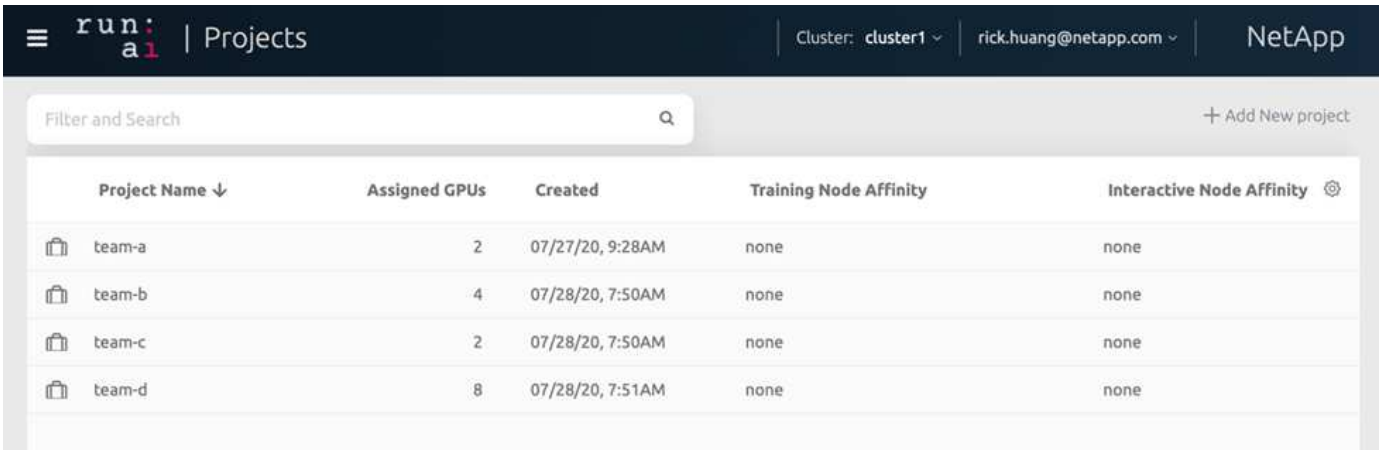
研究者は、 AI CLI や Kubeflow などのプロセスを使ってワークロードを送信できます。リソースの割り当てを合理化して優先順位を設定するために、 Run ： AI では「プロジェクト」という概念が導入されています。プロジェクトは、プロジェクト名を GPU 割り当てと優先設定に関連付けるクォータエンティティです。複数のデータサイエンスチームを管理するシンプルで便利な方法です。

ワークロードを送信する研究者は、プロジェクトをワークロード要求に関連付ける必要があります。Run ： AI スケジューラは、リクエストを現在の割り当てとプロジェクトと比較して、ワークロードにリソースを割り当てることができるかどうか、またはワークロードを保留中の状態にするかどうかを判断します。

システム管理者は、実行： AI プロジェクトタブで次のパラメータを設定できます。

- * モデルプロジェクト。 * ユーザーごとにプロジェクトを設定し、ユーザーのチームごとにプロジェクトを設定し、実際の組織プロジェクトごとにプロジェクトを設定します。
- * プロジェクトの割り当て量。 * 各プロジェクトは、このプロジェクトに同時に割り当てることができる GPU の割り当て量に関連付けられています。これは、このプロジェクトを使用している研究者が、クラスタ内のどの状態であっても、この数の GPU を取得できることが保証されるという意味で、保証されたクォータです。原則として、プロジェクトの割り当ての合計は、クラスタ内の GPU の数と等しくなければなりません。さらに、このプロジェクトのユーザは、過剰割り当てを受け取ることができます。GPU が使用されていない限り、このプロジェクトを使用する研究者は GPU を増やすことができます。に、クォータ超過テストのシナリオと公平性に関する考慮事項を示します "オーバークォータの GPU 割り当てによる高いクラスタ利用率の達成"、 "基本的な資源配分フェアネス"および "オーバークォータフェアネス"。
- 新しいプロジェクトを作成し、既存のプロジェクトを更新して、既存のプロジェクトを削除します。
- * 特定のノードグループで実行するジョブを制限 * 。特定のプロジェクトを割り当てて、特定のノードでのみ実行することができます。これは、プロジェクトチームが十分なメモリを備えた特殊なハードウェアを必要とする場合などに便利です。また、プロジェクトチームは、特別な予算で取得された特定のハードウェアの所有者になる場合もあります。また、パフォーマンスが低いハードウェアで作業したり、長いトレーニングや無人ワークロードを高速ノードに誘導したりするために、ビルドや対話型のワークロードを処理する必要がある場合もあります。ノードをグループ化し、特定のプロジェクトにアフィニティを設定するコマンドについては、を参照してください "「 AI Documentation 」を実行します"。
- * 対話型ジョブの期間を制限します * 。研究者たちは、対話型の仕事を閉じることを忘れがちだ。これにより、リソースの無駄が発生する可能性があります。一部の組織では、対話型ジョブの期間を制限し、自動的に終了することを希望しています。

次の図は、 4 つのチームが作成されたプロジェクトビューを示しています。各チームには、異なるワークロードに対応するために異なる数の GPU が割り当てられます。 GPU の総数は、 2 台の DGX-1 で構成されるクラスタ内の使用可能な GPU の総数と同じです。



実行中のジョブの送信： **AI CLI**

このセクションでは、 Kubernetes ジョブの実行に使用できる基本的な Run ： AI コマンドについて詳しく説明します。ワークロードの種類に応じて 3 つの要素に分割されます。AI / ML / DL のワークロードは、次の 2 つの汎用タイプに分けることができます。

- * 無人トレーニングセッション * 。このようなタイプのワークロードを扱うことで、データサイエンティストは自己実行ワークロードを準備し、実行のためにワークロードを送信します。実行中に、お客様は結果を確認できます。このタイプのワークロードは、多くの場合、本番環境で使用されます。また、モデル開発が段階で行われるため、手動操作は必要ありません。
- * インタラクティブビルドセッション * 。このようなワークロードの場合、データサイエンティストは Bash 、 Jupyter Notebook 、リモート PyCharm などの IDE を使用した対話型セッションを開始し、 GPU リソースに直接アクセスします。次に、対話型のワークロードを実行してポートを接続し、コンテナユーザに内部ポートを表示するシナリオを紹介します。

無人トレーニングのワークロード

プロジェクトをセットアップして GPU を割り当てたら、コマンドラインで次のコマンドを使用して Kubernetes のワークロードを実行できます。

```
$ runai project set team-a runai submit hyper1 -i gcr.io/run-ai-
demo/quickstart -g 1
```

このコマンドは、単一の GPU を割り当てたチーム A の無人トレーニングジョブを開始します。このジョブは、サンプルの Docker イメージである「 gcr.io/run-ai-demo/QuickStart 」に基づいています。私たちはジョブ「 hyper1 」と名付けました。その後、次のコマンドを実行して、ジョブの進捗状況を監視します。

```
$ runai list
```

次の図に 'runai list' コマンドの結果を示します一般的なステータスは次のとおりです。

- 「 ContainerCreating 」を参照してください。Docker コンテナをクラウドリポジトリからダウンロードしています。
- 「保留中」。ジョブはスケジュールされるのを待っています。
- 「ランニング」。ジョブが実行中です。

```
~> runai list
Showing jobs for project team-a
NAME    STATUS   AGE  NODE                                  IMAGE                        TYPE         PROJECT  USER   GPUs
hyper1  Running  11s  gke-dev-yaron1-gpu-4-pool-154f511d-5nk5  gcr.io/run-ai-demo/quickstart  Train        team-a   yaron  1
```

ジョブのステータスをさらに表示するには、次のコマンドを実行します。

```
$ runai get hyper1
```

ジョブのログを表示するには、「 runai logs <job-name> 」コマンドを実行します。

```
$ runai logs hyper1
```

この例では、現在のトレーニングエポック、 ETA 、損失関数値、精度、および各ステップの経過時間を含む、実行中の DL セッションのログを確認する必要があります。

クラスタのステータスは、 Run ： AI UI で確認できます から "https://app.run.ai/"。[ダッシュボード]>[概要] で、 GPU 利用率を監視できます。

このワークロードを停止するには、次のコマンドを実行します。

```
$ runai delte hyper1
```

このコマンドを実行すると、トレーニングワークロードが停止します。このアクションを確認するには 'runai list' をもう一度実行します詳細については、を参照してください "無人トレーニングワークロードの開始"。

ワークロードを対話型に構築

プロジェクトをセットアップして GPU を割り当てたら、コマンドラインで次のコマンドを使用して対話型ビルドワークロードを実行できます。

```
$ runai submit build1 -i python -g 1 --interactive --command sleep --args
infinity
```

このジョブは、サンプルの Docker イメージ Python に基づいています。ジョブ build1 という名前を付けました。

> 「 --interactive 」フラグは、ジョブが開始または終了していないことを意味します研究者の責任で業務を終了してください。管理者は、対話型ジョブがシステムによって終了されるまでの時間制限を定義できます。

--g 1' フラグは ' このジョブに 1 つの GPU を割り当てます与えられたコマンドと引数は、「 --command sleep — args インフィニティ」です。コマンドを指定するか、コンテナを開始してすぐに終了する必要があります。

次のコマンドは、で説明したコマンドと同様に機能します [無人トレーニングのワークロード]：

- runai list` ：名前、ステータス、経過時間、ノード、イメージ、 プロジェクト、ユーザ、 GPU によるジョブの処理
- runai get build1`: ジョブ build1 の追加ステータスを表示します。
- runai delete build1`: 対話型のワークロード build1 を停止しますコンテナに bash シェルを取得するには ' 次のコマンドを実行します

```
$ runai bash build1
```

これにより、コンピュータにシェルが直接挿入されます。データサイエンティストは、コンテナ内でモデルを開発または微調整できます。

クラスタのステータスは、 Run ： AI UI で確認できます から "https://app.run.ai"。詳細については、を参照してください "対話型ビルドワークロードの開始と使用"。

ポートが接続された対話型のワークロード

Run ： AI CLI でコンテナを開始する際に、対話型ビルドワークロードを拡張することで、コンテナユーザに内部ポートを公開できます。これは、クラウド環境、 Jupyter Notebook の操作、その他のマイクロサービスへの接続に役立ちます。 "入力" Kubernetes クラスタの外部から Kubernetes サービスへのアクセスを許可します。アクセスを設定するには、どのインバウンド接続がどのサービスに到達するかを定義する一連のルールを作成します。

クラスタ内のサービスへの外部アクセスを適切に管理するには、クラスタ管理者がインストールすることを推奨します "入力" ロードバランサを設定します。

入力をサービスタイプとして使用するには、次のコマンドを実行して、ワークロード送信時にメソッドタイプとポートを設定します。

```
$ runai submit test-ingress -i jupyter/base-notebook -g 1 \
  --interactive --service-type=ingress --port 8888 \
  --args="--NotebookApp.base_url=test-ingress" --command=start-notebook.sh
```

コンテナが正常に起動したら、「 runai list 」を実行して、 Jupyter Notebook にアクセスする「サービス URL (S) 」を確認します。URL は、入力エンドポイント、ジョブ名、およびポートで構成されます。たとえば、を参照してください https://10.255.174.13/test-ingress-8888。

詳細については、を参照してください "ポートが接続されている対話型のビルドワークロードを起動する"。

高いクラスタ利用率の達成

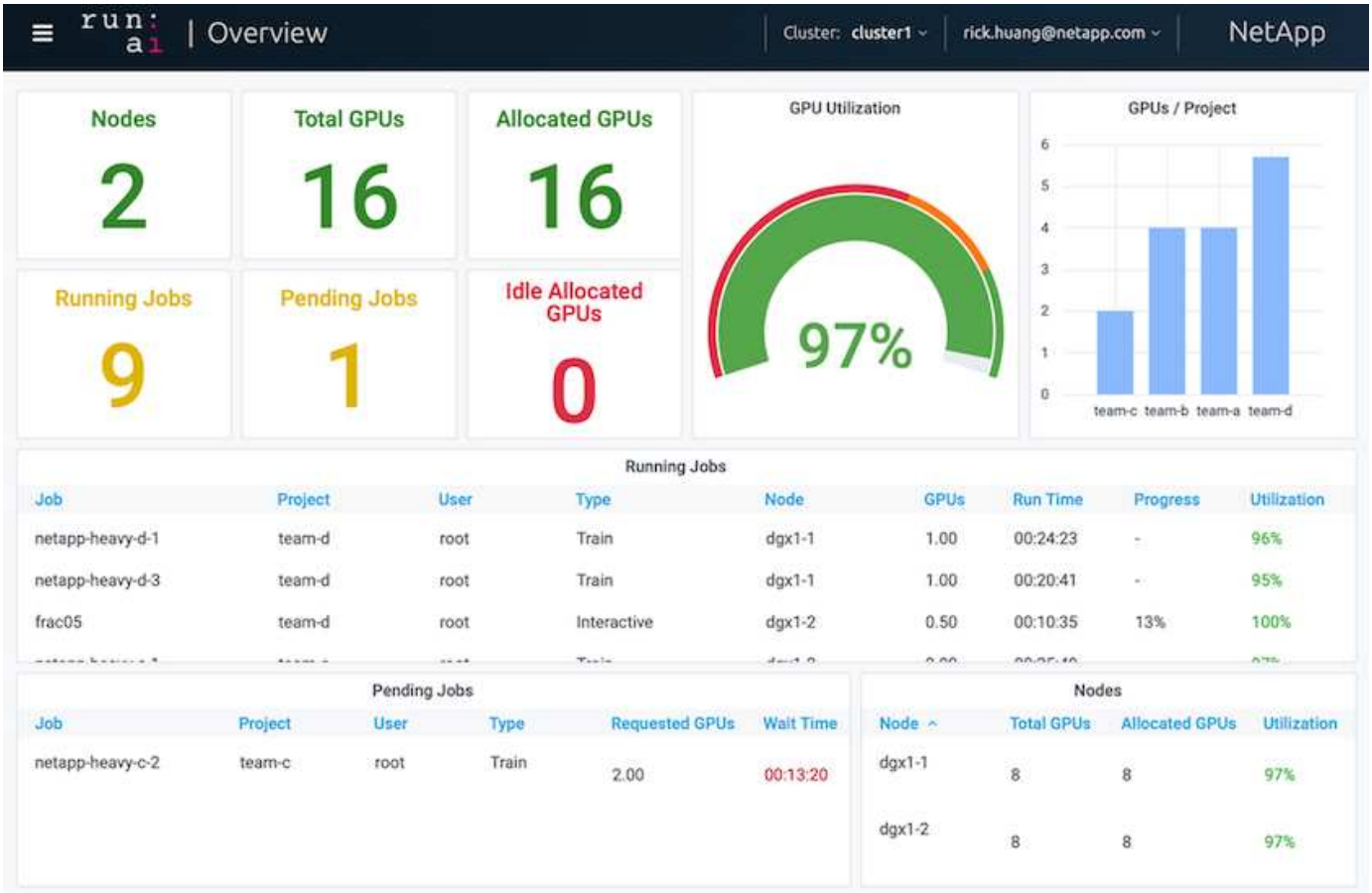
このセクションでは、 4 つのデータサイエンスチームがそれぞれ独自のワークロードを送信して実行： AI オーケストレーション解決策を実証する現実的なシナリオをエミュレ

ートしています。このシナリオでは、 GPU リソースの優先順位付けとバランシングを維持しながら、クラスタの利用率を高めることができますまず、で説明した ResNet-50 ベンチマークを使用します セクション "ResNet-50 と ImageNet データセットベンチマークの概要"：

```
$ runai submit netapp1 -i netapp/tensorflow-tf1-py3:20.01.0 --local-image
--large-shm  -v /mnt:/mnt -v /tmp:/tmp --command python --args
"/netapp/scripts/run.py" --args "--
dataset_dir=/mnt/mount_0/dataset/imagenet/imagenet_original/" --args "--
num_mounts=2"  --args "--dgx_version=dgx1" --args "--num_devices=1" -g 1
```

ResNet-50 ベンチマークを実行しました（を参照） "NVA-1121."。パブリック Docker リポジトリに存在しないコンテナには、フラグ「 --local-image 」を使用しました。ディレクトリ「 /mnt/' 」と「 /tmp' 」をホスト DGX-1 ノード上の「 /mnt/' 」と「 /tmp' 」にそれぞれコンテナにマウントしました。データセットは、ディレクトリを指す「 dataset_dir 」引数を持つ NetApp AFFA800 にあります。どちらの場合も '--num_devices =1' と '-g 1' は ' このジョブに 1 つの GPU を割り当てていることを意味します前者は「 run.py 」スクリプトの引数で、後者は「 runai submit 」コマンドのフラグです。

次の図は、 97% の GPU 利用率を備え、 16 個の使用可能な GPU が割り当てられたシステム概要ダッシュボードを示しています。GPU / プロジェクトの棒グラフでは、各チームに割り当てられている GPU の数を簡単に確認できます。[実行中のジョブ] ウィンドウ枠には、現在実行中のジョブ名、プロジェクト、ユーザー、タイプ、ノード、 GPU の消費、実行時間、進捗状況、利用率の詳細。キューに登録されているワークロードとその待機時間のリストが「保留中のジョブ」に表示されます。さらに、 Nodes ボックスは、クラスタ内の個々の DGX-1 ノードの GPU 番号と利用率を表示します。



開発、ハイパーパラメータチューニング、デバッグのどの段階であっても、研究者や開発者が自分のモデルに取り組んでいる場合、このようなワークロードに必要なコンピューティングリソースは通常少なくなります。したがって、フラクショナル GPU とメモリをプロビジョニングして、同じ GPU を他のワークロードに同時に割り当てることがより効率的です。実行： AI のオーケストレーション解決策は、 Kubernetes 上のコンテナ化されたワークロード向けのフラクショナル GPU 共有システムを提供します。CUDA プログラムを実行するワークロードをサポートするシステムで、推論やモデル構築などの軽量な AI タスクに特に適しています。フラクショナル GPU システムを使用すると、データサイエンスチームや AI エンジニアリングチームは、 1 つの GPU で複数のワークロードを同時に実行できます。これにより、コンピュータビジョン、音声認識、自然言語処理など、より多くのワークロードを同じハードウェア上で実行できるようになり、コストが削減されます。

実行： AI のフラクショナル GPU システムは、仮想化された論理 GPU を独自のメモリとコンピューティングスペースで効率的に作成します。このスペースは、コンテナが自己完結型のプロセッサであるかのように使用およびアクセスできます。これにより、複数のワークロードを同じ GPU 上で並行して実行でき、互いに影響することはありません。解決策は透過的でシンプル、かつ移植可能であり、コンテナ自体に変更を加える必要はありません。

一般的な usecase では、同じ GPU 上で 2 〜 8 個のジョブが実行されていることがわかります。つまり、同じハードウェアで作業を 8 倍行うことができます。

次の図のプロジェクト「 team -d 」に属するジョブ「分割 05 」については、割り当てられた GPU の数が 0.50 であることがわかります。さらに 'nvidia-smi コマンドによって確認されますこのコマンドは ' コンテナに使用できる GPU メモリが 'DGX-1 ノードの V100 GPU あたりの 32GB の半分である 16,255MB であることを示します

```
root@run-deploy:~# runai bash frac05 -p team-d
root@frac05-0:/workload# nvidia-smi
Tue Jul 28 15:17:03 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 450.51.05    Driver Version: 450.51.05    CUDA Version: 11.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  On   | 00000000:07:00.0 Off |                    0 |
| N/A   57C    P0   240W / 300W |  15525MiB / 16255MiB |     100%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A       156      C   python3                         15525MiB |
+-----------------------------------------------------------------------------+
```

オーバークォータの **GPU** 割り当てによる高いクラスタ利用率の達成

を参照してください "基本的な資源配分フェアネス"および "オーバークォータフェアネス"また、高度なテストシナリオを考案し、複雑なワークロード管理、自動プリエンプティブスケジューリング、オーバークォータ GPU プロビジョニングを実現する Run ： AI オーケストレーション機能をデモしました。これにより、 ONTAP AI 環境でクラスタリソースの利用率を高め、エンタープライズレベルのデータサイエンスチームの生産性を最適化できました。

これらの 3 つのセクションで、次のプロジェクトとクォータを設定します。

プロジェクト	クォータ
チーム A	4.
チーム - b	2.
チーム -c	2.
チーム -d	8.

また、この 3 つのセクションでは、次のコンテナを使用します。

- Jupyter Notebook ： "jupyter/base-notebook
- 「 GCR.IO/run-ai-demo/QuickStart 」を実行します

このテストシナリオでは、次の目標を設定します。

- リソースのプロビジョニングの簡易性と、リソースの使用方法を説明します ユーザから抽象化されます
- ユーザが GPU のフラクションを簡単にプロビジョニングする方法を説明します GPU の数を整数で指定します
- チームを編成してコンピューティングのボトルネックを解消する方法を説明します リソースクォータがある場合は、ユーザがそのリソースクォータを確認します クラスタ内に GPU が搭載されている
- ネットアップコンテナなどの大量の計算処理を行うジョブを実行する際に、 NetApp 解決策を使用してデータパイプラインのボトルネックを解消する方法を説明する
- を使用して複数のタイプのコンテナを実行する方法を説明します システム
  - Jupyter ノートブック
  - 実行： AI コンテナ
- クラスタがフルの状態のときに高い利用率を表示します

テスト中に実行される実際のコマンドシーケンスの詳細については、を参照してください "セクション 4.8 のテストの詳細"。

13 個のワークロードすべてが送信されると、次の図に示すように、コンテナ名と割り当てられている GPU のリストが表示されます。7 つのトレーニングと 6 つの対話型ジョブが用意されており、 4 つのデータサイエンスチームをシミュレーションして、それぞれに独自のモデルを実行しているか開発中であるかを確認していますインタラクティブなジョブの場合、個々の開発者は Jupyter Notebook を使用してコードの書き込みやデバッグを行っています。そのため、クラスタリソースをあまり使用せずに GPU フラクションをプロビジョニングすることをお勧めします。

```
root@run-deploy:~# runai list -A
NAME           STATUS   AGE  NODE    IMAGE                          TYPE        PROJECT  USER  GPUs  CREATED BY CLI  SERVICE URL(S)
b-4-gg         Running  2m   dgx1-2  gcr.io/run-ai-demo/quickstart  Train       team-b   root  2     true
c-5-g          Running  2m   dgx1-2  gcr.io/run-ai-demo/quickstart  Train       team-c   root  1     true
c-4-gg         Running  2m   dgx1-1  gcr.io/run-ai-demo/quickstart  Train       team-c   root  2     true
b-3-g          Running  2m   dgx1-1  gcr.io/run-ai-demo/quickstart  Train       team-b   root  1     true
c-3-g02        Running  2m   dgx1-1  gcr.io/run-ai-demo/quickstart  Interactive team-c   root  0.2   true
d-1-gggg       Running  2m   dgx1-2  gcr.io/run-ai-demo/quickstart  Train       team-d   root  4     true
c-2-g03        Running  2m   dgx1-1  gcr.io/run-ai-demo/quickstart  Interactive team-c   root  0.3   true
c-1-g05        Running  2m   dgx1-1  gcr.io/run-ai-demo/quickstart  Interactive team-c   root  0.5   true
a-2-gg         Running  3m   dgx1-1  gcr.io/run-ai-demo/quickstart  Train       team-a   root  2     true
b-2-g04        Running  3m   dgx1-2  gcr.io/run-ai-demo/quickstart  Interactive team-b   root  0.4   true
a-1-g          Running  3m   dgx1-1  gcr.io/run-ai-demo/quickstart  Train       team-a   root  1     true
b-1-g06        Running  3m   dgx1-2  gcr.io/run-ai-demo/quickstart  Interactive team-b   root  0.6   true
a-1-1-jupyter  Running  3m   dgx1-1  jupyter/base-notebook          Interactive team-a   root  1     true            http://10.61.218.134/a-1-1-jupyter,
https://10.61.218.134/a-1-1-jupyter
```

このテストシナリオの結果は次のようになります。

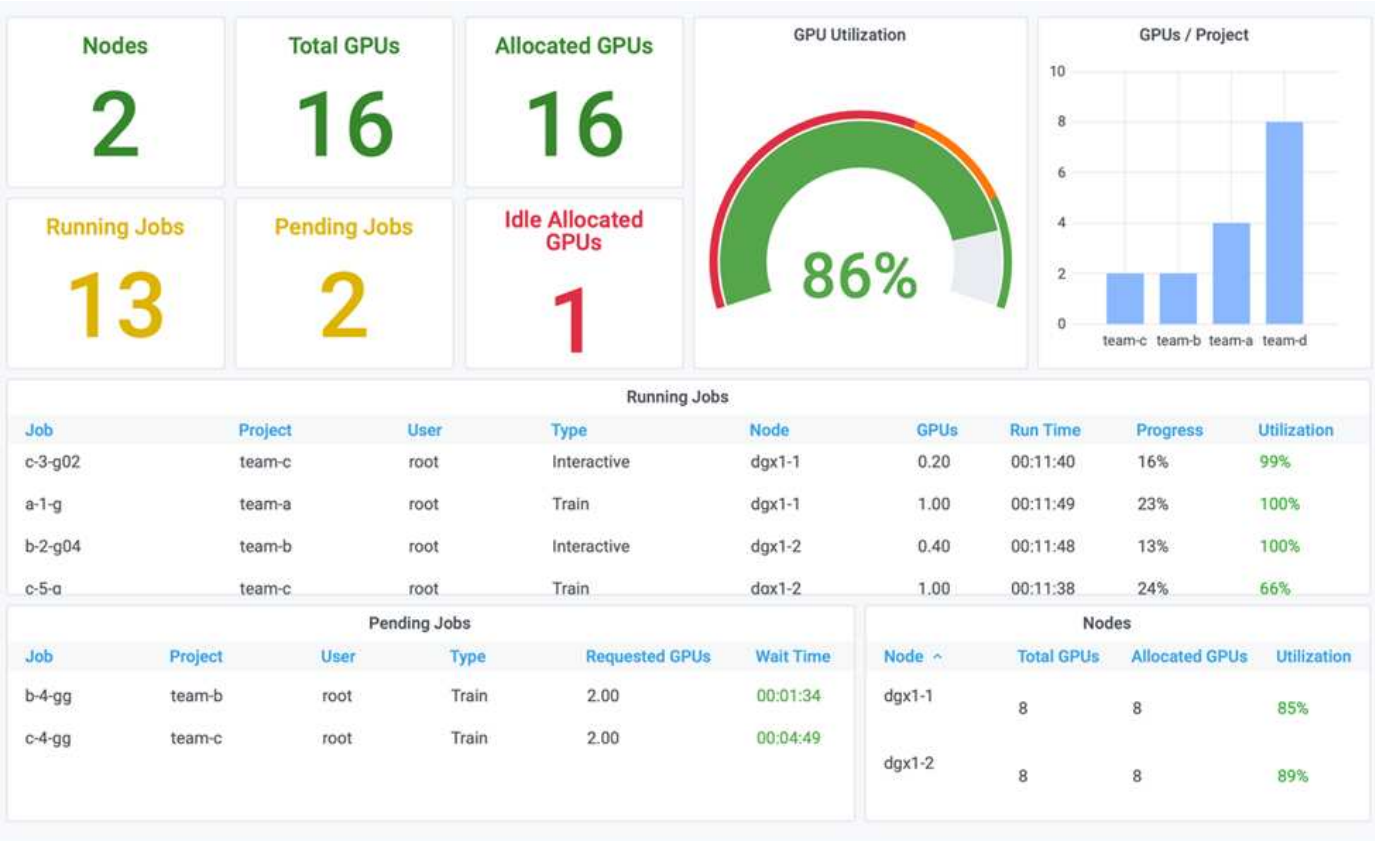
- クラスタがいっぱいである必要があります。 16 基の GPU が使用されています。
- クラスタの利用率が高い。
- フラクショナル割り当てにより、 GPU よりも多くの実験が行われています。
- 「 team -d 」では、すべてのクォータが使用されているわけではありません。したがって、「 team -b 」と「 team -c 」では、実験に追加の GPU を使用することができるため、イノベーションにかかる時間が短縮されます。

基本的な資源配分フェアネス

このセクションでは 'team -d がより多くの GPU を要求した場合 (それらは割り当ての下にあります) ' システムは 'team -b' および 'team -c のワークロードを一時停止し ' 公正な共有方法で保留状態に移行することを示しています

ジョブの送信、使用するコンテナイメージ、実行するコマンドシーケンスなどの詳細については、を参照してください "セクション 4.9 のテストの詳細"。

次の図は、自動ロードバランシングとプリエンプティブスケジューリングにより、クラスタ使用率、チームごとに割り当てられた GPU 、保留中のジョブを示しています。すべてのチームワークロードが要求した GPU の総数が、クラスタ内で使用可能な合計 GPU 数を超えると、実行： AI の内部公正性アルゴリズムによって、「 team -b 」と「 team -c 」のそれぞれに 1 つのジョブが一時停止されます。これは、それらがプロジェクトの割り当て量を満たしているためです。これにより、クラスタ全体の利用率が向上しますが、データサイエンスチームは、管理者が設定したリソースの制約に基づいて引き続き作業できます。



このテストシナリオの結果は、次のことを示しています。

- * 自動ロードバランシング。 * システムは、各チームが割り当てを使用するように GPU の割り当て量を自動的に調整します。一時停止されていたワークロードは、そのクォータを超えていたチームに属しています。
- * 公正な共有の一時停止。 * システムは、ノルマを達成したチームのワークロードを停止してから、もう一方のチームのワークロードを停止します。実行： AI には内部的な公正性アルゴリズムがあります。

オーバークォータフェアネス

このセクションでは、複数のチームがワークロードを送信し、クォータを超過するシナリオを拡張します。この方法では、 Run ： AI の公正性アルゴリズムが、事前設定されたクォータの比率に従ってクラスタリソースを割り当てる方法を説明します。

このテストシナリオの目標：

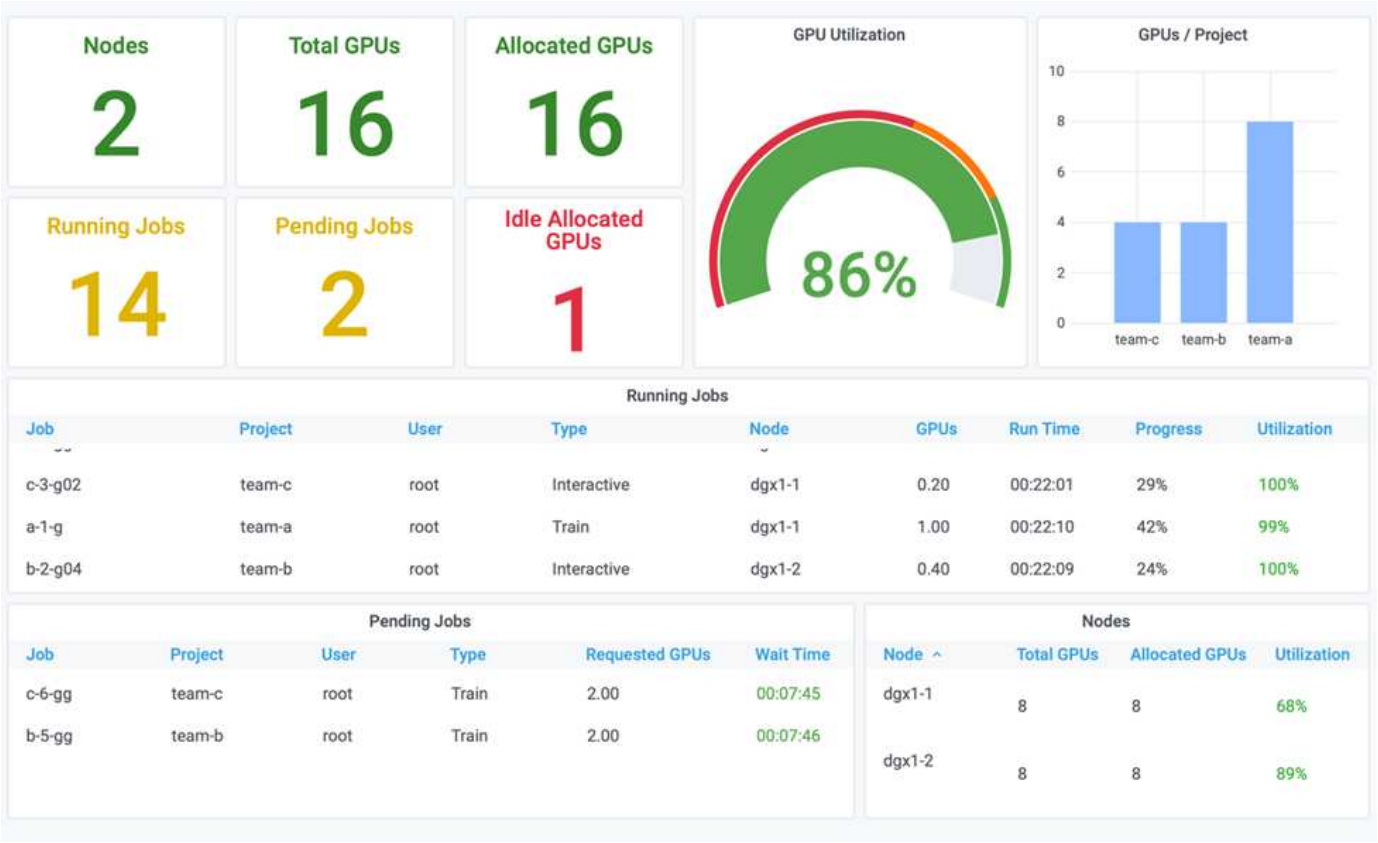
- 複数のチームがクォータを介して GPU を要求しているときのキューイングメカニズムを示します。

- システムが、クォータの比率に従って、クォータを超過した複数のチーム間にクラスタの適正な共有を分散し、クォータが大きいチームがスペア容量の大部分を占めるようにする方法を示します。

の末尾 "基本的な資源配分フェアネス"では、キューに入れられるワークロードは 2 つあります。 1 つは「 team -b 」用、もう 1 つは「 team -c 」用です。このセクションでは、追加のワークロードをキューに登録します。

ジョブの送信、使用されるコンテナイメージ、実行されるコマンドシーケンスなどの詳細については、を参照してください "セクション 4.10 のテストの詳細"。

すべてのジョブがセクションに従って送信される場合 "セクション 4.10 のテストの詳細"システム・ダッシュボードには 'team a'team -b'および 'team -c` のすべての GPU の数が ' 事前設定されたクォータよりも多くなっていることが示されています「 team -a 」は、初期設定のソフトクォータ（ 4 ）よりも 4 基の GPU を占有し、「 team -b 」と「 team -c 」はソフトクォータ（ 2 つ）よりも 2 つの GPU を占有します。割り当てられたクォータ超過 GPU の比率は、事前設定されたクォータの比率と同じです。これは、システムが優先順位の基準として事前設定されたクォータを使用し、複数のチームがクォータを超えて GPU を追加するように要求した場合に応じてプロビジョニングされるためです。このような自動ロードバランシングは、企業のデータサイエンスチームが AI モデルの開発と運用に積極的に関与している場合に、公平性と優先順位付けを提供します。



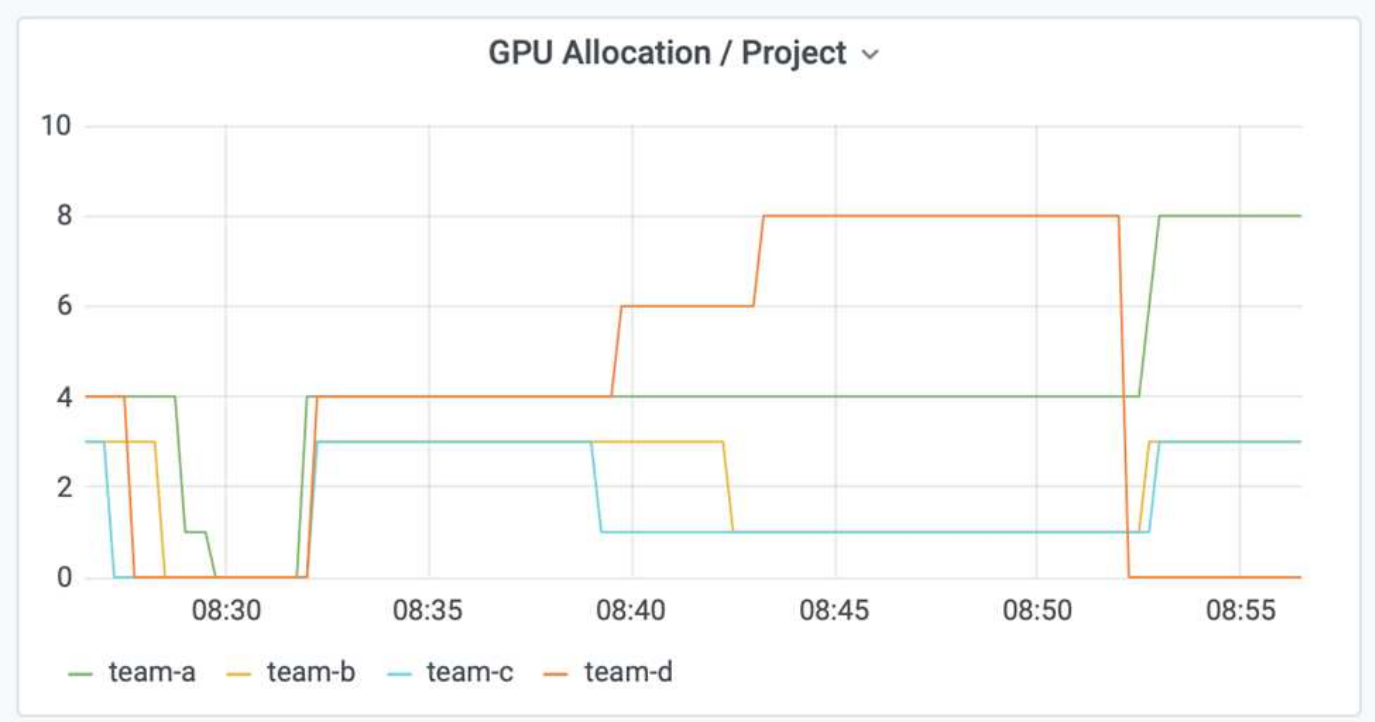
このテストシナリオの結果は次のようになります。

- 他のチームのワークロードのキュー解除が開始されます。
- キュー解除の順序は ' 公正性アルゴリズムに従って決定されますたとえば 'team -b と 'team -c は ' 同等のクォータを持つため ' 同じ量のオーバークォータ GPU を取得します また 'team -a は 'team -b と 'team -c のクォータの 2 倍のクォータを備えているため 'GPU の量が 2 倍になります
- すべての割り当てが自動的に行われます。

したがって、システムは次の状態で安定します。

プロジェクト	**GPU** が割り当てられました	コメント（ **Comment** ）
チーム A	8/4	クォータを介した 4 基の GPU空のキューです。
チーム - b	4 月 2 日	クォータを介した 2 つの GPU 。1 つのワークロードがキューに登録
チーム -c	4 月 2 日	クォータを介した 2 つの GPU 。1 つのワークロードがキューに登録
チーム -d	0/8	GPU をまったく使用しないので、キューに登録されているワークロードはありません

次の図は、プロジェクトごとの GPU 割り当てを示しています Run ：セクションの AI Analytics ダッシュボードに表示される時間 "オーバークォータの GPU 割り当てによる高いクラスタ利用率の達成"、 "基本的な資源配分フェアネス"および "オーバークォータフェアネス"。図の各行は、特定のデータサイエンスチーム用にプロビジョニングされた GPU の数を常に表しています。システムは、送信されたワークロードに応じて GPU を動的に割り当てることがわかります。これにより、クラスタ内に使用可能な GPU がある場合はクォータを超過し、公平性に従ってジョブをプリエンプトしてから、 4 つのチームすべてが最終的に安定した状態に到達することができます。



Trident でプロビジョニングされた永続的ボリュームにデータを保存する

NetApp Trident は、コンテナ化されたアプリケーションが求める高度な永続性のニーズに対応できるように設計された、完全にサポートされているオープンソースプロジェクトです。Trident でプロビジョニングされた Kubernetes PersistentVolume （ PV ）に対してデータの読み取りと書き込みを行うことができ、データ階層化、暗号化、 NetApp Snapshot テクノロジ、コンプライアンス、 NetApp ONTAP データ管理ソフトウェアが

提供する高パフォーマンスといったメリットも活用できます。

既存の名前空間での **PVC** の再利用

大規模な AI プロジェクトでは、異なるコンテナで同じ Kubernetes PV に対してデータの読み取りや書き込みを行う方が効率的な場合があります。Kubernetes Persistent Volume Claim （ PVC ；永続ボリューム要求）を再利用するには、ユーザが PVC を作成しておく必要があります。を参照してください "NetApp Trident のドキュメント" PVC 作成の詳細については、を参照してください。次に、既存の PVC を再利用する例を示します。

```
$ runai submit pvc-test -p team-a --pvc test:/tmp/pvc1mount -i gcr.io/run-
ai-demo/quickstart -g 1
```

次のコマンドを実行して ' プロジェクト 'team -a' のジョブ pvc-test' のステータスを表示します

```
$ runai get pvc-test -p team-a
```

'team -a' ジョブ 'pvctest' にマウントされた PV /tmp/pvc1mount が表示されますこのようにして、複数のコンテナが同じボリュームから読み取ることができるため、開発中または本番環境で競合する複数のモデルが存在する場合に便利です。データサイエンティストは、モデルのアンサンブルを構築し、大多数の投票またはその他の技術によって予測結果を組み合わせることができます。

次のコマンドを使用してコンテナシェルにアクセスします。

```
$ runai bash pvc-test -p team-a
```

その後、マウントされたボリュームを確認し、コンテナ内のデータにアクセスできます。

PVC の再利用というこの機能は、 NetApp FlexVol ボリュームと NetApp ONTAP FlexGroup ボリュームと連携します。そのため、データエンジニアは、より柔軟で堅牢なデータ管理オプションを利用して、ネットアップのデータファブリックを活用できます。

まとめ

ネットアップと Run ：このテクニカルレポートでは、 AI と、 NetApp ONTAP AI 解決策の独自機能と、 Run ： AI プラットフォームを組み合わせることで、 AI ワークロードのオーケストレーションを簡易化できることを紹介しています。前の手順では、ディープラーニングのためのデータパイプラインとワークロードオーケストレーションのプロセスを合理化するリファレンスアーキテクチャを示します。これらのソリューションの導入を検討しているお客様には、ネットアップと Run ： AI の活用で詳細を確認することをお勧めします。

セクション **4.8** のテストの詳細

ここでは、のテストの詳細について説明します "オーバークォータの GPU 割り当てによ

る高いクラスタ利用率の達成"。

次の順序でジョブを送信します。

プロジェクト	イメージ（ **Image** ）	**GPU** の数	合計	コメント（ **Comment** ）
チーム A	Jupyter	1.	1/4	–
チーム A	ネットアップ	1.	2/4	–
チーム A	実行： AI	2.	4 月 4 日	すべてのクォータを使用しています
チーム - b	実行： AI	0.6	0.6/2	フラクショナル GPU
チーム - b	実行： AI	0.4	1/2	フラクショナル GPU
チーム - b	ネットアップ	1.	2/2.	–
チーム - b	ネットアップ	2.	4 月 2 日	クォータに 2 つ
チーム -c	実行： AI	0.5	0.5/2	フラクショナル GPU
チーム -c	実行： AI	0.3	0.8/2.	フラクショナル GPU
チーム -c	実行： AI	0.2	1/2	フラクショナル GPU
チーム -c	ネットアップ	2.	3/2	クォータに 1 つ
チーム -c	ネットアップ	1.	4 月 2 日	クォータに 2 つ
チーム -d	ネットアップ	4.	4/8	クォータの半分を使用します

コマンドの構造：

```
$ runai submit <job-name> -p <project-name> -g <#GPUs> -i <image-name>
```

テストで使用する実際のコマンドシーケンス：

```
$ runai submit a-1-1-jupyter -i jupyter/base-notebook -g 1 \
  --interactive --service-type=ingress --port 8888 \
  --args="--NotebookApp.base_url=team-a-test-ingress" --command=start
-notebook.sh -p team-a
$ runai submit a-1-g -i gcr.io/run-ai-demo/quickstart -g 1 -p team-a
$ runai submit a-2-gg -i gcr.io/run-ai-demo/quickstart -g 2 -p team-a
$ runai submit b-1-g06 -i gcr.io/run-ai-demo/quickstart -g 0.6
--interactive -p team-b
$ runai submit b-2-g04 -i gcr.io/run-ai-demo/quickstart -g 0.4
--interactive -p team-b
$ runai submit b-3-g -i gcr.io/run-ai-demo/quickstart -g 1 -p team-b
$ runai submit b-4-gg -i gcr.io/run-ai-demo/quickstart -g 2 -p team-b
$ runai submit c-1-g05 -i gcr.io/run-ai-demo/quickstart -g 0.5
--interactive -p team-c
$ runai submit c-2-g03 -i gcr.io/run-ai-demo/quickstart -g 0.3
--interactive -p team-c
$ runai submit c-3-g02 -i gcr.io/run-ai-demo/quickstart -g 0.2
--interactive -p team-c
$ runai submit c-4-gg -i gcr.io/run-ai-demo/quickstart -g 2 -p team-c
$ runai submit c-5-g -i gcr.io/run-ai-demo/quickstart -g 1 -p team-c
$ runai submit d-1-gggg -i gcr.io/run-ai-demo/quickstart -g 4 -p team-d
```

この時点で、次の状態になります。

プロジェクト	**GPU** が割り当てられました	ワークロードキュー
チーム A	4/4 （ソフトクォータ / 実際の割り当て）	なし
チーム - b	4 月 2 日	なし
チーム -c	4 月 2 日	なし
チーム -d	4/8	なし

を参照してください "「 Achieving High Cluster Utilization With over-uota 」 GPU 割り当て" 進行中のテストシナリオについてのディスカッションにご参加ください。

セクション **4.9** のテストの詳細

ここでは、のテストについて詳しく説明します "基本的な資源配分フェアネス"。

次の順序でジョブを送信します。

プロジェクト	**GPU** の数	合計	コメント（ **Comment** ）
チーム -d	2.	6 月 8 日	team -b/c ワークロードが一時停止し、「 pending 」に移動します。

プロジェクト	GPU の数	合計	コメント（ Comment ）
チーム -d	2.	8 月 8 日	他のチーム (b/c) のワークロードは一時停止し、「保留中」に移動します。

実行される次のコマンドシーケンスを参照してください。

```
$ runai submit d-2-gg -i gcr.io/run-ai-demo/quickstart -g 2 -p team-d$
runai submit d-3-gg -i gcr.io/run-ai-demo/quickstart -g 2 -p team-d
```

この時点で、次の状態になります。

プロジェクト	GPU が割り当てられました	ワークロードキュー
チーム A	4 月 4 日	なし
チーム - b	2/2.	なし
チーム -c	2/2.	なし
チーム -d	8 月 8 日	なし

を参照してください "基本的な資源配分フェアネス" を参照してください。

セクション **4.10** のテストの詳細

ここでは、のテストについて詳しく説明します "オーバークォータフェアネス"。

「 team -a 」、「 team -b 」、「 team -c 」の順にジョブを送信します。

プロジェクト	GPU の数	合計	コメント（ Comment ）
チーム A	2.	4 月 4 日	1 個のワークロードをキューに登録
チーム A	2.	4 月 4 日	2 個のワークロードがキューに登録
チーム - b	2.	2/2.	2 個のワークロードがキューに登録
チーム -c	2.	2/2.	2 個のワークロードがキューに登録

実行される次のコマンドシーケンスを参照してください。

```
$ runai submit a-3-gg -i gcr.io/run-ai-demo/quickstart -g 2 -p team-a$
runai submit a-4-gg -i gcr.io/run-ai-demo/quickstart -g 2 -p team-a$ runai
submit b-5-gg -i gcr.io/run-ai-demo/quickstart -g 2 -p team-b$ runai
submit c-6-gg -i gcr.io/run-ai-demo/quickstart -g 2 -p team-c
```

この時点で、次の状態になります。

プロジェクト	**GPU** が割り当てられました	ワークロードキュー
チーム A	4 月 4 日	2 つのワークロードがそれぞれ 2 つの GPU を必要とします
チーム - b	2/2.	2 つのワークロードがそれぞれ 2 つの GPU を必要とします
チーム -c	2/2.	2 つのワークロードがそれぞれ 2 つの GPU を必要とします
チーム -d	8 月 8 日	なし

次に 'team -d のすべてのワークロードを削除します

```
$ runai delete -p team-d d-1-gggg d-2-gg d-3-gg
```

を参照してください "オーバークォータフェアネス"を参照してください。

追加情報の検索場所

このドキュメントに記載されている情報の詳細については、次のリソースを参照してください。

- NVIDIA DGX システム
  - NVIDIA DGX-1 システムhttps://www.nvidia.com/en-us/data-center/dgx-1/[]
  - NVIDIA V100 Tensor コア GPUhttps://www.nvidia.com/en-us/data-center/tesla-v100/[]
  - NVIDIA NGChttps://www.nvidia.com/en-us/gpu-cloud/[]
- 実行： AI コンテナオーケストレーション解決策
  - 実行： AI 製品の概要https://docs.run.ai/home/components/[]
  - 実行： AI インストールドキュメントhttps://docs.run.ai/Administrator/Cluster-Setup/Installing-Run-AI-on-an-on-premise-Kubernetes-Cluster/[] https://docs.run.ai/Administrator/Researcher-Setup/Installing-the-Run-AI-Command-Line-Interface/
  - 実行時のジョブの送信： AI CLIhttps://docs.run.ai/Researcher/Walkthroughs/Walkthrough-Launch-Unattended-Training-Workloads-/[] https://docs.run.ai/Researcher/Walkthroughs/Walkthrough-Start-and-Use-Interactive-Build-Workloads-/
  - 実行時の GPU フラクションの割り当て： AI CLIhttps://docs.run.ai/Researcher/Walkthroughs/Walkthrough-Using-GPU-Fractions/[]
- NetApp AI コントロールプレーン
  - テクニカルレポートをご参照くださいhttps://www.netapp.com/us/media/tr-4798.pdf[]
  - 簡単なデモhttps://youtu.be/gfr_sO27Rvo[]
  - GitHub リポジトリhttps://github.com/NetApp/kubeflow_jupyter_pipeline[]
- NetApp AFF システム

- NetApp AFF A シリーズのデータシートhttps://www.netapp.com/us/media/ds-3582.pdf[]
- ネットアップの All Flash FAS 向けフラッシュソリューションの利点https://www.netapp.com/us/media/ds-3733.pdf[]
- ONTAP 9 情報ライブラリhttp://mysupport.netapp.com/documentation/productlibrary/index.html?productID=62286[]
- NetApp ONTAP FlexGroup Volume テクニカルレポートhttps://www.netapp.com/us/media/tr-4557.pdf[]

- NetApp ONTAP AI
  - DGX-1 と Cisco Networking Design Guide による ONTAP AIhttps://www.netapp.com/us/media/nva-1121-design.pdf[]
  - DGX-1 と Cisco Networking Deployment Guide を使用した ONTAP AIhttps://www.netapp.com/us/media/nva-1121-deploy.pdf[]
  - DGX-1 と Mellanox のネットワーキング設計ガイドで構成される ONTAP AIhttp://www.netapp.com/us/media/nva-1138-design.pdf[]
  - DGX-2 を使用した ONTAP AI 設計ガイドhttps://www.netapp.com/us/media/nva-1135-design.pdf[]

TR-4799 -設計：自動運転ワークロード向けのNetApp ONTAP AIリファレンスアーキテクチャ

ネットアップDavid ArnetteとSung-Han Lin

NVIDIA DGXファミリーのシステムは、エンタープライズAIに特化して設計された、世界初の統合人工知能（AI）プラットフォームです。NetApp AFF ストレージシステムは、卓越したパフォーマンスと業界をリードするハイブリッドクラウドデータ管理機能を提供します。ネットアップとNVIDIAは提携を通じてNetApp ONTAP AIリファレンスアーキテクチャを構築し、お客様にAIと機械学習（ML）のワークロードをサポートし、エンタープライズクラスのパフォーマンス、信頼性、サポートを提供するターンキー解決策 を提供しています。

"TR-4799 -設計：自動運転ワークロード向けのNetApp ONTAP AIリファレンスアーキテクチャ"

TR-4211：『NetApp ONTAP AI reference architecture for healthcare：Diagnostic imaging -解決策 design』

NVIDIA、NetApp Jacci Cenci、Sathish Thyagarajan、Rick Huang、Sung-Han Lin

このリファレンスアーキテクチャでは、ヘルスケアユースケース向けにNVIDIA DGX-2システムとNetApp AFF ストレージを使用して人工知能（AI）インフラを構築するお客様のガイドラインを紹介します。また、医療診断画像、検証済みのテストケース、結果のディープラーニング（DL）モデルの開発に使用される高レベルのワークフローに関する情報も含まれています。また、お客様の導入に推奨されるサイジングについても説明します。

"TR-4211：『NetApp ONTAP AI reference architecture for healthcare：Diagnostic imaging -解決策 design』"

TR-4807：『NetApp ONTAP AI reference architecture for Financial Services Workloads -解決策 design』

Karthikeyan Nagalingam、Sung-Han Lin、NetApp Jacci Cenci、NVIDIA

このリファレンスアーキテクチャでは、金融セクターのユースケース向けにNVIDIA DGX-1システムとNetApp AFF ストレージを使用して人工知能インフラを構築するお客様向けのガイドラインを提供します。また、金融サービスのテストケースや結果のディープラーニングモデルの開発に使用されるワークフローの概要についても説明します。また、お客様の導入に推奨されるサイジングについても説明します。

"TR-4807：『NetApp ONTAP AI reference architecture for Financial Services Workloads -解決策 design』"

AIとNetAppの価値を生成

著者: Sathish Thyagarajan, NetApp

概要

生成型人工知能（AI）の需要は、さまざまな業界の混乱を引き起こし、ビジネスの創造性と製品のイノベーションを強化しています。多くの組織が、ジェネレーティブAIを使用して、新しい製品機能の構築、エンジニアリングの生産性の向上、より優れた結果と消費者体験を提供するAIベースのアプリケーションのプロトタイプ作成を行っています。Generative Pre-Trained Transformers（GPT）などのジェネレーティブAIは、ニューラルネットワークを使用して、テキスト、オーディオ、ビデオなど多様な新しいコンテンツを作成します。大規模な言語モデル（LLM）に関連する膨大な規模のデータセットを考えると、オンプレミス、ハイブリッド、マルチクラウドの導入オプションにある魅力的なデータストレージ機能を活用し、データモビリティに関連するリスクを軽減する堅牢なAIインフラを設計することが重要です。 企業がAIソリューションを設計する前に、データ保護とガバナンスを強化しましょう。このホワイトペーパーでは、これらの考慮事項と、生成型AIモデルのトレーニング、再トレーニング、微調整、推論のためのAIデータパイプライン全体でのシームレスなデータ管理とデータ移動を可能にする、対応するNetApp®AIの機能について説明します。

エグゼクティブサマリー

最近では、2022年11月にGPT-3のスピンオフであるChatGPTがリリースされた後、テキスト、コード、画像、さらには治療用タンパク質をユーザーのプロンプトに応答して生成するために使用される新しいAIツールが大きな名声を得ています。これは、ユーザーが自然言語を使用して要求を行うことができることを示しています。AIは、ユーザーの要求を反映したニュース記事や製品説明などのテキストを解釈して生成したり、既存のデータでトレーニングされたアルゴリズムを使用してコード、音楽、スピーチ、視覚効果、3Dアセットを生成したりします。その結果、AIシステムの設計には、「安定した拡散」、「幻覚」、「迅速なエンジニアリング」、「価値の整合」などのフレーズが急速に登場しています。これらの自己管理型または半管理型機械学習（ML）モデルは、クラウドサービスプロバイダや他のAIベンダーを通じて、事前トレーニング済みの基礎モデル（FM）として広く利用されるようになり、さまざまな業界のさまざまな事業所で、下流の自然言語処理（NLP）タスクに採用されています。マッキンゼーのような調査アナリスト企業は、「ジェネレーティブAIが生産性に与える影響は、世界経済に数兆ドルの価値をもたらす可能性がある」と主張しています。 企業はAIを人間の思考パートナーとして再定義しつつあり、FMSは、企業や機関がジェネレーティブAIでできることを同時に拡大していますが、膨大な量のデータを管理する機会は今後も増え続けるでしょう。このドキュメントでは、オンプレミス環境とハイブリッド環境またはマルチクラウド環境の両方で、NetAppのお客様に価値をもたらすNetApp機能に関連した、生成型AIの概要と設計の概念について説明します。

では、お客様が**AI**環境で**NetApp**を使用するための**IT**には何が含まれていますか。 NetAppは、データとクラウドの急速な増加、マルチクラウド管理、AIなどの次世代テクノロジの採用によって生じる複雑さに対応するの

に役立ちます。NetAppは、さまざまな機能をインテリジェントなデータ管理ソフトウェアとストレージインフラに統合し、AIワークロード向けに最適化されたハイパフォーマンスとのバランスを良好に保ちました。LLMのような生成型AIソリューションでは、インテリジェンスを強化するために、ストレージからソースデータセットを何度も読み取り、処理してメモリに格納する必要があります。NetAppは、エッジからコア、クラウドまでのエコシステム全体で、データモビリティ、データガバナンス、データセキュリティのテクノロジ分野で業界をリードしており、大規模なAIソリューションを構築する大企業のお客様にサービスを提供しています。NetAppには強力なパートナーネットワークがあり、最高データ責任者、AIエンジニア、エンタープライズアーキテクト、データサイエンティストが自由に流れるデータパイプラインを設計し、データの前処理、データ保護、 AIモデルのトレーニングと推論に関する戦略的データ管理責任を負い、AI / MLライフサイクルのパフォーマンスと拡張性を最適化します。NetAppのデータテクノロジと機能には、ディープラーニングデータパイプライン向けのNetApp®ONTAP AI®、ストレージエンドポイント間でシームレスかつ効率的にデータを転送するNetApp®SnapMirror®などがあります。 また、リアルタイムレンダリングのためのNetApp®FlexCache®データフローがバッチからリアルタイムに移行し、データエンジニアリングが迅速に行われる場合、リアルタイムジェネレーティブAIモデルの導入に価値をもたらします。あらゆるタイプの企業が新しいAIツールを導入する際、エッジからデータセンター、クラウドまで、拡張性と責任性に優れた説明可能なAIソリューションが求められるデータの課題に直面します。ハイブリッドクラウドとマルチクラウドにおけるデータ管理のオーソリティであるNetAppは、パートナーと共同ソリューションのネットワークの構築に取り組んでいます。このネットワークは、生成AIモデルのトレーニング（プレトレーニング）、微調整、コンテキストベースの推論、LLMのモデル崩壊監視のためのデータパイプラインとデータレイクの構築のあらゆる側面を支援します。

ジェネレーティブ**AI**とは

ジェネレーティブAIは、コンテンツの作成方法、新しいデザインコンセプトの生成方法、新しい構成の探索方法を変えています。ここでは、Generative Adversarial Network（GAN）、Variational Autoencoders（VAE）、Generative Pre-Trained Transformers（GPT）などのニューラルネットワークフレームワークについて説明します。これらのフレームワークは、テキスト、コード、画像、オーディオ、ビデオ、 合成データもありますOpenAIのChat-GPT、GoogleのBard、Hugging Face’s Bloom、Meta’s llamaなどのトランスベースモデルは、大規模な言語モデルの多くの進歩を支える基盤技術として登場しています。同様に、OpenAIのdall-E、MetaのCM3leon、GoogleのImagenは、テキストと画像のセマンティクスをリンクするデータセット拡張とテキストと画像の合成を使用して、新しい複雑な画像をゼロから作成したり、既存の画像を編集して高品質のコンテキスト認識画像を生成する、かつてないほどのフォトリアリズムを提供するテキスト間拡散モデルの例です。デジタルアーティストは、静的な2D画像を没入型の3Dシーンに変換するために、Nerf（Neural Radiance Field）などのレンダリング技術をジェネレーティブAIと組み合わせて適用し始めています。一般に、LLMは、(1)モデルのサイズ(通常は数十億個のパラメータ)、(2)トレーニングデータセットのサイズ、(3)トレーニングコスト、(4)トレーニング後のモデルパフォーマンスの4つのパラメータによって大まかに特徴付けられます。LLMは、主に3つの変圧器アーキテクチャに分類されます。（i）エンコーダのみの機種例：BERT（Google、2018）、（ii）エンコーダ-デコーダ例：BART（Meta、2020）、（iii）デコーダのみのモデル。例：llama（Meta、2023）、palm-E（Google、2023）。ビジネス要件に応じて、モデルパラメータの数（N）とトレーニングデータセット内のトークンの数（D）を選択するアーキテクチャに関係なく、一般にトレーニングのベースラインコスト（プレトレーニング）またはLLMの微調整を決定します。

エンタープライズユースケースと下流工程の自然言語処理タスク

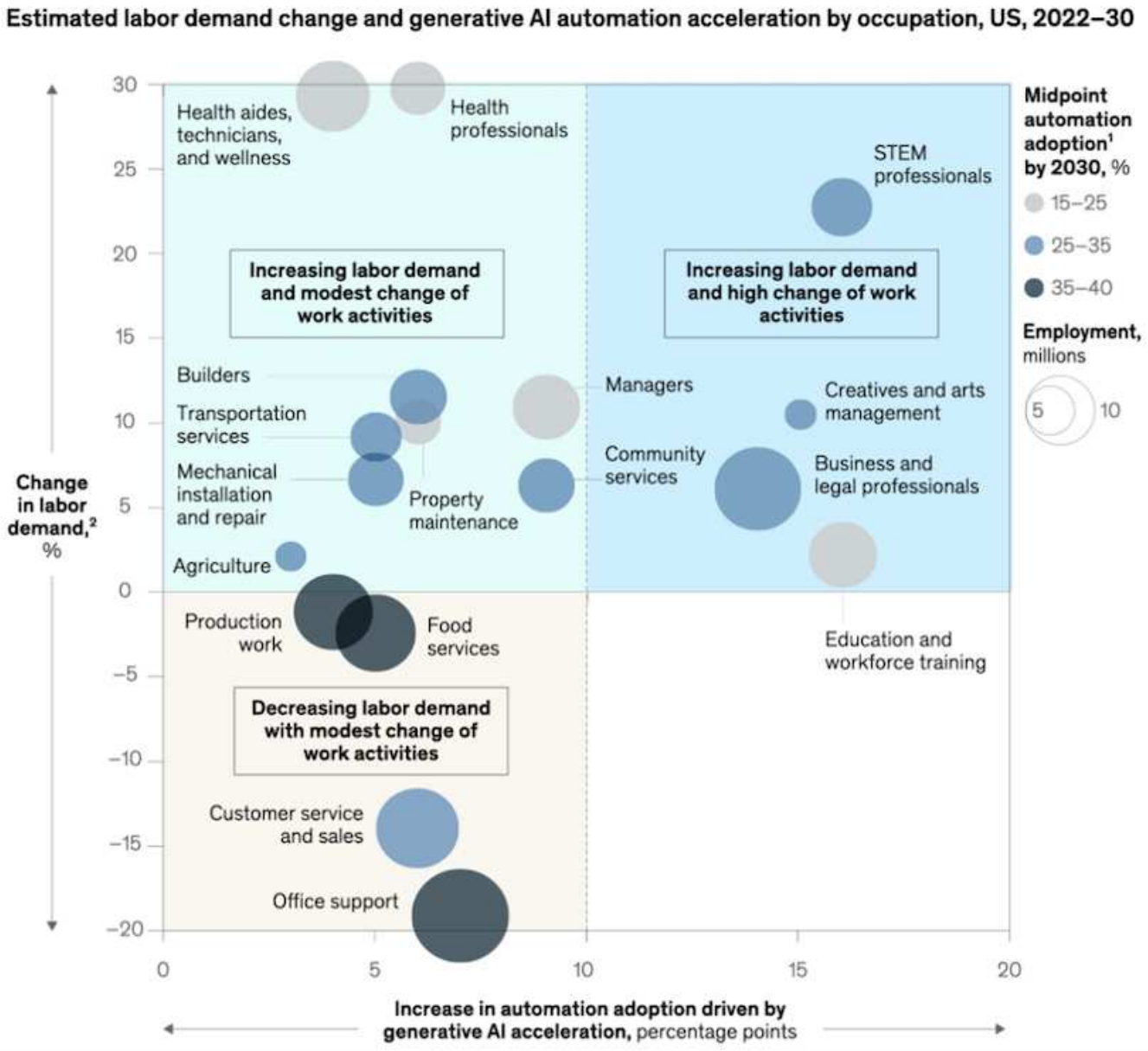
さまざまな業界の企業が、AIが既存のデータから新しい形の価値を引き出し、創出する可能性をますます明らかにしています。これは、事業運営、営業、マーケティング、法務サービスのためです。グローバルジェネレーティブAIのユースケースと投資に関するIDC（International Data Corporation）のマーケットインテリジェンスによると、ソフトウェア開発と製品設計におけるナレッジ管理が最も影響を受け、続いて開発者向けのマーケティングとコード生成のストーリーライン作成が行われます。医療の分野では、臨床研究機関が医療の新しい分野を開拓しています。ProteinBERTのような事前トレーニングされたモデルには、Gene Ontology（GO）アノテーションが組み込まれており、医薬品のタンパク質構造を迅速に設計できます。これは、創薬、バイオインフォマティクス、分子生物学における重要なマイルストーンとなります。バイオテクノロジー企業は、人工知能によって発見された生成医療のためのヒト試験を開始しました。これは、肺線維症(IPF)のような肺組織の不可逆的な瘢痕化を引き起こす肺疾患の治療を目的としたものです。

図1：ジェネレーティブAIの推進要因となっているユースケース

Chatbots	Drug discovery
Text generation	Genome model expression
Image generation	Classification
Code generation	Speech-to-Text

ジェネレーティブAIによる自動化の採用の増加は、多くの職種の業務活動の需要と供給にも変化をもたらしています。マッキンゼーによると、米国の労働市場（下の図）は急速な変化を遂げており、この変化はAIの影響を考慮した場合にのみ継続する可能性があります。

出典：McKinsey & Company



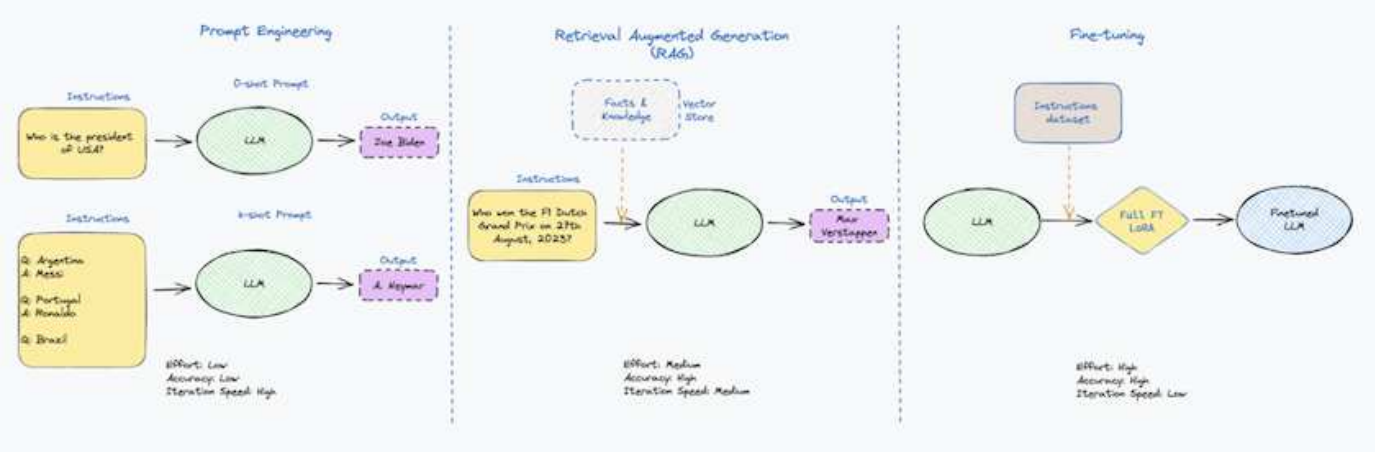
ジェネレーティブAIにおけるストレージの役割

LLMは主にディープラーニング、GPU、コンピューティングに依存しています。ただし、GPUバッファがいっぱいになると、データをストレージにすばやく書き込む必要があります。一部のAIモデルはメモリで実行できるほど小型ではありますが、LLMでは大規模なデータセットへの高速アクセスを実現するために、高いIOPSと高スループットのストレージが必要です。特に、数十億のトークンや数百万の画像が含まれる場合はなおさらです。LLMの一般的なGPUメモリ要件では、10億個のパラメータを持つモデルをトレーニングするために必要なメモリは、32ビットの完全精度で最大80GBになる可能性があります。その場合、Meta's llama 2は70億から700億までの規模のLLMファミリであり、70 x 80、つまり約70億個のパラメータが必要になる可能性があります。5600GBまたは5.6TBのGPU RAM。さらに、必要なメモリの量は、生成するトークンの最大数に直接比例します。たとえば、最大512トークン(約380ワード)の出力を生成する場合は、 "512 MB"。重要ではないように思えるかもしれませんが、より大きなバッチを実行したい場合は、合計が始まります。そのため、組織がメモリ内でLLMのトレーニングや微調整を行うには非常にコストがかかり、ストレージは生成AIの基盤となります。

LLMへの3つの主なアプローチ

ほとんどの企業では、現在の傾向に基づいて、LLMの導入アプローチを3つの基本シナリオにまとめることができます。最近の "「Harvard Business Review」" 記事：（1）トレーニング（事前トレーニング）LLMをゼロから作成する–コストがかかり、エキスパートのAI/MLスキルが必要。（2）エンタープライズデータを使用した基盤モデルの微調整（複雑で実行可能）。（3）検索拡張生成（RAG）を使用して、企業データを含むドキュメントリポジトリ、API、ベクターデータベースを照会する。これらのそれぞれには、さまざまなタイプの問題を解決するために使用される、実装における労力、反復速度、コスト効率、モデルの精度の間にトレードオフがあります(下の図)。

図3：問題の種類



基盤モデル

基礎モデル(FM)は、ベースモデルとも呼ばれ、ラベル付けされていない膨大な量のデータでトレーニングされ、大規模な自己管理を使用して、一般的に下流のNLPタスクの広い範囲に適応された大規模なAIモデル(LLM)です。トレーニングデータは人間によってラベル付けされていないため、モデルは明示的にエンコードされるのではなく出現する。これは、モデルが明示的にプログラムされていなくても、ストーリーや独自の物語を生成できることを意味します。したがってFMの重要な特徴は均質化であり、同じ方法が多くの領域で使われていることを意味する。しかし、パーソナライゼーションと微調整の技術により、最近登場した製品に統合されたFMSは、テキスト、テキストから画像、テキストからコードの生成だけでなく、ドメイン固有のタスクやデバッグコードの説明にも適しています。例えば、OpenAIのCodexやMetaのCode Llamaのようななが、プログラミングタスクの自然言語記述に基づいて複数のプログラミング言語でコードを生成することができる。これらのモデルは、Python、C#、JavaScript、Perl、Ruby、 およびSQLを使用します。ユーザーの意図を理解し、ソフトウェア開発、コードの最適化、プログラミングタスクの自動化に役立つ目的のタスクを実行する特定のコードを生成します。

微調整、ドメイン特異性、再トレーニング

データ前処理とデータ前処理に続くLLM導入では、大規模で多様なデータセットでトレーニングされた事前トレーニングモデルを選択することが一般的です。微調整のコンテキストでは、次のようなオープンソースの大規模言語モデルになります。 "Meta’s Llama 2" 700億個のパラメータと2兆個のトークンでトレーニングされています。事前トレーニング済みモデルを選択したら、次のステップでは、ドメイン固有のデータに基づいてモデルを微調整します。これには、モデルのパラメータを調整し、特定のドメインやタスクに適応するように新しいデータをトレーニングすることが含まれます。たとえば、BloombergGPTは、金融業界にサービスを提供する幅広い金融データのトレーニングを受けた独自のLLMです。特定のタスクのために設計され訓練されたドメイン固有のモデルは、通常、その範囲内でより高い精度とパフォーマンスを発揮しますが、他のタスクやドメイン間での転送性は低くなります。ビジネス環境やデータが一定期間にわたって変化すると、FMの予測精度は、テスト中のパフォーマンスと比較して低下し始める可能性があります。これは、モデルの再トレーニングや微調整が重要になるときです。従来のAI / MLでのモデルの再トレーニングとは、導入したMLモデルを新しいデータで更新することを指します。通常、2種類のドリフトを排除するために実行されます。(1)概念ド

リフト–入力変数とターゲット変数のリンクが時間の経過とともに変化すると、変化を予測したいものの概要が発生するため、モデルは不正確な予測を生成する可能性があります。(2)データドリフト–入力データの特性が変化し、時間の経過とともに顧客の習慣や行動が変化し、モデルがそのような変化に対応できない場合に発生します。同様の方法で、環境FMS/LLMの再トレーニングを行いますが、コストが高くなる可能性があります(数百万ドル)。したがって、ほとんどの組織が検討することはできません。現在も活発な研究が行われており、LLMOpsの分野で発展している。そのため、再トレーニングの代わりに、微調整されたFMSでモデルの崩壊が発生した場合、企業は新しいデータセットで再び微調整(はるかに安価)を選択することができます。コストの観点から、以下はAzure-OpenAI Servicesのモデル価格表の例です。タスクカテゴリごとに、特定のデータセットのモデルを微調整して評価できます。

出典：Microsoft Azure

Model	**Per 1000 token**
Text-Ada	$0.0001
GPT-3.5 Turbo	$0.003
GPT-4	$0.06
Text-Davinci	$0.02
Model	**Per 100 images**
Dall-E	$2

迅速なエンジニアリングと推論

プロンプトエンジニアリングとは、モデルの重みを更新せずに必要なタスクを実行するためにLLMと通信する効果的な方法を指します。AIモデルのトレーニングと微調整が自然言語処理アプリケーションにとって重要であるのと同じように、推論も同様に重要であり、トレーニング済みモデルがユーザプロンプトに応答します。一般に、推論のシステム要件は、最適な応答を生成するために数十億個の保存モデルパラメータを適用できる必要があるため、LLMからGPUにデータを供給するAIストレージシステムの読み取りパフォーマンスにはるかに依存します。

LLMOps、モデルモニタリング、およびベクトルストア

従来の機械学習運用（MLOps）と同様に、Large Language Model Operations（LLMOps）でも、データサイエンティストやDevOpsエンジニアと、本番環境でLLMを管理するためのツールやベストプラクティスを連携させる必要があります。ただし、LLMのワークフローと技術スタックは、いくつかの点で異なる場合があります。たとえば、LangChain stringなどのフレームワークを使用して構築されたLLMパイプラインは、ベクトルストアやベクトルデータベースなどの外部埋め込みエンドポイントへの複数のLLM API呼び出しを組み合わせて構築されます。（ベクターデータベースのように）ダウンストリームコネクタに埋め込みエンドポイントとベクトルストアを使用することは、データの格納方法とアクセス方法の重要な発展を表しています。ゼロから開発された従来のMLモデルとは異なり、LLMは、より特定の領域でパフォーマンスを向上させるために新しいデータで微調整されたFMSから始まるため、転送学習に依存することがよくあります。したがって、LLMOPは、リスク管理とモデル崩壊モニタリングの機能を提供することが非常に重要です。

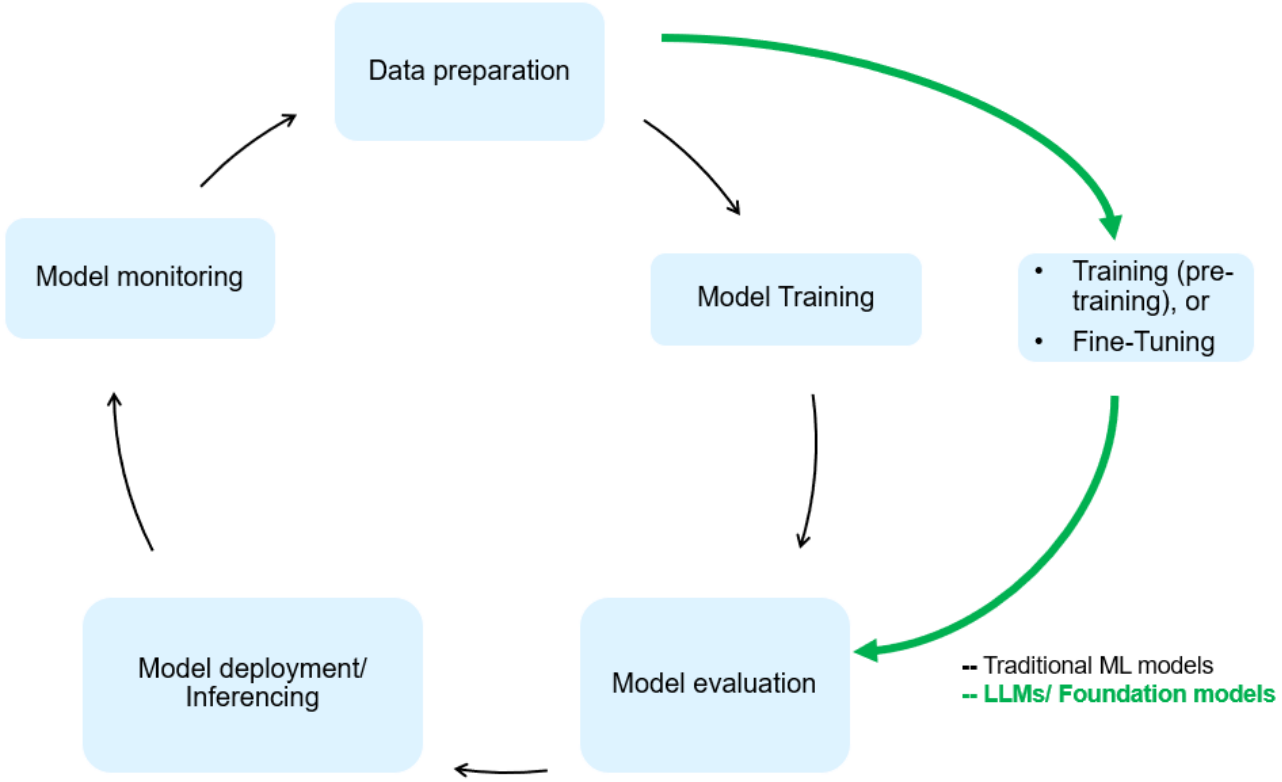
ジェネレーティブ**AI**の時代におけるリスクと倫理

「ChatGPT–It’s slick but still spews nonsense.」–MIT Tech Review.ガベージイン-ガベージアウトは、コンピューティングにおいて常に困難な課題でした。生成型AIとの唯一の違いは、ごみの信頼性が高く、結果が不正

確になることです。LLMは、構築している物語に合うように事実を発明する傾向があります。そのため、生成型AIを同等のAIでコストを削減する絶好の機会と見なしている企業は、システムを正直で倫理的に保つために、ディープフェイクを効率的に検出し、バイアスを減らし、リスクを軽減する必要があります。エンドツーエンドの暗号化とAIガードレールにより、データモビリティ、データ品質、データガバナンス、データ保護をサポートする堅牢なAIインフラを備えた自由に流れるデータパイプラインは、責任ある説明可能な生成AIモデルの設計において卓越しています。

お客様のシナリオと**NetApp**

図3：機械学習/大規模言語モデルのワークフロー



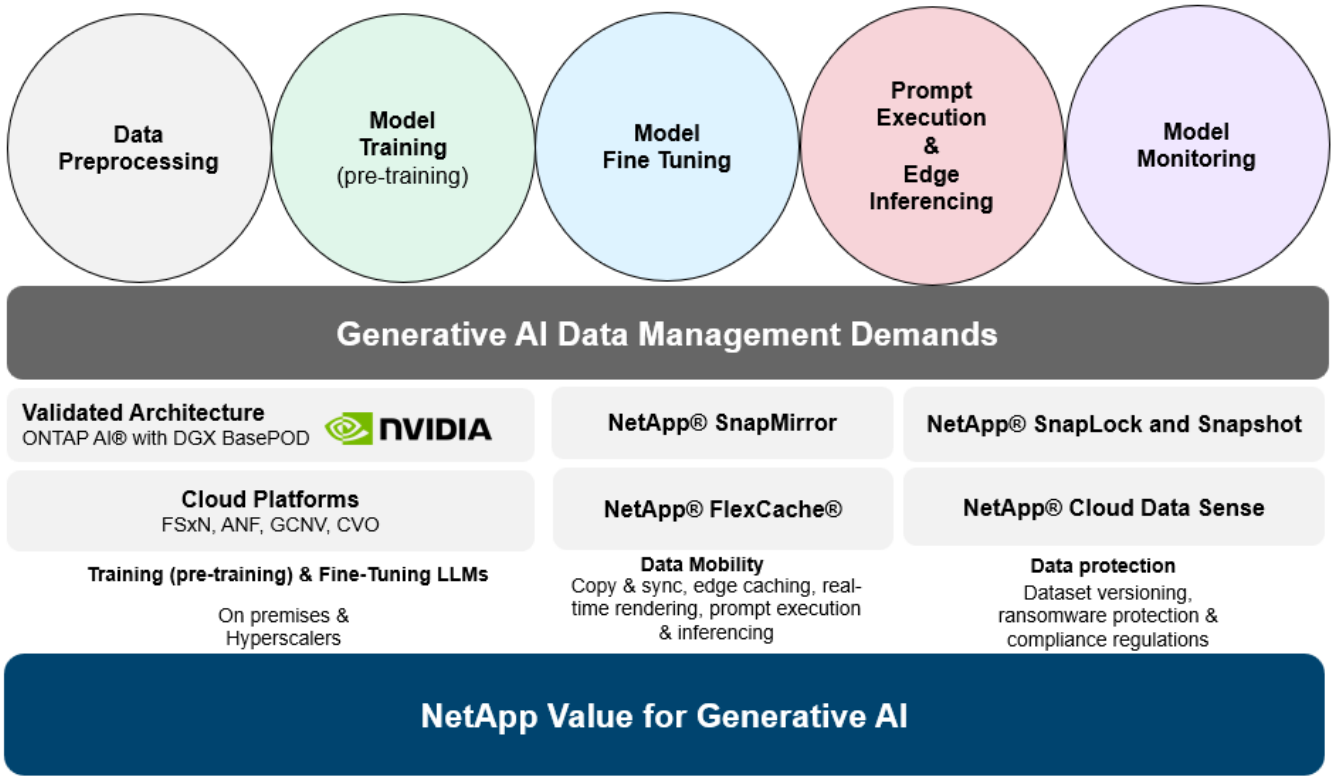
トレーニングか微調整か？ LLMモデルを最初からトレーニングするか、事前にトレーニングされたFMを微調整するか、RAGを使用して基礎モデル以外のドキュメントリポジトリからデータを取得し、プロンプトを強化するか、 （b）オープンソースのLLM（Llama 2など）または独自のFMS（ChatGPT、Bard、AWS Bedrockなど）を活用することは、組織にとって戦略的な決定です。各アプローチには、コスト効率、データの重力、運用、モデルの精度、LLMの管理のトレードオフがあります。

企業としてのNetAppは、社内のワークカルチャーや、製品設計やエンジニアリングの取り組みにAIを取り入れています。たとえば、ネットアップの自律型ランサムウェア対策は、AIと機械学習を使用して構築されています。ファイルシステムの異常を早期に検出し、運用に影響が及ぶ前に脅威を特定するのに役立ちます。次に、NetAppは、販売や在庫予測、チャットボットなどのビジネスオペレーションに予測AIを使用して、コールセンター製品サポートサービス、技術仕様、保証、サービスマニュアルなどの顧客を支援します。3つ目は、NetAppが、需要予測、医療画像処理、センチメント分析などの予測AIソリューションを構築するお客様にサービスを提供する製品とソリューションを通じて、AIデータパイプラインとML / LLMワークフローでお客様に価値を提供することです。 また、NetApp®ONTAP AI®、NetApp®SnapMirror®、NetApp®FlexCache®などのNetApp製品と機能を使用して、製造部門での産業画像の異常検出や、銀行や金融サービスでのマネーロンダリング防止や不正検出に対応するGANなどの生成AIソリューションも提供します。

NetAppの機能

チャットボット、コード生成、画像生成、ゲノムモデル表現などの生成AIアプリケーションでのデータの移動と管理は、エッジ、プライベートデータセンター、ハイブリッドマルチクラウドエコシステム全体にわたって可能です。例えば、ChatGPTのような事前訓練されたモデルのAPIを介して公開されたエンドユーザーアプリから航空券をビジネスクラスにアップグレードするのを支援するリアルタイムAIボットは、乗客情報がインターネット上で公開されていないため、単独でそのタスクを達成することはできません。APIは、ハイブリッドまたはマルチクラウドエコシステムに存在する可能性のある航空会社からの乗客の個人情報とチケット情報にアクセスする必要があります。同様のシナリオは、LLMを使用して1対多のバイオ医療研究機関を含む創薬全体の臨床試験を完了するエンドユーザーアプリケーションを介して、薬物分子と患者データを共有する科学者にも当てはまるかもしれません。FMSまたはLLMに渡される機密データには、PII、財務情報、健康情報、生体認証データ、位置情報、 通信データ、オンライン行動、法的情報。リアルタイムのレンダリング、迅速な実行、エッジでの推論の場合、エンドユーザアプリケーションからストレージエンドポイントへ、オープンソースまたは独自のLLMモデルを介して、オンプレミスのデータセンターやパブリッククラウドプラットフォームにデータが移動されます。このようなすべてのシナリオで、大規模なトレーニングデータセットとその移動に依存するLLMを使用するAI運用では、データモビリティとデータ保護が不可欠です。

図4：AIとLLMの生成データパイプライン



ネットアップのストレージインフラ、データ、クラウドサービスのポートフォリオには、インテリジェントなデータ管理ソフトウェアが搭載されています。

データの準備: LLM技術スタックの最初の柱は、従来のMLスタックからほとんど変更されていません。AIパイプラインでのデータの前処理は、トレーニングや微調整の前にデータを正規化してクレンジングするために必要です。この手順には、Amazon S3階層の形式で格納されている場所、またはオンプレミスのストレージシステム（ファイルストアやNetApp StorageGRIDなどのオブジェクトストア）にある場所にデータを取り込むためのコネクタが含まれます。

- NetApp®ONTAP *は、データセンターとクラウドにおけるネットアップの重要なストレージ・ソリューションの基盤となる基盤テクノロジです。ONTAPには、サイバー攻撃に対するランサムウェアの自動保護、組み込みのデータ転送機能、オンプレミス、ハイブリッド、NAS、SAN、オブジェクトのマルチクラウド

など、さまざまなアーキテクチャ向けのStorage Efficiency機能など、データの管理と保護に関するさまざまな機能が搭載されています。 また、LLM環境のSoftware-Defined Storage（SDS）の状況についても説明します。

- NetApp®ONTAP AI®*は、ディープラーニングモデルのトレーニングに最適です。NetApp®ONTAP®は、ONTAPストレージクラスタとNVIDIA DGXコンピューティングノードを使用するNetAppのお客様向けに、NFS over RDMAを使用してNVIDIA GPU Direct Storage™をサポートします。ストレージからメモリへのソースデータセットの読み取りと処理を何度も実行できるコスト効率に優れたパフォーマンスにより、インテリジェンスが強化され、LLMへのトレーニング、微調整、拡張アクセスが可能になります。
- NetApp®FlexCache®*は、ファイル配信を簡素化し、アクティブに読み取られたデータのみをキャッシュするリモートキャッシュ機能です。これは、LLMのトレーニング、再トレーニング、微調整に役立ち、リアルタイムレンダリングやLLM推論などのビジネス要件を持つお客様に価値を提供します。
- NetApp®SnapMirror *は、任意の2つのONTAPシステム間でボリュームSnapshotをレプリケートするONTAP機能です。この機能により、エッジからオンプレミスのデータセンターやクラウドへのデータ転送が最適化されます。お客様がエンタープライズデータを含むRAGを使用してクラウドで生成型AIを開発したい場合は、SnapMirrorを使用して、オンプレミスクラウドとハイパースケーラクラウド間で安全かつ効率的にデータを移動できます。変更のみを効率的に転送し、帯域幅を節約し、レプリケーションを高速化するため、FMSまたはLLMのトレーニング、再トレーニング、微調整の運用中に不可欠なデータ移動機能を提供します。
- NetApp®SnapLock *は、ONTAPベースのストレージシステムでデータセットのバージョンを変更できないディスク機能を提供します。マイクロコアアーキテクチャは、FPolicy™ゼロトラストエンジンを使用して顧客データを保護するように設計されています。NetAppは、攻撃者が特にリソースを消費する方法でLLMとやり取りするときにサービス拒否(DoS)攻撃に対抗することで、顧客データの可用性を確保します。
- NetApp®Cloud Data Sense *は、エンタープライズデータセットに存在する個人情報の特定、マッピング、分類、ポリシーの制定、オンプレミスまたはクラウドのプライバシー要件への対応、セキュリティ体制の改善、規制への準拠を支援します。
- Cloud Data Senseを基盤とするNetApp®BlueXP™*分類。お客様は、データ資産全体にわたってデータのスキャン、分析、分類、対処、セキュリティリスクの検出、ストレージの最適化、クラウド導入の高速化を自動で実行できます。統合されたコントロールプレーンを介してストレージとデータサービスを統合し、GPUインスタンスをコンピューティングに使用し、ハイブリッドマルチクラウド環境をコールドストレージの階層化やアーカイブとバックアップに使用できます。
- NetAppファイル-オブジェクトの二重性*。NetApp ONTAPを使用すると、NFSとS3に対するデュアルプロトコルアクセスが可能になります。この解決策を使用すると、Amazon AWS SageMakerノートブックのNFSデータに、NetApp Cloud Volumes ONTAPのS3バケットを介してアクセスできます。これにより、NFSとS3の両方のデータを共有できるため、異種データソースへの簡単なアクセスが必要なお客様に柔軟性が提供されます。 たとえば、SageMaker上のMetaのLlama 2テキスト生成モデルのようなFMSを微調整し、ファイルオブジェクトバケットにアクセスできます。
- NetApp®Cloud Sync *サービスは、クラウドまたはオンプレミスの任意のターゲットにデータを移行するシンプルで安全な方法を提供します。Cloud Syncは、オンプレミスやクラウドのストレージ、NASストア、オブジェクトストア間でデータをシームレスに転送して同期します。
- NetApp XCP *は、Any-to-NetAppおよびネットアップ間のデータ移行を高速かつ信頼性の高い方法で実現するクライアントソフトウェアです。XCPは、Hadoop HDFSファイルシステムからONTAP NFS、S3、またはStorageGRIDに一括データを効率的に移動する機能も提供し、XCPファイル分析によってファイルシステムを可視化できます。
- NetApp®DataOps Toolkit *は、データサイエンティスト、DevOps、データエンジニアがさまざまなデータ管理タスクを簡単に実行できるPythonライブラリです。ハイパフォーマンスなスケールアウトNetAppストレージを基盤とするデータボリュームやJupyterLabワークスペースのプロビジョニング、クローニング、スナップショット作成など、さまざまなデータ管理タスクをほぼ瞬時に実行できます。

ネットアップの製品セキュリティ。LLMは、応答の中で不注意に機密データを明らかにする可能性があるため、LLMを活用するAIアプリケーションに関連する脆弱性を調査するCISOにとって懸念事項となります。OWASP(Open Worldwide Application Security Project)で概説されているように、データ中毒、データ漏えい、サービス拒否、LLM内での迅速な注入などのセキュリティ問題は、データの露出から不正アクセスへの攻撃者にサービスを提供する攻撃者に至るまで、企業に影響を与える可能性があります。データストレージの要件には、構造化データ、半構造化データ、非構造化データの整合性チェックと書き換え不可のスナップショットが含まれている必要があります。データセットのバージョン管理にはNetApp SnapshotとSnapLockが使用されています。厳格なロールベースアクセス制御（RBAC）、セキュアなプロトコル、保存中と転送中の両方のデータを保護する業界標準の暗号化を提供します。Cloud InsightsとCloud Data Senseを組み合わせることで、脅威の原因をフォレンジックで特定し、リストアするデータに優先順位を付けることができます。

*** ONTAP AIとDGX BasePOD ***

NVIDIA DGX BasePODを搭載したNetApp®ONTAP®AIリファレンスアーキテクチャは、機械学習（ML）と人工知能（AI）のワークロード向けの拡張性に優れたアーキテクチャです。LLMの重要なトレーニングフェーズでは、データは通常、データストレージからトレーニングクラスタに一定の間隔でコピーされます。このフェーズで使用されるサーバは、GPUを使用して計算処理を並列化し、大量のデータに備えています。高いGPU利用率を維持するには、物理I/O帯域幅のニーズを満たすことが非常に重要です。

*** NVIDIA AI Enterprise搭載ONTAP AI ***

NVIDIA AI Enterpriseは、NVIDIA認定システムを搭載したVMware vSphere上で動作するようにNVIDIAによって最適化、認定、サポートされている、AIとデータ分析のためのエンドツーエンドのクラウドネイティブスイートです。AIワークロードの導入、管理、拡張を簡易化し、最新のハイブリッドクラウド環境で容易に実行できます。ネットアップとVMwareを基盤とするNVIDIA AI Enterpriseは、シンプルで使いやすいパッケージで、エンタープライズクラスのAIワークロードとデータ管理を実現します。

*** 1P**クラウドプラットフォーム*****

フルマネージドのクラウドストレージサービスは、Microsoft AzureではAzure NetApp Files（ANF）、AWSではAmazon FSx for NetApp ONTAP（FSxN）、GoogleではGoogle Cloud NetApp Volumes（GNCV）としてネイティブに利用できます。1Pは、ハイパフォーマンスなマネージドファイルシステムです。パブリッククラウドでデータセキュリティを強化しながら可用性の高いAIワークロードを実行し、AWS SageMaker、Azure-OpenAI Services、GoogleのVertex AIなどのクラウドネイティブMLプラットフォームでLLM / FMSを微調整できます。

NetAppパートナー解決策スイート

NetAppは、コアデータ製品、テクノロジ、機能に加えて、強力なAIパートナーネットワークと緊密に連携し、お客様に付加価値を提供しています。

- AIシステムのNVIDIAガードレール*は、AIテクノロジーの倫理的かつ責任ある使用を保証するための保護手段として機能します。AI開発者は、特定のトピックに関するLLMベースのアプリケーションの動作を定義し、不要なトピックに関するディスカッションに参加できないようにすることができます。オープンソースのツールキットであるGuardrailsは、LLMを他のサービスにシームレスかつ安全に接続し、信頼性が高く安全で安全なLLM会話システムを構築する機能を提供します。
- Domino Data Lab *は、AI導入のどの段階にいても、ジェネレーティブAIを迅速、安全、経済的に構築し、製品化するための汎用性に優れたエンタープライズクラスのツールを提供します。Dominoのエンタープライズ MLOpsプラットフォームを使用すると、データサイエンティストは、好みのツールとすべてのデータを使用し、モデルをどこでも簡単にトレーニングして導入し、リスクとコスト効率に優れた方法で管理できます。すべてを1つのコントロールセンターから実行できます。

*エッジAI向けModzy *。NetApp®とModzyは提携して、画像、音声、テキスト、表など、あらゆる種類のデー

タに大規模なAIを提供しています。Modzyは、AIモデルを導入、統合、実行するためのMLOpsプラットフォームであり、データサイエンティストにモデル監視、ドリフト検出、説明性の機能を提供し、シームレスなLLM推論のための統合解決策を備えています。

- Run：AI *とNetAppは提携して、NetApp ONTAP AI解決策の独自機能とRun：AIクラスタ管理プラットフォームを実証し、AIワークロードのオーケストレーションを簡易化します。Spark、Ray、Dask、Rapidsの組み込み統合フレームワークにより、データ処理パイプラインを数百台のマシンに拡張するように設計されたGPUリソースを自動的に分割して結合します。

まとめ

ジェネレーティブAIは、質の高いデータを基にモデルをトレーニングした場合にのみ効果的な結果を生み出すことができます。LLMは目覚ましいマイルストーンを達成していますが、データモビリティとデータ品質に関連する制約、設計上の課題、リスクを認識することが重要です。LLMは、異種データソースの大規模で異なるトレーニングデータセットに依存しています。モデルによって生成された不正確な結果や偏った結果は、企業と消費者の両方を危険にさらす可能性があります。これらのリスクは、データ品質、データセキュリティ、データモビリティに関連するデータ管理の課題から生じる可能性のあるLLMの制約に対応する可能性があります。NetAppは、データの急増、データモビリティ、マルチクラウド管理、AIの採用によって発生する複雑さに対応するのに役立ちます。大規模なAIインフラと効率的なデータ管理は、ジェネレーティブAIなどのAIアプリケーションの成功を定義するうえで不可欠です。コスト効率、データガバナンス、倫理的なAIプラクティスを制御しながら、企業が必要に応じて拡張する機能を犠牲にすることなく、すべての導入シナリオをカバーすることが重要です。NetAppは、お客様のAI導入の簡易化と高速化を常に支援しています。

TR-4785：『NetApp EシリーズおよびBeeGFSを使用したAI導入』

Nagalakshmi Raju、Daniel Landes、Nathan Swartz、ネットアップAmine Bennani

人工知能（AI）、機械学習（ML）、ディープラーニング（DL）の各アプリケーションには、大規模なデータセットと計算処理が伴います。これらのワークロードを正常に実行するには、ストレージノードとコンピューティングノードの両方をシームレスにスケールアウトできるアジャイルインフラが必要です。このレポートには、AIトレーニングモデルを分散モードで実行するための手順が含まれており、コンピューティングノードとストレージノードをシームレスにスケールアウトできます。また、このレポートには、解決策 EシリーズストレージとBeeGFS並列ファイルシステムを組み合わせることで、AIワークロード向けに柔軟で対費用効果の高いシンプルな解決策 を実現する方法を示すさまざまなパフォーマンス指標も含まれています。

"TR-4785：『NetApp EシリーズおよびBeeGFSを使用したAI導入』"

NVA-1150-design：NetApp Eシリーズシステムの設計ガイドでQuantum StorNextを使用しました

ネットアップ、Ryan Rodine氏

このドキュメントでは、NetApp Eシリーズストレージシステムを使用したStorNextパラレルファイルシステム解決策 の設計方法について詳しく説明します。この解決策 では、NetApp EF280オールフラッシュアレイ、NetApp EF300オールフラッシュNVMeアレイ、EF600オールフラッシュNVMeアレイ、およびNetApp E5760ハイブリッドシステムについて説明します。Frametestベンチマークに基づいてパフォーマンス特性を評価し

ます。Frametestベンチマークは、メディアおよびエンターテイメント業界で広く使用されているツールです。

"NVA-1150-design：NetApp Eシリーズシステムの設計ガイドでQuantum StorNextを使用しました"

NVA-1150-deploy：Quantum StorNext with NetApp E-Series systems deployment guide

ネットアップ、Ryan Rodine氏

このドキュメントでは、NetApp EシリーズストレージシステムにStorNext並列ファイルシステム解決策 を導入する方法について詳しく説明します。この解決策 では、NetApp EF280オールフラッシュアレイ、NetApp EF300オールフラッシュNVMeアレイ、NetApp EF600オールフラッシュNVMeアレイ、およびNetApp E5760ハイブリッドシステムについて説明します。Frametestベンチマークに基づいてパフォーマンス特性を評価します。Frametestベンチマークは、メディアおよびエンターテイメント業界で広く使用されているツールです。

"NVA-1150-deploy：Quantum StorNext with NetApp E-Series systems deployment guide"