aws

Guida per gli sviluppatori

AWS Deep Learning AMIs

AWS Deep Learning AMIs: Guida per gli sviluppatori

Table of Contents

Che cos'è AWS Deep Learning AMIs?

AWS Deep Learning AMIs (DLAMI) fornisce immagini di macchine personalizzate che puoi utilizzare per il deep learning nel cloud. DLAMIsSono disponibili nella maggior parte Regioni AWS per una varietà di tipi di istanze Amazon Elastic Compute Cloud (AmazonEC2), da una CPU sola istanza di piccole dimensioni alle più recenti multiistanze ad alta potenza. GPU Sono DLAMIs preconfigurate con NVIDIACUDAe con le versioni più recenti dei NVIDIAframework DNN di deep learning più diffusi.

Informazioni sulla guida

Il contenuto di può aiutarti ad avviare e utilizzare il. DLAMIs La guida copre diversi casi d'uso comuni del deep learning, sia per la formazione che per l'inferenza. Spiega anche come scegliere quello giusto AMI per il tuo scopo e il tipo di istanze che potresti preferire.

Inoltre, DLAMIs includono diversi tutorial forniti dai framework supportati. Questa guida può mostrarti come attivare ogni framework e trovare i tutorial appropriati per iniziare. Contiene anche tutorial sulla formazione distribuita, sul debug e sull'utilizzo AWS Inferentia e AWS Trainium e altri concetti chiave. Per istruzioni su come configurare un server notebook Jupyter per eseguire i tutorial nel browser, consulta. Configurazione di un server di notebook Jupyter

Prerequisiti

Per eseguire correttamenteDLAMIs, ti consigliamo di conoscere gli strumenti da riga di comando e Python di base.

Casi d'uso di DLAMI di esempio

Di seguito sono riportati alcuni esempi di alcuni casi d'uso comuni per AWS Deep Learning AMIs (DLAMI).

Apprendimento del deep learning: DLAMI è un'ottima scelta per l'apprendimento o l'insegnamento di framework di machine learning e deep learning. In questo modo non è più DLAMIs necessario risolvere i problemi relativi alle installazioni di ciascun framework e farle funzionare sullo stesso computer. DLAMIsIncludono un notebook Jupyter e semplificano l'esecuzione dei tutorial forniti dai framework per chi non conosce l'apprendimento automatico e il deep learning.

Sviluppo di app: se sei uno sviluppatore di app interessato a utilizzare il deep learning per far sì che le tue app utilizzino gli ultimi progressi dell'intelligenza artificiale, allora è il banco di prova perfetto per te. DLAMI Ogni framework dispone di tutorial su come iniziare a utilizzare l'apprendimento profondo e molti di questi includono serie di modelli che ne semplificano l'utilizzo eliminando la necessità di creare personalmente reti neurali o eseguire il training dei modelli. Alcuni esempi mostrano come creare un'applicazione di rilevamento delle immagini in pochi minuti oppure un'app di riconoscimento vocale per un servizio di chatbot.

Apprendimento automatico e analisi dei dati: se sei un data scientist o sei interessato a elaborare i tuoi dati con il deep learning, scoprirai che molti framework supportano R e Spark. Troverai tutorial su come eseguire semplici regressioni, fino alla creazione di sistemi di elaborazione dati scalabili per sistemi di personalizzazione e di stima.

Ricerca: se sei un ricercatore che vuole provare un nuovo framework, testare un nuovo modello o addestrare nuovi modelli, allora e DLAMI AWS le funzionalità di scalabilità possono alleviare il problema delle noiose installazioni e della gestione di più nodi di formazione.

> **Note**
>
> Sebbene la scelta iniziale possa essere quella di aggiornare il tipo di istanza a un'istanza più grande con più istanze GPUs (fino a 8), è possibile anche scalare orizzontalmente creando un cluster di istanze. DLAMI Per ulteriori informazioni sulle build di cluster, consulta Informazioni correlate su DLAMI.

Caratteristiche di DLAMI

Le caratteristiche di AWS Deep Learning AMIs (DLAMI) includono framework di deep learning, GPU software, server di modelli e strumenti di visualizzazione dei modelli preinstallati.

Framework preinstallati

Attualmente esistono due versioni principali di DLAMI altre varianti relative al sistema operativo (OS) e alle versioni del software:

- Apprendimento approfondito AMI con Conda— Framework installati separatamente utilizzando `conda` pacchetti e ambienti Python separati.
- Deep Learning Learning Learning AMI— Nessun framework installato; solo e altre dipendenze. NVIDIACUDA

Il Deep Learning AMI con Conda utilizza `conda` ambienti per isolare ogni framework, in modo da poter passare da uno all'altro a piacimento senza preoccuparsi che le dipendenze entrino in conflitto. Il Deep Learning AMI con Conda supporta i seguenti framework:

- PyTorch
- TensorFlow 2

> **Note**
>
> DLAMInon supporta più i seguenti framework di deep learning: Apache, MXNet Microsoft Cognitive Toolkit (CNTK), Caffe, Caffe2, Theano, Chainer e Keras.

Software preinstallato GPU

Anche se usi un'istanza CPU -only, DLAMIs will have NVIDIACUDAe NVIDIA cu. DNN Il software installato è lo stesso indipendentemente dal tipo di istanza. Tieni presente che gli strumenti GPU specifici funzionano solo su un'istanza che ne ha almeno una. GPU Per ulteriori informazioni sui tipi di istanze, consultaSelezione del tipo di istanza per la DLAMI.

Per ulteriori informazioni suCUDA, vedereInstallazioni CUDA e binding di framework.

Gestione e visualizzazione dei modelli

Deep Learning AMI with Conda è preinstallato con server modello per TensorFlow e TensorBoard per la visualizzazione dei modelli. Per ulteriori informazioni, consulta TensorFlow Servire.

Nozioni di base

Come iniziare con DLAMI

Questa guida include suggerimenti su come scegliere DLAMI quella più adatta a te, selezionare un tipo di istanza adatto al tuo caso d'uso e al tuo budget e Informazioni correlate su DLAMI descrive le configurazioni personalizzate che potrebbero interessarti.

Se è la prima volta che usi AWS o utilizzando AmazonEC2, inizia conApprendimento approfondito AMI con Conda. Se conosci Amazon EC2 e altri AWS servizi come Amazon EMREFS, Amazon o Amazon S3 e se sei interessato a integrare tali servizi per progetti che richiedono formazione o inferenza distribuita, verifica se uno è adatto Informazioni correlate su DLAMI al tuo caso d'uso.

Ti consigliamo di consultare dapprima Scegli il tuo DLAMI per avere un'idea del tipo di istanza più adatto per la tua applicazione.

Fase succcessiva

Scegli il tuo DLAMI

Scegli il tuo DLAMI

Offriamo una vasta gamma di DLAMI opzioni. Per aiutarti a scegliere quella più DLAMI adatta al tuo caso d'uso, raggruppiamo le immagini in base al tipo di hardware o alla funzionalità per cui sono state sviluppate. I nostri raggruppamenti di primo livello sono:

- DLAMITipo: rispetto a Base CUDA rispetto a Framework singolo rispetto a Framework multiplo (Conda) DLAMI
- Architettura di calcolo: basata su x86 contro basata su ARM64 AWS Gravitone
- Tipo di processore: GPUhttps://docs.aws.amazon.com/dlami/latest/devguide/gpu versus Inferentia CPUversus Trainium
- SDKCUDA: contro AWS Neurone
- Sistema operativo: Amazon Linux contro Ubuntu

Gli altri argomenti di questa guida ti aiutano a informarti ulteriormente e ad approfondire i dettagli.

Argomenti

Argomento successivo

Apprendimento approfondito AMI con Conda

Installazioni CUDA e binding di framework

Sebbene il deep learning sia tutto piuttosto all'avanguardia, ogni framework offre versioni «stabili». Queste versioni stabili potrebbero non funzionare con l'implementazione e le funzionalità più recenti di CUDA o cuDNN. Il tuo caso d'uso e le funzionalità di cui hai bisogno possono aiutarti a scegliere un framework. Se non sei sicuro, usa l'ultima AMI Deep Learning con Conda. Dispone di `pip` binari ufficiali per tutti i framework con CUDA, utilizzando la versione più recente supportata da ciascun framework. Se desideri le versioni più recenti e personalizzare il tuo ambiente di deep learning, usa l'AMI Deep Learning Base.

Per ulteriori informazioni, consulta la nostra guida Stable e candidati alla release.

Scegli un DLAMI con CUDA

Deep Learning Learning Learning AMIHa tutte le serie di versioni CUDA disponibili

Apprendimento approfondito AMI con CondaHa tutte le serie di versioni CUDA disponibili

> **Note**
>
> Non includiamo più gli ambienti MXNet, CNTK, Caffe, Caffe2, Theano, Chainer o Keras Conda nel. AWS Deep Learning AMIs

Per i numeri di versione specifici del framework, consulta Note di rilascio per DLAMIs

Scegli questo tipo di DLAMI o scopri di più sui diversi DLAMI con l'opzione Next Up.

Scegli una delle versioni CUDA e consulta l'elenco completo dei DLAMI che hanno quella versione nell'Appendice, oppure scopri di più sui diversi DLAMI con l'opzione Next Up.

Argomento successivo

Deep Learning Learning Learning AMI

Argomenti correlati

- Per le istruzioni su come passare da una versione CUDA all'altra, consulta il tutorial Utilizzo della Deep Learning Base AMI.

Deep Learning Learning Learning AMI

L'AMI Deep Learning Base è come una tela vuota per il deep learning. Viene fornito con tutto il necessario fino al momento dell'installazione di un particolare framework e offre una scelta di versioni CUDA.

Perché scegliere la Base DLAMI

Questo gruppo di AMI è utile per chi collabora ai progetti e intende eseguire il fork di un progetto di apprendimento profondo e compilare la versione più recente. È destinato a chiunque desideri utilizzare il proprio ambiente avendo la certezza che il software NVIDIA più recente sia installato e funzionante, in modo da potersi concentrare sulla scelta dei framework e delle versioni che intende installare.

Scegli questo tipo di DLAMI o scopri di più sui diversi DLAMI con l'opzione Next Up.

Argomento successivo

DLAMI con Conda

Argomenti correlati

- Utilizzo dell'AMI Deep Learning Base

Apprendimento approfondito AMI con Conda

Il Conda DLAMI utilizza ambienti `conda` virtuali, sono presenti sia in framework multiplo che in framework singolo. DLAMIs Questi ambienti sono configurati per mantenere separate le diverse installazioni del framework e semplificare il passaggio da un framework all'altro. È ottimo per imparare e sperimentare tutti i framework che ha da offrire. DLAMI La maggior parte degli utenti ritiene che il nuovo Deep Learning AMI con Conda sia perfetto per loro.

Vengono aggiornati spesso con le ultime versioni dei framework e dispongono dei GPU driver e del software più recenti. Sono generalmente denominati AWS Deep Learning AMIs nella maggior parte dei documenti. Questi DLAMIs supportano i sistemi operativi Ubuntu 20.04, Amazon Linux 2. Il supporto dei sistemi operativi dipende dal supporto del sistema operativo upstream.

Stable e candidati alla release

Conda AMIs utilizza file binari ottimizzati delle versioni formali più recenti di ciascun framework. Versioni candidate e funzionalità sperimentali non sono previste. Le ottimizzazioni dipendono dal supporto del framework per tecnologie di accelerazione come quella di Intel MKLDNN, che velocizza l'addestramento e l'inferenza sui tipi di istanze C5 e C4. CPU I file binari sono inoltre compilati per supportare set di istruzioni Intel avanzati, inclusi, a titolo esemplificativoAVX, AVX -2, .1 e .2. SSE4 SSE4 Questi accelerano le operazioni vettoriali e a virgola mobile sulle architetture Intel. CPU Inoltre, GPU ad esempio i tipi, the CUDA e cu DNN vengono aggiornati con la versione supportata dall'ultima versione ufficiale.

Il Deep Learning AMI con Conda installa automaticamente la versione più ottimizzata del framework per la tua EC2 istanza Amazon alla prima attivazione del framework. Per ulteriori informazioni, vedi AMIUtilizzo del Deep Learning con Conda.

Se desideri eseguire l'installazione dal codice sorgente, utilizzando opzioni di build personalizzate o ottimizzate, Deep Learning Learning Learning AMI s potrebbe essere l'opzione migliore per te.

Impostare Python 2 come obsoleto

La comunità open source di Python ha ufficialmente interrotto il supporto per Python 2 il 1 gennaio 2020. La PyTorch community TensorFlow and ha annunciato che le versioni TensorFlow 2.1 e PyTorch 1.4 sono le ultime a supportare Python 2. Le versioni precedenti di DLAMI (v26, v25, ecc.) che contengono ambienti Python 2 Conda continuano a essere disponibili. Tuttavia, forniamo aggiornamenti agli ambienti Python 2 Conda su DLAMI versioni pubblicate in precedenza solo se sono presenti correzioni di sicurezza pubblicate dalla community open source per tali versioni. DLAMIle versioni con le ultime versioni dei PyTorch framework TensorFlow and non contengono gli ambienti Python 2 Conda.

CUDASupport

I numeri di CUDA versione specifici sono disponibili nelle note di GPU DLAMI rilascio.

Argomento successivo

Opzioni di architettura DLAMI

Argomenti correlati

- Per un tutorial sull'utilizzo di un Deep Learning AMI con Conda, consulta il AMIUtilizzo del Deep Learning con Conda tutorial.

Opzioni di architettura DLAMI

AWS Deep Learning AMIsI sono offerti con architetture Graviton2 basate su x86 o ARM64.AWS

Per informazioni su come iniziare a usare la GPU ARM64 DLAMI, vedere. La ARM64 DLAMI Per ulteriori dettagli sui tipi di istanze disponibili, consulta. Selezione del tipo di istanza per la DLAMI

Argomento successivo

Opzioni del sistema operativo DLAMI

Opzioni del sistema operativo DLAMI

I DLAMI sono disponibili nei seguenti sistemi operativi.

- Amazon Linux 2
- Ubuntu 20.04
- Ubuntu 22.04

Le versioni precedenti dei sistemi operativi sono disponibili su DLAMI obsoleti. Per ulteriori informazioni sulla deprecazione DLAMI, consulta Deprecazioni per DLAMI

Prima di scegliere un DLAMI, valuta il tipo di istanza di cui hai bisogno e identifica la tua AWS regione.

Argomento successivo

Selezione del tipo di istanza per la DLAMI

Selezione del tipo di istanza per la DLAMI

Più in generale, considera quanto segue quando selezioni un tipo di istanza per aDLAMI.

- Se non conosci il deep learning, allora un'istanza con una sola istanza GPU potrebbe soddisfare le tue esigenze.
- Se sei attento al budget, puoi utilizzare le istanze CPU -only.
- Se stai cercando di ottimizzare alte prestazioni ed efficienza in termini di costi per l'inferenza dei modelli di deep learning, puoi utilizzare istanze con AWS Chip Inferentia.
- Se stai cercando un'GPUistanza ad alte prestazioni con un'CPUarchitettura basata su ARM64, puoi utilizzare il tipo di istanza G5g.
- Se sei interessato a eseguire un modello preaddestrato per inferenza e previsioni, puoi collegare un Amazon Elastic Inference alla tua istanza Amazon. EC2 Amazon Elastic Inference ti dà accesso a un acceleratore con una frazione di a. GPU
- Per i servizi di inferenza ad alto volume, una singola CPU istanza con molta memoria o un cluster di tali istanze potrebbe essere una soluzione migliore.
- Se utilizzi un modello di grandi dimensioni con molti dati o batch di grandi dimensioni, allora hai bisogno di un'istanza più grande con più memoria. Puoi anche distribuire il tuo modello in un cluster diGPUs. Potresti scoprire che l'utilizzo di un'istanza con meno memoria è una soluzione migliore se riduci la dimensione del batch. Ciò potrebbe influire sulla precisione e sulla velocità di allenamento.
- Se sei interessato a eseguire applicazioni di machine learning utilizzando NVIDIA Collective Communications Library (NCCL) che richiedono alti livelli di comunicazioni tra i nodi su larga scala, potresti voler utilizzare Elastic Fabric Adapter () EFA.

Per maggiori dettagli sulle istanze, consulta Tipi di .

I seguenti argomenti forniscono informazioni sulle considerazioni relative al tipo di istanza.

> **Important**
>
> Il Deep Learning AMIs include driver, software o toolkit sviluppati, posseduti o forniti da Corporation. NVIDIA Accetti di utilizzare questi NVIDIA driver, software o toolkit solo su EC2 istanze Amazon che includono hardware. NVIDIA

Argomenti

- Prezzi delle DLAMI
- Disponibilità nelle regioni DLAMI
- GPUIstanze consigliate

- Istanze consigliate CPU
- Istanze Inferentia consigliate
- Istanze Trainium consigliate

Prezzi delle DLAMI

I framework di deep learning inclusi in DLAMI sono gratuiti e ognuno ha le proprie licenze open source. Sebbene il software incluso DLAMI sia gratuito, devi comunque pagare per l'hardware sottostante dell'EC2istanza Amazon.

Alcuni tipi di EC2 istanze Amazon sono etichettati come gratuiti. È possibile eseguirlo DLAMI su una di queste istanze gratuite. Ciò significa che l'utilizzo di DLAMI è completamente gratuito quando si utilizza solo la capacità dell'istanza. Se hai bisogno di un'istanza più potente con più CPU core, più spazio su disco, più o una o più RAMGPUs, allora hai bisogno di un'istanza che non rientri nella classe delle istanze di livello libero.

Per ulteriori informazioni sulla selezione e sui prezzi delle istanze, consulta EC2i prezzi di Amazon.

Disponibilità nelle regioni DLAMI

Ogni regione supporta una gamma diversa di tipi di istanza e spesso un tipo di istanza ha un costo leggermente diverso nelle diverse regioni. DLAMIsnon sono disponibili in tutte le regioni, ma è possibile DLAMIs copiarle nella regione desiderata. Per ulteriori informazioni, consulta AMI Copiare un file. Prendi nota dell'elenco di selezione delle regioni e assicurati di scegliere una regione più vicina a te o ai tuoi clienti. Se prevedi di utilizzarne più di una DLAMI e potenzialmente creare un cluster, assicurati di utilizzare la stessa regione per tutti i nodi del cluster.

Per maggiori informazioni sulle regioni, visita Regioni e .

Argomento successivo

GPUIstanze consigliate

GPUIstanze consigliate

Consigliamo un'GPUistanza per la maggior parte degli scopi di deep learning. La formazione di nuovi modelli è più rapida su un'GPUistanza che su un'CPUistanza. È possibile scalare in modo sublineare quando si dispone di più GPU istanze o se si utilizza un training distribuito su più istanze con. GPUs

I seguenti tipi di istanza supportano. DLAMI Per informazioni sulle opzioni relative ai tipi di GPU istanza e ai relativi utilizzi, vedi istanza e seleziona Accelerated Computing.

> **Note**
>
> La dimensione del modello dovrebbe essere un elemento di selezione di un'istanza. Se il modello supera quello disponibile per un'istanzaRAM, seleziona un tipo di istanza diverso con memoria sufficiente per l'applicazione.

- Le istanze Amazon EC2 P5e hanno fino a 8 NVIDIA Tesla H200. GPUs
- Le istanze Amazon EC2 P5 hanno fino a 8 NVIDIA Tesla H100. GPUs
- Le istanze Amazon EC2 P4 hanno fino a 8 NVIDIA Tesla A100. GPUs
- Le istanze Amazon EC2 P3 hanno fino a 8 NVIDIA Tesla V100. GPUs
- Le istanze Amazon EC2 G3 hanno fino a 4 NVIDIA Tesla M60. GPUs
- Le istanze Amazon EC2 G4 hanno fino a 4 T4NVIDIA. GPUs
- Le istanze Amazon EC2 G5 hanno fino a 8 NVIDIA istanze A10G. GPUs
- Le istanze Amazon EC2 G6 hanno fino a 8 NVIDIA istanze L4. GPUs
- Le istanze Amazon EC2 G6e hanno fino a 8 Tensor Core NVIDIA L40S. GPUs
- Le istanze Amazon EC2 G5g sono basate su ARM64 AWS Processori Graviton2.

DLAMIle istanze forniscono strumenti per monitorare e ottimizzare i processi. GPU Per ulteriori informazioni sul monitoraggio dei GPU processi, consulta. GPUMonitoraggio e ottimizzazione

Per tutorial specifici sull'utilizzo delle istanze G5g, consulta. La ARM64 DLAMI

Argomento successivo

Istanze consigliate CPU

Istanze consigliate CPU

Se hai un budget limitato, stai imparando a conoscere il deep learning o desideri semplicemente gestire un servizio di previsione, hai molte opzioni convenienti nella CPU categoria. Alcuni framework sfruttano i vantaggi di Intel MKLDNN, che velocizza la formazione e l'inferenza sui tipi di istanze C5

(non disponibile in tutte le regioni). CPU Per informazioni sui tipi di CPU istanza, consulta Tipi di e seleziona EC2Compute Optimized.

> **Note**
>
> La dimensione del modello dovrebbe essere un elemento di selezione di un'istanza. Se il modello supera quello disponibile per un'istanzaRAM, seleziona un tipo di istanza diverso con memoria sufficiente per l'applicazione.

- Le istanze Amazon EC2 C5 hanno fino a 72 processori Intel. vCPUs Le istanze C5 eccellono nella modellazione scientifica, nell'elaborazione in batch, nell'analisi distribuita, nell'elaborazione ad alte prestazioni (HPC) e nell'inferenza di machine e deep learning.

Argomento successivo

Istanze Inferentia consigliate

Istanze Inferentia consigliate

AWS Le istanze Inferentia sono progettate per fornire prestazioni elevate ed efficienza in termini di costi per i carichi di lavoro di inferenza dei modelli di deep learning. In particolare, i tipi di istanze Inf2 utilizzano AWS Chip Inferentia e AWS Neuron SDK, che è integrato con i più diffusi framework di apprendimento automatico come e. TensorFlow PyTorch

I clienti possono utilizzare le istanze Inf2 per eseguire applicazioni di inferenza di machine learning su larga scala come ricerca, motori di raccomandazione, visione artificiale, riconoscimento vocale, elaborazione del linguaggio naturale, personalizzazione e rilevamento delle frodi, al costo più basso del cloud.

> **Note**
>
> La dimensione del modello dovrebbe essere un elemento di selezione di un'istanza. Se il modello supera quello disponibile per un'istanzaRAM, seleziona un tipo di istanza diverso con memoria sufficiente per l'applicazione.

- Le istanze Amazon EC2 Inf2 ne hanno fino a 16 AWS Chip Inferentia e 100 Gbps di throughput di rete.

Per ulteriori informazioni su come iniziare con AWS InferentiaDLAMIs, vedi. Il AWS Chip Inferentia con DLAMI

Argomento successivo

Istanze Trainium consigliate

Istanze Trainium consigliate

AWS Le istanze Trainium sono progettate per fornire prestazioni elevate ed efficienza in termini di costi per i carichi di lavoro di inferenza dei modelli di deep learning. In particolare, i tipi di istanze Trn1 utilizzano AWS Chip Trainium e il AWS Neuron SDK, che è integrato con i più diffusi framework di apprendimento automatico come e. TensorFlow PyTorch

I clienti possono utilizzare le istanze Trn1 per eseguire applicazioni di inferenza di machine learning su larga scala come ricerca, motori di raccomandazione, visione artificiale, riconoscimento vocale, elaborazione del linguaggio naturale, personalizzazione e rilevamento delle frodi, al costo più basso del cloud.

> **Note**
>
> La dimensione del modello dovrebbe essere un elemento di selezione di un'istanza. Se il modello supera quello disponibile per un'istanzaRAM, seleziona un tipo di istanza diverso con memoria sufficiente per l'applicazione.

- Le istanze Amazon EC2 Trn1 ne hanno fino a 16 AWS Chip Trainium e 100 Gbps di throughput di rete.

Avvio e configurazione di una DLAMI

Se sei qui dovresti già avere una buona idea di quale AMI vuoi lanciare. In caso contrario, trova a DLAMI e il relativo hardware, i framework e il recupero degli ID in. Note di rilascio per DLAMIs

Devi inoltre sapere quale tipo di istanza e regione vuoi utilizzare. Se non hai ancora deciso, consulta Selezione del tipo di istanza per la DLAMI.

> **Note**
>
> Useremo p3.16xlarge come tipo di istanza predefinito negli esempi. Sostituisci questo tipo con quello che desideri utilizzare.

> **Important**
>
> Se prevedi di utilizzare Elastic Inference, devi completare Elastic Inference Setup prima di lanciare il tuo. DLAMI

Argomenti

Fase 1: avvio di una DLAMI

> **Note**
>
> Per questa procedura dettagliata, potremmo fare riferimenti specifici al Deep Learning AMI (Ubuntu 18.04). Anche se ne selezioni un altroDLAMI, dovresti essere in grado di seguire questa guida.

1. Trova l'ID del tuo DLAMI
2. Avvia un'EC2istanza Amazon dal tuo DLAMI

   Utilizzerai la EC2 console Amazon. Segui le istruzioni dettagliate in Avvio da Amazon EC2 Console

   > **Tip**
   >
   > CLIOpzione: Se scegli di far girare un file DLAMI usando il AWS CLI, avrai bisogno dell'ID, AMI della regione e del tipo di istanza e delle informazioni sul token di sicurezza. Assicurati che la tua istanza AMI sia IDs pronta. Se non hai configurato il AWS CLItuttavia, fallo prima usando la guida per l'installazione di AWS Interfaccia a riga di comando.

3. Una volta completata la procedura di una di queste opzioni, attendere che l'istanza sia pronta. Questo processo richiede in genere soltanto alcuni minuti. È possibile verificare lo stato dell'istanza nella EC2console.

Recupera l'ID DLAMI

Ciascuno AMI possiede un identificatore univoco (ID). Puoi richiedere l'ID DLAMI di tua scelta con il AWS Interfaccia a riga di comando (AWS CLI). Se non hai già il AWS CLI installato, vedi Guida introduttiva a AWS CLI.

> **Note**
>
> Consulta le note di DLAMI rilascio riportate in Note di rilascio per DLAMIs per ulteriori configurazioni software (driver, versioni python, EBS tipo Amazon).

1. Assicurati che il tuo AWS le credenziali sono configurate.

   ```
   aws configure
   ```

2. Usa il seguente comando per recuperare l'ID del tuo DLAMI o trovare la query fornita nel AWS Deep Learning AMIs note di rilascio.

   ```
   aws ec2 describe-images --region us-east-1 --owners amazon \
   ```

```
--filters 'Name=name,Values=Deep Learning Base OSS Nvidia Driver GPU AMI (Ubuntu
 22.04) ????????' 'Name=state,Values=available' \
--query 'reverse(sort_by(Images, &CreationDate))[:1].ImageId' --output text
```

> **Note**
>
> È possibile specificare una versione di rilascio per un determinato framework o ottenere l'ultima versione sostituendo il numero di versione con un punto interrogativo.

3. L'output visualizzato dovrebbe essere simile al seguente:

```
ami-09ee1a996ac214ce7
```

Copia questo DLAMI ID e premi q per uscire dal prompt.

Fase succcessiva

Avvio da Amazon EC2 Console

Avvio da Amazon EC2 Console

> **Note**
>
> Per avviare un'istanza con Elastic Fabric Adapter (EFA), fai riferimento a questi passaggi.

1. Apri la EC2console.
2. Notare la regione corrente nella parte superiore del riquadro di navigazione. Se questo non è quello che desideri Regione AWS, modifica questa opzione prima di procedere. Per ulteriori informazioni, consulta .
3. Scegliere Launch Instance (Avvia istanza).
4. Inserisci un nome per l'istanza e seleziona DLAMI quello più adatto a te.
   a. Trovane una esistente DLAMI in My AMIs o scegli Quick Start.
   b. Cerca per DLAMI ID. Sfoglia le opzioni, quindi seleziona la tua scelta.

5. Scegliere un tipo di istanza. Puoi trovare le famiglie di istanze consigliate DLAMI nel AWS Deep Learning AMIs note di rilascio. Per consigli generali sui tipi di DLAMI istanza, vedi Selezione delle istanze.

   > **Note**
   >
   > Se desideri utilizzare Elastic Inference (EI), fai clic su Configure Instance Details, seleziona Aggiungi un acceleratore Amazon EI, quindi seleziona la dimensione dell'acceleratore Amazon EI.

6. Scegliere Launch Instance (Avvia istanza).

> **Tip**
>
> Dai un'occhiata a Come iniziare a usare il deep learning usando il AWS Deep Learning AMI per una guida dettagliata con schermate.

Fase succcessiva

Fase 2: connessione alla DLAMI

Fase 2: connessione alla DLAMI

Connect a DLAMI quello che hai avviato da un client (Windows, macOS o Linux). Per ulteriori informazioni, consulta Connect to Your Linux Instance nella Amazon EC2 User Guide.

Tieni a portata di mano una copia del comando di SSH login se desideri eseguire la configurazione di Jupyter dopo l'accesso. Ne utilizzerai una variante per la connessione alla pagina Web di Jupyter.

Fase succcessiva

Fase 3: Metti alla prova il tuo DLAMI

Fase 3: Metti alla prova il tuo DLAMI

A seconda della DLAMI versione, hai diverse opzioni di test:

- Apprendimento approfondito AMI con Conda— vai aAMIUtilizzo del Deep Learning con Conda.

- Deep Learning Learning Learning AMI— fate riferimento alla documentazione di installazione del framework desiderato.

Puoi anche creare un notebook Jupyter, provare dei tutorial o scrivere codice in Python. Per ulteriori informazioni, consulta Configurazione di un server di notebook Jupyter.

Fase 4: Gestisci la tua DLAMI istanza

Aggiorna regolarmente il sistema operativo e le altre applicazioni software utilizzate mediante l'installazione di patch e aggiornamenti non appena diventano disponibili.

Se utilizzi Amazon Linux o Ubuntu, quando accedi al tuoDLAMI, ricevi una notifica se sono disponibili aggiornamenti e consulta le istruzioni per l'aggiornamento. Per ulteriori informazioni sulla manutenzione di Amazon Linux, consulta Updating Instance Software. Per le istanze Ubuntu, consulta la documentazione ufficiale di Ubuntu.

In Windows, consulta Windows Update regolarmente per verificare se sono disponibili nuovi aggiornamenti software e di sicurezza. Se lo preferisci, puoi installare gli aggiornamenti automaticamente.

> **Important**
>
> Per informazioni sulle vulnerabilità di Meltdown e Spectre e su come applicare patch al sistema operativo per risolverle, consulta il Bollettino sulla sicurezza AWS-2018-013.

Pulizia

Quando non ti serve piùDLAMI, puoi interromperlo o interromperlo per evitare di incorrere in addebiti continui. Un'istanza arrestata non viene chiusa e può essere riavviata in seguito. Le configurazioni, i file e altre informazioni non volatili vengono archiviate in un volume su Amazon S3. Per la conservazione del volume durante l'arresto dell'istanza ti verrà addebitato la tariffa S3 minore, ma non incorrerai nei costi relativi alle risorse di calcolo. Quando riavvii l'istanza, questa monterà quel volume e i tuoi dati saranno disponibili. Se termini un'istanza, questa non sarà più disponibile e non potrà essere riavviata. In realtà, i dati sono ancora presenti in S3, quindi per evitare ulteriori costi devi eliminare anche il volume. Per ulteriori istruzioni, consulta Terminate Your Instance nella Amazon EC2 User Guide.

Configurazione di un server di notebook Jupyter

Un server notebook Jupyter consente di creare ed eseguire notebook Jupyter dalla propria istanza DLAMI. Con i notebook Jupyter, è possibile condurre esperimenti di apprendimento automatico (ML) per l'addestramento e l'inferenza utilizzando l'AWSinfrastruttura e accedendo ai pacchetti integrati nel DLAMI. Per ulteriori informazioni sui notebook Jupyter, consulta la documentazione di Jupyter Notebook.

Per configurare un server notebook Jupyter, è necessario:

- Configura il server notebook Jupyter sulla tua istanza DLAMI Amazon EC2.
- Configurare il client in modo da eseguire la connessione al server di notebook Jupyter. Forniamo istruzioni di configurazione per client Windows, macOS e Linux.
- Testare la configurazione accedendo al server di notebook Jupyter.

Per completare i passaggi per configurare un Jupyter, segui le istruzioni nei seguenti argomenti. Dopo aver configurato un server notebook Jupyter, consultaTutorial per l'esecuzione di notebook Jupyter le informazioni sull'esecuzione dei notebook di esempio forniti nel DLAMI.

Argomenti

Proteggi il tuo server Jupyter

Di seguito abbiamo configurato Jupyter con SSL e una password personalizzata.

Connect all'istanza Amazon EC2 e completa la seguente procedura.

Configurazione del server Jupyter

1. Jupyter fornisce una utility per le password. Esegui il comando seguente e inserisci la tua password preferita al prompt.

```
$ jupyter notebook password
```

Il risultato sarà simile al seguente:

```
Enter password:
Verify password:
[NotebookPasswordApp] Wrote hashed password to /home/ubuntu/.jupyter/
jupyter_notebook_config.json
```

2. Crea un certificato SSL autofirmato Segui i prompt per compilare la tua località. È necessario immettere `.` se si desidera lasciare vuoto un prompt. Le tue risposte non hanno alcun impatto su queste funzionalità del certificato.

```
$ cd ~
$ mkdir ssl
$ cd ssl
$ openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout mykey.key -out
 mycert.pem
```

> **Note**
>
> Può voler creare un certificato SSL normale firmato da terze parti e che non causi un avviso di protezione da parte del browser. Questo processo è decisamente più compesso. Consulta la documentazione di Jupyter per ulteriori informazioni.

Fase successiva

Avvio del server di notebook Jupyter

Avvio del server di notebook Jupyter

Ora puoi attivare il server Jupyter accedendo all'istanza ed eseguendo il comando seguente che usa il certificato SSL creato nel passaggio precedente.

```
$ jupyter notebook --certfile=~/ssl/mycert.pem --keyfile ~/ssl/mykey.key
```

Dopo l'avvio del server, puoi collegarti allo stesso tramite un tunnel SSH dal tuo computer client. Durante l'esecuzione del server, vedrai un messaggio di Jupyter che conferma tale condizione. A

questo punto, ignora il messaggio indicante che puoi accedere al server tramite un URL localhost, poiché non funzionerà fino a quando non crei un tunnel.

> **Note**
>
> Jupyter gestirà la commutazione di ambienti quando cambi framework Jupyter utilizzando l'interfaccia Web. Per ulteriori informazioni, consulta Passaggio a un altro ambiente con Jupyter.

Fase successiva

Configurazione del client per la connessione al server Jupyter

Configurazione del client per la connessione al server Jupyter

Dopo la configurazione del client per la connessione al server di notebook Jupyter, puoi creare e accedere ai notebook sul server nel workspace ed eseguire il codice di apprendimento profondo sul server.

Per informazioni sulla configurazione, scegli uno dei collegamenti seguenti.

Argomenti

- Configurazione di un client Windows
- Configurazione di un client Linux o macOS

Configurazione di un client Windows

Preparazione

Assicurati di disporre delle informazioni esposte di seguito, necessarie per configurare il tunnel SSH:

- Il nome DNS pubblico dell'istanza Amazon EC2. Puoi trovare tale nome nella console EC2.
- La coppia di chiavi per il file della chiave privata. Per ulteriori informazioni sull'accesso alla coppia di chiavi, consulta Coppia di chiavi Amazon EC2 nella Guida per l'utente per istanze Linux di Amazon EC2.

Utilizzo dei notebook Jupyter da un client Windows

Consulta queste guide sulla connessione all'istanza Amazon EC2 da un client Windows.

1. Risoluzione dei problemi di connessione all'istanza
2. Connessione all'istanza Linux da Windows tramite PuTTY

Per creare un tunnel su un server con Jupyter in esecuzione, un approccio consigliato è quello di installare Git Bash sul tuo client Windows, quindi seguire le istruzioni del client Linux/macOS . Tuttavia, puoi usare qualsiasi approccio desiderato per aprire un tunnel SSH con mapping di porta. Fai riferimento alla documentazione di Jupyter per ulteriori informazioni.

Fase successiva

Configurazione di un client Linux o macOS

Configurazione di un client Linux o macOS

1. Apri un terminale .
2. Esegui il seguente comando per inoltrare tutte le richieste dalla porta locale 8888 alla porta 8888 della tua istanza Amazon EC2 remota. Aggiorna il comando sostituendo la posizione della chiave per accedere all'istanza Amazon EC2 e il nome DNS pubblico della tua istanza Amazon EC2. Nota, per un'AMI Amazon Linux, il nome utente è `ec2-user` anziché `ubuntu`.

   ```
   $ ssh -i ~/mykeypair.pem -N -f -L 8888:localhost:8888 ubuntu@ec2-###-##-##-###.compute-1.amazonaws.com
   ```

   Questo comando apre un tunnel tra il client e l'istanza Amazon EC2 remota su cui è in esecuzione il server notebook Jupyter.

Fase successiva

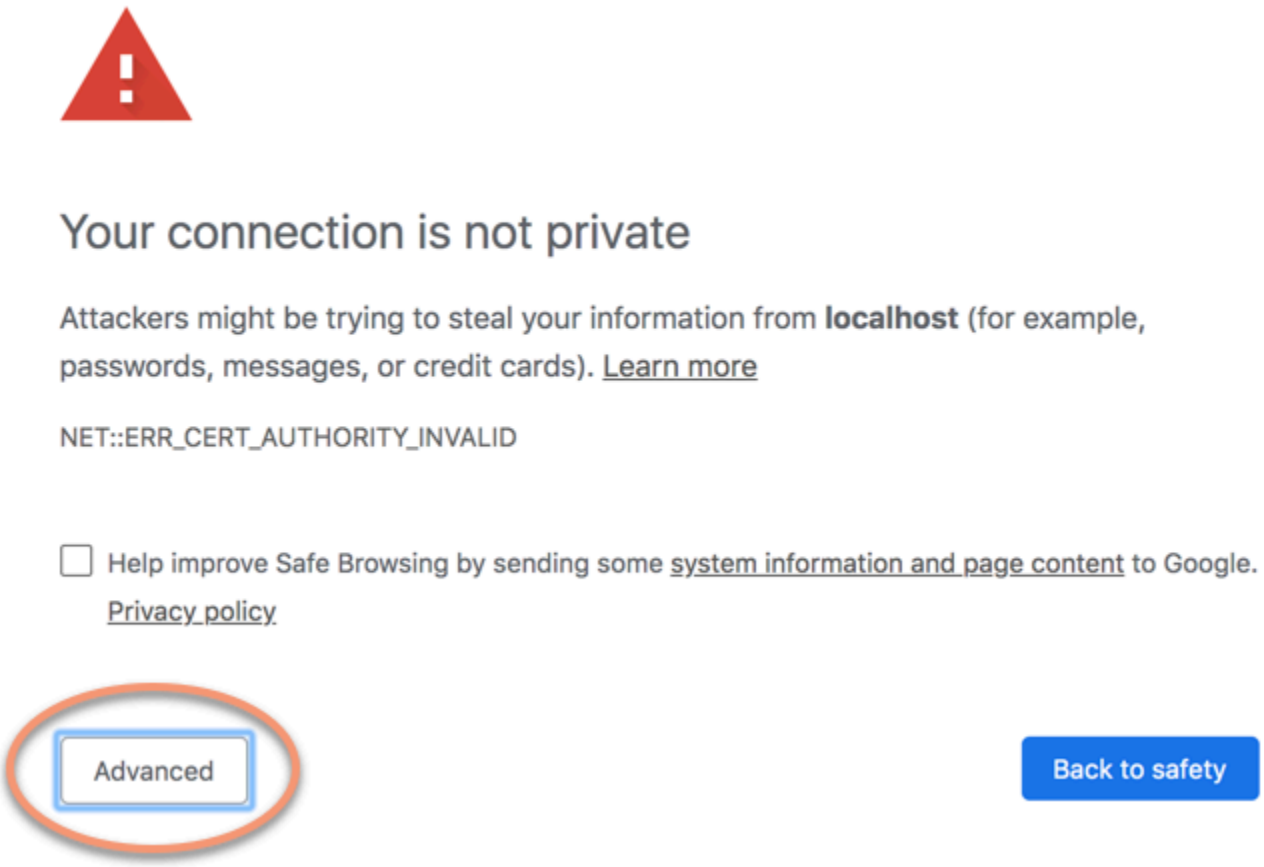
Test mediante l'accesso al server di notebook Jupyter

Test mediante l'accesso al server di notebook Jupyter

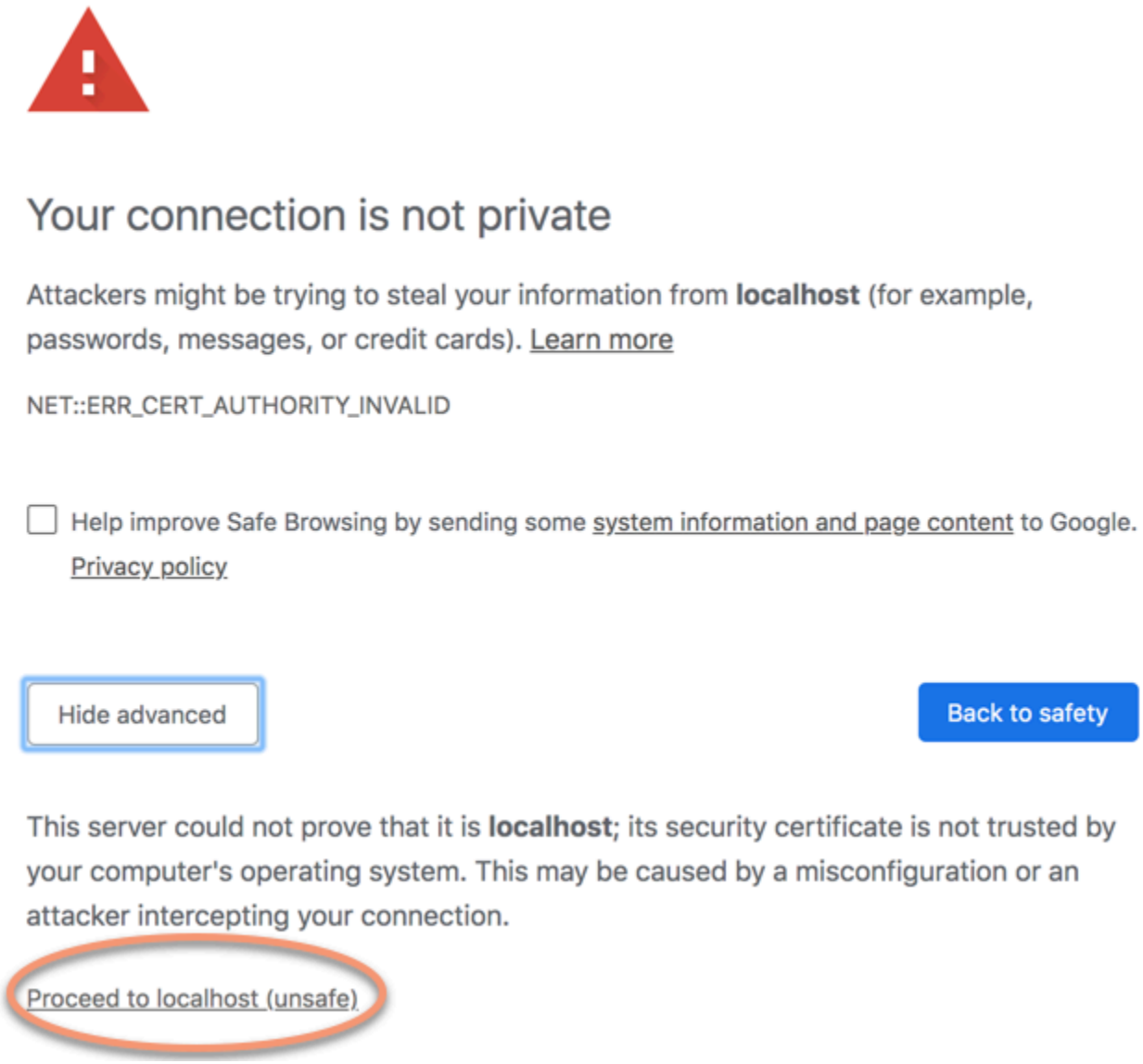
Ora sei pronto a accedere al server di notebook Jupyter.

La fase successiva consiste nel testare la connessione al server tramite il tuo browser.

1. Nella barra degli indirizzi del browser, digita il seguente URL, oppure fai clic su questo collegamento: https://localhost:8888
2. Con un certificato SSL autofirmato, il tuo browser ti mostrerà un avviso e ti verrà richiesto di evitare di proseguire nella visita del sito Web.



Poiché hai impostato tu stesso tale elemento, puoi proseguire in sicurezza.. A seconda del browser verrà visualizzato un pulsante denominato “avanzato”, “mostra dettagli” o simile.



Fai clic su questo elemento, quindi fai clic sul link "procedi verso il localhost". Se la connessione riesce, viene visualizzata la pagina Web del server di notebook Jupyter. A questo punto, ti verrà chiesta la password impostata precedentemente.

Ora hai accesso al server notebook Jupyter in esecuzione sul DLAMI. È possibile creare nuovi notebook o eseguire i Tutorial forniti.

Usando un DLAMI

Argomenti

Le sezioni seguenti descrivono come utilizzare il Deep Learning AMI con Conda per cambiare ambiente, eseguire codice di esempio da ciascuno dei framework ed eseguire Jupyter in modo da poter provare diversi tutorial per notebook.

AMIUtilizzo del Deep Learning con Conda

Argomenti

Introduzione al Deep Learning AMI con Conda

Conda è un sistema open source per la gestione di pacchetti e di ambienti eseguibile in Windows, macOS e Linux. Conda installa, esegue e aggiorna rapidamente i pacchetti e le relative dipendenze. Conda agevola la creazione, il salvataggio e il caricamento di ambienti sul computer locale nonché il passaggio dall'uno all'altro.

Il Deep Learning AMI con Conda è stato configurato per consentirti di passare facilmente da un ambiente di deep learning all'altro. Le istruzioni seguenti sono relative ad alcuni comandi `conda` di base. Ti consentono inoltre di verificare il corretto funzionamento dell'importazione di base del framework e che puoi eseguire alcune semplici operazioni con il framework. È quindi possibile passare a tutorial più approfonditi forniti con DLAMI o agli esempi dei framework disponibili sul sito del progetto di ciascun framework.

Accedi a Your DLAMI

Dopo aver effettuato l'accesso al server, verrà visualizzato il «messaggio del giorno» (MOTD) del server che descrive vari comandi Conda che è possibile utilizzare per passare da un framework di deep learning all'altro. Di seguito è riportato un esempio. MOTD Le specifiche MOTD possono variare man mano che DLAMI vengono rilasciate nuove versioni di.

```
=============================================================================
        AMI Name: Deep Learning OSS Nvidia Driver AMI (Amazon Linux 2) Version 77
        Supported EC2 instances: G4dn, G5, G6, Gr6, P4d, P4de, P5
            * To activate pre-built tensorflow environment, run: 'source activate
 tensorflow2_p310'
            * To activate pre-built pytorch environment, run: 'source activate
 pytorch_p310'
            * To activate pre-built python3 environment, run: 'source activate python3'

        NVIDIA driver version: 535.161.08

    CUDA versions available: cuda-11.7 cuda-11.8 cuda-12.0 cuda-12.1 cuda-12.2

    Default CUDA version is 12.1

    Release notes: https://docs.aws.amazon.com/dlami/latest/devguide/appendix-ami-
release-notes.html
    AWS Deep Learning AMI Homepage: https://aws.amazon.com/machine-learning/amis/
    Developer Guide and Release Notes: https://docs.aws.amazon.com/dlami/latest/
devguide/what-is-dlami.html
    Support: https://forums.aws.amazon.com/forum.jspa?forumID=263
    For a fully managed experience, check out Amazon SageMaker at https://
aws.amazon.com/sagemaker
    =============================================================================
```

Avvia l' TensorFlow ambiente

> **Note**
>
> Il caricamento del primo ambiente Conda può risultare alquanto lungo. Il Deep Learning AMI with Conda installa automaticamente la versione più ottimizzata del framework per l'EC2istanza alla prima attivazione del framework. Non dovrebbero aversi ulteriori ritardi.

1. Attiva l'ambiente TensorFlow virtuale per Python 3.

```
$ source activate tensorflow2_p310
```

2. Avvia il iPython terminale.

```
(tensorflow2_p310)$ ipython
```

3. Esegui un TensorFlow programma rapido.

```
import tensorflow as tf
hello = tf.constant('Hello, TensorFlow!')
sess = tf.Session()
print(sess.run(hello))
```

Viene visualizzato il messaggio "Hello, Tensorflow!".

Argomento successivo

Tutorial per l'esecuzione di notebook Jupyter

Passa all'ambiente PyTorch Python 3

Se sei ancora nella iPython console, usa`quit()`, quindi preparati a cambiare ambiente.

- Attiva l'ambiente PyTorch virtuale per Python 3.

```
$ source activate pytorch_p310
```

Prova del codice PyTorch

Per testare la tua installazione, usa Python per scrivere PyTorch codice che crea e stampa un array.

1. Avvia il iPython terminale.

```
(pytorch_p310)$ ipython
```

2. Importazione PyTorch.

```
import torch
```

È possibile che venga visualizzato un messaggio di avviso su un pacchetto di terze parti. Puoi ignorarla.

3. Crea una matrice 5x3 con gli elementi inizializzati in modo casuale. Stampare la matrice.

```
x = torch.rand(5, 3)
print(x)
```

Verificare il risultato.

```
tensor([[0.3105, 0.5983, 0.5410],
        [0.0234, 0.0934, 0.0371],
        [0.9740, 0.1439, 0.3107],
        [0.6461, 0.9035, 0.5715],
        [0.4401, 0.7990, 0.8913]])
```

Rimozione ambienti

Se esaurisci lo spazio suDLAMI, puoi scegliere di disinstallare i pacchetti Conda che non stai utilizzando:

```
conda env list
conda env remove --name <env_name>
```

Utilizzo della Deep Learning Base AMI

Utilizzo della Deep Learning Base AMI

The Base AMI è dotato di una piattaforma di base di GPU driver e librerie di accelerazione per implementare un ambiente di deep learning personalizzato. Per impostazione predefinita, AMI è configurato con qualsiasi ambiente di versione. NVIDIA CUDA È inoltre possibile passare da una versione all'altra diCUDA. Consulta le seguenti istruzioni per eseguire questa operazione.

Configurazione delle versioni CUDA

È possibile verificare la CUDA versione eseguendo NVIDIA il `nvcc` programma.

```
nvcc --version
```

È possibile selezionare e verificare una CUDA versione particolare con il seguente comando bash:

```
sudo rm /usr/local/cuda
sudo ln -s /usr/local/cuda-12.0 /usr/local/cuda
```

Per ulteriori informazioni, consultate le note di DLAMI rilascio di Base.

Tutorial per l'esecuzione di notebook Jupyter

I tutorial e gli esempi vengono forniti insieme ai sorgenti di ogni progetto di deep learning e nella maggior parte dei casi possono essere eseguiti su qualsiasi dispositivo. DLAMI Se scegli la Apprendimento approfondito AMI con Conda, beneficerai di alcuni tutorial selezionati già configurati e pronti per l'uso.

> **Important**
>
> Per eseguire i tutorial per notebook Jupyter installati su, è necessario. DLAMI Configurazione di un server di notebook Jupyter

Una volta che il server Jupyter è in esecuzione, puoi eseguire i tutorial mediante il browser web. Se stai usando il Deep Learning AMI con Conda o se hai configurato ambienti Python, puoi cambiare i kernel Python dall'interfaccia del notebook Jupyter. Seleziona il kernel appropriato prima di eseguire un tutorial specifico di un framework. Ulteriori esempi di ciò sono forniti agli utenti del Deep Learning with Conda. AMI

> **Note**
>
> Molti tutorial richiedono moduli Python aggiuntivi che potrebbero non essere configurati sul tuo. DLAMI Se ricevi un errore del tipo`"xyz module not found"`, accedi aDLAMI, attiva l'ambiente come descritto sopra, quindi installa i moduli necessari.

> **Tip**
>
> I tutorial e gli esempi di deep learning spesso si basano su uno o più. GPUs Se il tipo di istanza non dispone di unGPU, potrebbe essere necessario modificare parte del codice dell'esempio per farlo funzionare.

Esplorazione dei tutorial installati

Dopo aver effettuato l'accesso al server Jupyter e aver visualizzato la directory dei tutorial (solo su Deep Learning AMI con Conda), ti verranno presentate cartelle di tutorial in base al nome di ciascun framework. Se non vedi un framework nell'elenco, significa che i tutorial per quel framework non sono disponibili nella versione corrente. DLAMI Fai clic sul nome del framework per visualizzare i tutorial elencati, quindi fai clic su un tutorial per avviarlo.

La prima volta che esegui un notebook sul Deep Learning AMI con Conda, vorrà sapere quale ambiente desideri utilizzare. Ti verrà richiesto di selezionarlo da un elenco. Ogni ambiente è denominato in base a questo modello:

`Environment (conda_framework_python-version)`

Ad esempio, potresti vedere`Environment (conda_mxnet_p36)`, il che significa che l'ambiente ha MXNet Python 3. L'altra variante di questo sarebbe`Environment (conda_mxnet_p27)`, il che significa che l'ambiente ha MXNet Python 2.

> **Tip**
>
> Se sei preoccupato per quale versione di CUDA è attiva, un modo per vederlo è MOTD quando accedi per la prima volta a. DLAMI

Passaggio a un altro ambiente con Jupyter

Se decidi di provare un tutorial per un altro framework, assicurati di verificare il kernel attualmente in esecuzione. Questa informazione può essere visualizzata in alto a destra nell'interfaccia Jupyter, sotto il pulsante di disconnessione. Puoi cambiare il kernel su qualsiasi notebook aperto scegliendo la voce del menu Jupyter Kernel, quindi Change Kernel (Cambia kernel) e infine facendo clic sull'ambiente appropriato per il notebook in esecuzione.

A questo punto, devi rieseguire tutte le celle in quanto una modifica nel kernel cancellerà lo stato di quanto eseguito in precedenza.

> **Tip**
>
> Passare da un framework all'altro può risultare divertente e istruttivo, ma esiste il rischio di esaurimento della memoria. Se appaiono degli errori, esamina la finestra del terminale in cui il server Jupyter è in esecuzione. Qui sono presenti messaggi utili e registrazione degli errori, e potresti visualizzare un out-of-memory errore. Per correggere il problema, puoi accedere alla home page del server Jupyter, fare clic sulla scheda Running (In esecuzione) e quindi su Shutdown (Chiusura) per ogni tutorial che probabilmente è ancora in esecuzione in background e che utilizza tutta la memoria.

Tutorial

Di seguito sono riportati dei tutorial su come utilizzare il Deep Learning AMI con il software di Conda.

Argomenti

- Attivazione di framework
- Formazione distribuita con Elastic Fabric Adapter
- GPUMonitoraggio e ottimizzazione
- Il AWS Chip Inferentia con DLAMI
- La ARM64 DLAMI
- Inferenza
- Model serving

Attivazione di framework

Di seguito sono riportati i framework di deep learning installati su Deep Learning with Conda. AMI Fai clic su un framework per informazioni su come attivarlo.

Argomenti

- PyTorch
- TensorFlow 2

PyTorch

Attivazione PyTorch

Quando viene rilasciato un pacchetto Conda stabile di un framework, viene testato e preinstallato su. DLAMI Se desideri eseguire la build notturna più recente non testata, puoi eseguire l'Installa PyTorch Nightly Build (sperimentale) manualmente.

Per attivare il framework attualmente installato, segui queste istruzioni sul tuo Deep Learning AMI con Conda.

Per PyTorch su Python 3 con CUDA e MKL -DNN, esegui questo comando:

```
$ source activate pytorch_p310
```

Avvia il iPython terminale.

```
(pytorch_p310)$ ipython
```

Esegui un PyTorch programma rapido.

```
import torch
x = torch.rand(5, 3)
print(x)
print(x.size())
y = torch.rand(5, 3)
print(torch.add(x, y))
```

Dovrebbe essere visualizzata la matrice random iniziale stampata, quindi le dimensioni della stessa e infine l'aggiunta di un'altra matrice random.

Installa PyTorch Nightly Build (sperimentale)

Come eseguire l'installazione PyTorch da una build notturna

Puoi installare la PyTorch build più recente in uno o entrambi gli ambienti PyTorch Conda sul tuo Deep Learning AMI with Conda.

1. • (Opzione per Python 3) - Attiva l'ambiente Python 3: PyTorch

   ```
   $ source activate pytorch_p310
   ```

2. Per gli altri passaggi, si presuppone che venga utilizzato l'ambiente `pytorch_p310`. Rimuovi il file attualmente installato: PyTorch

```
(pytorch_p310)$ pip uninstall torch
```

3.
   - (Opzione per GPU istanze): installa l'ultima build notturna di PyTorch con .0: CUDA

```
(pytorch_p310)$ pip install torch_nightly -f https://download.pytorch.org/whl/
nightly/cu100/torch_nightly.html
```

   - (Opzione per CPU le istanze): installa l'ultima build notturna di per le istanze senza: PyTorch GPUs

```
(pytorch_p310)$ pip install torch_nightly -f https://download.pytorch.org/whl/
nightly/cpu/torch_nightly.html
```

4. Per verificare di aver installato correttamente l'ultima versione nightly, avvia il IPython terminale e controlla la versione di. PyTorch

```
(pytorch_p310)$ ipython
```

```
import torch
print (torch.__version__)
```

   L'output dovrebbe essere simile a `1.0.0.dev20180922`

5. Per verificare che la PyTorch nightly build funzioni correttamente con l'MNISTesempio, puoi eseguire uno script di test dal PyTorch repository degli esempi:

```
(pytorch_p310)$ cd ~
(pytorch_p310)$ git clone https://github.com/pytorch/examples.git pytorch_examples
(pytorch_p310)$ cd pytorch_examples/mnist
(pytorch_p310)$ python main.py || exit 1
```

Altri tutorial

Per ulteriori tutorial ed esempi, fate riferimento ai documenti ufficiali, alla documentazione e al sito Web del framework. PyTorch PyTorch

TensorFlow 2

Questo tutorial mostra come attivare TensorFlow 2 su un'istanza che esegue Deep Learning AMI con Conda (DLAMIsu Conda) ed eseguire un TensorFlow programma 2.

Quando viene rilasciato un pacchetto Conda stabile di un framework, viene testato e preinstallato su. DLAMI

Attivazione 2 TensorFlow

Per andare TensorFlow avanti DLAMI con Conda

1. Per attivarne TensorFlow 2, apri un'istanza Amazon Elastic Compute Cloud (AmazonEC2) DLAMI con Conda.
2. Per TensorFlow 2 e Keras 2 su Python 3 CUDA con 10.1 MKL e DNN -, esegui questo comando:

   ```
   $ source activate tensorflow2_p310
   ```
3. Avvia il terminale: iPython

   ```
   (tensorflow2_p310)$ ipython
   ```
4. Esegui un programma TensorFlow 2 per verificare che funzioni correttamente:

   ```
   import tensorflow as tf
   hello = tf.constant('Hello, TensorFlow!')
   tf.print(hello)
   ```

   Viene visualizzato `Hello, TensorFlow!`.

Altri tutorial

Per ulteriori tutorial ed esempi, consulta la TensorFlow documentazione per TensorFlow Python API o visita il sito Web. TensorFlow

Formazione distribuita con Elastic Fabric Adapter

Un Elastic Fabric Adapter (EFA) è un dispositivo di rete che puoi collegare all'DLAMIistanza per accelerare le applicazioni High Performance Computing (HPC). EFAconsente di ottenere le

prestazioni applicative di un HPC cluster locale, con la scalabilità, la flessibilità e l'elasticità fornite da AWS Cloud.

Negli argomenti seguenti viene illustrato come iniziare EFA a utilizzareDLAMI.

> **Note**
>
> Scegli il tuo DLAMI da questo GPUDLAMIelenco di base

Argomenti

- Avvio di un AWS Deep Learning AMIs Istanza con EFA
- Utilizzo su EFA DLAMI

Avvio di un AWS Deep Learning AMIs Istanza con EFA

La versione più recente di Base DLAMI è pronta all'uso EFA e viene fornita con i driver necessari, i moduli del kernel, libfabric, openmpi e il plugin per le istanze. NCCL OFI GPU

Puoi trovare le CUDA versioni supportate di una Base DLAMI nelle note di rilascio.

Nota:

- Quando si esegue un'NCCLapplicazione utilizzando `mpirun` onEFA, è necessario specificare il percorso completo dell'installazione EFA supportata nel modo seguente:

```
/opt/amazon/openmpi/bin/mpirun <command>
```

- Per abilitare l'uso dell'applicazioneEFA, aggiungilo `FI_PROVIDER="efa"` al `mpirun` comando come mostrato inUtilizzo su EFA DLAMI.

Argomenti

- Preparazione di un gruppo di sicurezza abilitato per EFA
- Avvio dell'istanza
- Verifica allegato EFA

Preparazione di un gruppo di sicurezza abilitato per EFA

EFArichiede un gruppo di sicurezza che consenta tutto il traffico in entrata e in uscita da e verso il gruppo di sicurezza stesso. Per ulteriori informazioni, consulta la EFA documentazione.

1. Apri la EC2 console Amazon all'indirizzo https://console.aws.amazon.com/ec2/.
2. Nel riquadro di navigazione, scegliere Security Groups (Gruppi di sicurezza) e quindi Create Security Group (Crea gruppo di sicurezza).
3. Nella finestra Create Security Group (Crea gruppo di sicurezza) effettuare le operazioni seguenti:
   - In Nome gruppo di sicurezza, immettere un nome descrittivo per il gruppo di sicurezza, ad esempio `EFA-enabled security group`.
   - (Facoltativo) In Description (Descrizione), inserire una breve descrizione del gruppo di sicurezza.
   - Per VPC, seleziona le istanze VPC in cui intendi avviare le istanze EFA abilitate.
   - Scegli Create (Crea) .
4. Selezionare il gruppo di sicurezza creato e, nella scheda Description (Descrizione), copiare il valore Group ID (ID gruppo).
5. Nelle schede In entrata e In uscita, procedi come segue:
   - Seleziona Edit (Modifica).
   - In Type (Tipo), selezionare All traffic (Tutto il traffico).
   - In Source (Origine), scegliere Custom (Personalizzata).
   - Incollare nel campo l'ID del gruppo di sicurezza copiato in precedenza.
   - Seleziona Salva.
6. Abilitare il traffico in entrata facente riferimento a Autorizzazione del traffico in entrata per le istanze Linux. Se salti questo passaggio, non sarai in grado di comunicare con la tua istanza. DLAMI

Avvio dell'istanza

EFAsul AWS Deep Learning AMIs è attualmente supportato con i seguenti tipi di istanze e sistemi operativi:

- P3DN.24xlarge: Amazon Linux 2, Ubuntu 20.04
- P4D.24xlarge: Amazon Linux 2, Ubuntu 20.04

- P5.48xLarge: Amazon Linux 2, Ubuntu 20.04

La sezione seguente mostra come avviare un'istanza abilitata. EFA DLAMI Per ulteriori informazioni sull'avvio di un'istanza EFA abilitata, consulta Launch EFA -Enabled Instances into a Cluster Placement Group.

1. Apri la EC2 console Amazon all'indirizzo https://console.aws.amazon.com/ec2/.
2. Scegliere Launch Instance (Avvia istanza).
3. Nella pagina Scegli una AMI pagina, seleziona una delle opzioni supportate DLAMI disponibili nella pagina delle note di DLAMI rilascio
4. Nella pagina Scegli un tipo di istanza, seleziona uno dei seguenti tipi di istanza supportati, quindi scegli Avanti: Configura i dettagli dell'istanza. Fai riferimento a questo link per l'elenco delle istanze supportate: Guida introduttiva a and EFA MPI
5. Nella pagina Configure Instance Details (Configura i dettagli dell'istanza), procedere come segue:
   - Per Numero di istanze, inserisci il numero di istanze EFA abilitate che desideri avviare.
   - Per Rete e sottorete, seleziona la sottorete VPC e in cui avviare le istanze.
   - [Facoltativo] Per il gruppo di posizionamento, selezionate Aggiungi istanza al gruppo di posizionamento. Per ottenere prestazioni ottimali, avviare le istanze all'interno di un gruppo di collocazione.
   - [Facoltativo] Per il nome del gruppo di collocamento, selezionate Aggiungi a un nuovo gruppo di collocamento, inserite un nome descrittivo per il gruppo di collocamento, quindi per la strategia del gruppo di collocamento, selezionate cluster.
   - Assicurati di abilitare «Elastic Fabric Adapter» in questa pagina. Se questa opzione è disabilitata, modificare la subnet in una che supporta il tipo di istanza selezionato.
   - Nella sezione Network Interfaces (Interfacce di rete), per il dispositivo eth0 scegliere New network interface (Nuova interfaccia di rete). Facoltativamente, puoi specificare un IPv4 indirizzo principale e uno o più IPv4 indirizzi secondari. Se stai avviando l'istanza in una sottorete a cui è associato un IPv6 CIDR blocco, puoi facoltativamente specificare un IPv6 indirizzo principale e uno o più indirizzi secondari. IPv6
   - Scegli Passaggio successivo: aggiunta dello storage.
6. Nella pagina Aggiungi spazio di archiviazione, specifica i volumi da collegare alle istanze oltre ai volumi specificati da AMI (ad esempio il volume del dispositivo root), quindi scegli Avanti: Aggiungi tag.

7. Nella pagina Add Tags (Aggiungi tag) specificare i tag per l'istanza, ad esempio un nome intuitivo, quindi selezionare Next: Configure Security Group (Successivo: configurazione del gruppo di sicurezza).
8. Nella pagina Configura gruppo di sicurezza, per Assegna un gruppo di sicurezza, seleziona Seleziona un gruppo di sicurezza esistente, quindi seleziona il gruppo di sicurezza creato in precedenza.
9. Scegliere Review and Launch (Analizza e avvia).
10. Nella pagina Review Instance Launch (Verifica avvio istanza) controllare le impostazioni e selezionare Launch (Avvia) per scegliere una coppia di chiavi e avviare l'istanza.

Verifica allegato EFA

Dalla console

Dopo aver avviato l'istanza, controlla i dettagli dell'istanza nella AWS Console. Per fare ciò, seleziona l'istanza nella EC2 console e guarda la scheda Descrizione nel riquadro inferiore della pagina. Trova il parametro 'Interfacce di rete: eth0' e fai clic su eth0 che apre un pop-up. Assicurati che «Elastic Fabric Adapter» sia abilitato.

Se non EFA è abilitato, puoi risolvere il problema in uno dei seguenti modi:

- Chiusura dell'EC2istanza e avvio di una nuova istanza con gli stessi passaggi. Assicurati che EFA sia collegato.
- Collega EFA a un'istanza esistente.

  1. Nella EC2 console, vai a Interfacce di rete.
  2. Fai clic su Create a Network Interface (Crea un'interfaccia di rete).
  3. Seleziona la stessa subnet in cui si trova l'istanza.
  4. Assicurati di abilitare «Elastic Fabric Adapter» e fai clic su Crea.
  5. Torna alla scheda EC2 Istanze e seleziona la tua istanza.
  6. Vai a Azioni: Stato dell'istanza e interrompi l'istanza prima di EFA collegarla.
  7. Da Actions (Operazioni), seleziona Networking: Attach Network Interface (Rete: Collega interfaccia di rete).
  8. Seleziona l'interfaccia appena creata e clicca su attach (collega).
  9. Riavviare l'istanza.

Dall'istanza

Il seguente script di test è già presente suDLAMI. Eseguilo per assicurarti che i moduli del kernel siano caricati correttamente.

```
$ fi_info -p efa
```

L'aspetto dell'output sarà simile al seguente.

```
provider: efa
    fabric: EFA-fe80::e5:56ff:fe34:56a8
    domain: efa_0-rdm
    version: 2.0
    type: FI_EP_RDM
    protocol: FI_PROTO_EFA
provider: efa
    fabric: EFA-fe80::e5:56ff:fe34:56a8
    domain: efa_0-dgrm
    version: 2.0
    type: FI_EP_DGRAM
    protocol: FI_PROTO_EFA
provider: efa;ofi_rxd
    fabric: EFA-fe80::e5:56ff:fe34:56a8
    domain: efa_0-dgrm
    version: 1.0
    type: FI_EP_RDM
    protocol: FI_PROTO_RXD
```

Verifica della configurazione del gruppo di sicurezza

Il seguente script di test è già presente inDLAMI. Eseguilo per assicurarti che il gruppo di sicurezza creato sia configurato correttamente.

```
$ cd /opt/amazon/efa/test/
$ ./efa_test.sh
```

L'aspetto dell'output sarà simile al seguente.

```
Starting server...
Starting client...
bytes   #sent   #ack     total       time     MB/sec    usec/xfer   Mxfers/sec
```

```
64      10      =10     1.2k        0.02s      0.06    1123.55       0.00
256     10      =10     5k          0.00s     17.66      14.50       0.07
1k      10      =10     20k         0.00s     67.81      15.10       0.07
4k      10      =10     80k         0.00s    237.45      17.25       0.06
64k     10      =10     1.2m        0.00s    921.10      71.15       0.01
1m      10      =10     20m         0.01s   2122.41     494.05       0.00
```

Se smette di rispondere o non viene completato, assicurati che il tuo gruppo di sicurezza disponga delle regole in entrata/uscita corrette.

Utilizzo su EFA DLAMI

La sezione seguente descrive come utilizzare per EFA eseguire applicazioni multinodo su AWS Deep Learning AMIs.

Esecuzione di applicazioni a più nodi con EFA

Per eseguire un'applicazione su un cluster di nodi è richiesta la seguente configurazione

Argomenti

- Abilita Passwordless SSH
- Creazione di file hosts
- NCCLTest

Abilita Passwordless SSH

Seleziona un nodo nel cluster come il nodo principale. I nodi rimanenti sono indicati come nodi membro.

1. Sul nodo leader, genera la coppia di chiavi. RSA

   ```
   ssh-keygen -t rsa -N "" -f ~/.ssh/id_rsa
   ```

2. Modifica le autorizzazioni della chiave privata sul nodo principale.

   ```
   chmod 600 ~/.ssh/id_rsa
   ```

3. Copia la chiave `~/.ssh/id_rsa.pub` pubblica e aggiungila a uno `~/.ssh/authorized_keys` dei nodi membri del cluster.

4. Puoi ora accedere direttamente ai nodi membro dal nodo principale utilizzando l'IP privato.

```
ssh <member private ip>
```

5. Disabilita strictHostKey Checking e abilita l'inoltro degli agenti sul nodo leader aggiungendo quanto segue al file ~/.ssh/config sul nodo leader:

```
Host *
    ForwardAgent yes
Host *
    StrictHostKeyChecking no
```

6. Nelle istanze Amazon Linux 2, esegui il seguente comando sul nodo leader per fornire le autorizzazioni corrette al file di configurazione:

```
chmod 600 ~/.ssh/config
```

Creazione di file hosts

Nel nodo principale, creare un file hosts per identificare i nodi nel cluster. Il file hosts deve contenere una voce per ogni nodo del cluster. Crea un file ~/hosts e aggiungi ogni nodo utilizzando l'IP privato come riportato di seguito:

```
localhost slots=8
<private ip of node 1> slots=8
<private ip of node 2> slots=8
```

NCCLTest

> **Note**
>
> Questi test sono stati eseguiti utilizzando la EFA versione 1.30.0 e il OFI NCCL Plugin 1.7.4.

Di seguito sono elencati un sottoinsieme di NCCL test forniti da Nvidia per testare funzionalità e prestazioni su più nodi di elaborazione

Istanze supportate: P3dn, P4, P5

Test di funzionalità

NCCLTest multinodo per il trasferimento di messaggi

Il nccl_message_transfer è un semplice test per garantire che il plugin funzioni come previsto. NCCL OFI Il test convalida la funzionalità della creazione della connessione e del trasferimento dei dati. NCCL APIs Assicurati di utilizzare il percorso completo di mpirun come mostrato nell'esempio mentre NCCL esegui applicazioni con. EFA Modifica i parametri in N base al numero di istanze np e al numero di istanze presenti nel cluster. GPUs Per ulteriori informazioni, consultare la .AWS OFINCCLdocumentazione.

Il seguente test nccl_message_transfer è per una versione xx.x generica. CUDA Puoi eseguire i comandi per qualsiasi CUDA versione disponibile nella tua EC2 istanza Amazon sostituendo la CUDA versione nello script.

```
$/opt/amazon/openmpi/bin/mpirun -n 2 -N 1 --hostfile hosts \
-x LD_LIBRARY_PATH=/usr/local/cuda-xx.x/efa/lib:/usr/local/cuda-xx.x/lib:/usr/
local/cuda-xx.x/lib64:/usr/local/cuda-xx.x:$LD_LIBRARY_PATH \
--mca btl tcp,self --mca btl_tcp_if_exclude lo,docker0 --bind-to none \
opt/aws-ofi-nccl/tests/nccl_message_transfer
```

L'aspetto dell'output sarà simile al seguente. Puoi controllare l'output per vedere che EFA viene utilizzato come OFI provider.

```
INFO: Function: nccl_net_ofi_init Line: 1069: NET/OFI Selected Provider is efa (found 4
 nics)
INFO: Function: nccl_net_ofi_init Line: 1160: NET/OFI Using transport protocol SENDRECV
INFO: Function: configure_ep_inorder Line: 261: NET/OFI Setting
 FI_OPT_EFA_SENDRECV_IN_ORDER_ALIGNED_128_BYTES not supported.
INFO: Function: configure_nccl_proto Line: 227: NET/OFI Setting NCCL_PROTO to "simple"
INFO: Function: main Line: 86: NET/OFI Process rank 1 started. NCCLNet device used on
 ip-172-31-13-179 is AWS Libfabric.
INFO: Function: main Line: 91: NET/OFI Received 4 network devices
INFO: Function: main Line: 111: NET/OFI Network supports communication using CUDA
 buffers. Dev: 3
INFO: Function: main Line: 118: NET/OFI Server: Listening on dev 3
INFO: Function: main Line: 131: NET/OFI Send connection request to rank 1
INFO: Function: main Line: 173: NET/OFI Send connection request to rank 0
INFO: Function: main Line: 137: NET/OFI Server: Start accepting requests
INFO: Function: main Line: 141: NET/OFI Successfully accepted connection from rank 1
INFO: Function: main Line: 145: NET/OFI Send 8 requests to rank 1
INFO: Function: main Line: 179: NET/OFI Server: Start accepting requests
```

```
INFO: Function: main Line: 183: NET/OFI Successfully accepted connection from rank 0
INFO: Function: main Line: 187: NET/OFI Rank 1 posting 8 receive buffers
INFO: Function: main Line: 161: NET/OFI Successfully sent 8 requests to rank 1
INFO: Function: main Line: 251: NET/OFI Got completions for 8 requests for rank 0
INFO: Function: main Line: 251: NET/OFI Got completions for 8 requests for rank 1
```

Test delle prestazioni

Test NCCL delle prestazioni multinodo su P4D.24XLarge

Per verificare NCCL le prestazioni conEFA, esegui il test NCCL delle prestazioni standard disponibile nel repository ufficiale NCCL-Tests. DLAMIViene fornito con questo test già creato per CUDA XX.X. Allo stesso modo puoi eseguire il tuo script con. EFA

Quando costruisci il tuo script, fai riferimento alla seguente guida:

- Usa il percorso completo di mpirun come mostrato nell'esempio mentre esegui applicazioni con. NCCL EFA
- Modifica i parametri np e N in base al numero di istanze e al tuo cluster. GPUs
- Aggiungi il INFO flag NCCL _ DEBUG = e assicurati che i log indichino l'EFAutilizzo come «Il provider selezionato è». EFA
- Imposta la posizione del registro di addestramento da analizzare per la convalida

```
TRAINING_LOG="testEFA_$(date +"%N").log"
```

Utilizzate il comando `watch nvidia-smi` su uno qualsiasi dei nodi membri per monitorare GPU l'utilizzo. I `watch nvidia-smi` comandi seguenti si riferiscono a una versione CUDA xx.x generica e dipendono dal sistema operativo dell'istanza. Puoi eseguire i comandi per qualsiasi CUDA versione disponibile nella tua EC2 istanza Amazon sostituendo la CUDA versione nello script.

- Amazon Linux 2:

```
 $ /opt/amazon/openmpi/bin/mpirun -n 16 -N 8 \
-x NCCL_DEBUG=INFO --mca pml ^cm \
-x LD_LIBRARY_PATH=/usr/local/cuda-xx.x/efa/lib:/usr/local/cuda-xx.x/lib:/usr/
local/cuda-xx.x/lib64:/usr/local/cuda-xx.x:/opt/amazon/efa/lib64:/opt/amazon/openmpi/
lib64:$LD_LIBRARY_PATH \
--hostfile hosts --mca btl tcp,self --mca btl_tcp_if_exclude lo,docker0 --bind-to
 none \
```

```
/usr/local/cuda-xx.x/efa/test-cuda-xx.x/all_reduce_perf -b 8 -e 1G -f 2 -g 1 -c 1 -n
 100 | tee ${TRAINING_LOG}
```

- Ubuntu 20.04:

```
$ /opt/amazon/openmpi/bin/mpirun -n 16 -N 8 \
-x NCCL_DEBUG=INFO --mca pml ^cm \
-x LD_LIBRARY_PATH=/usr/local/cuda-xx.x/efa/lib:/usr/local/cuda-xx.x/lib:/usr/
local/cuda-xx.x/lib64:/usr/local/cuda-xx.x:/opt/amazon/efa/lib:/opt/amazon/openmpi/
lib:$LD_LIBRARY_PATH \
--hostfile hosts --mca btl tcp,self --mca btl_tcp_if_exclude lo,docker0 --bind-to
 none \
/usr/local/cuda-xx.x/efa/test-cuda-xx.x/all_reduce_perf -b 8 -e 1G -f 2 -g 1 -c 1 -n
 100 | tee ${TRAINING_LOG}
```

L'aspetto dell'output deve essere simile al seguente:

```
# nThread 1 nGpus 1 minBytes 8 maxBytes 1073741824 step: 2(factor) warmup iters: 5
 iters: 100 agg iters: 1 validation: 1 graph: 0
#
# Using devices
#  Rank  0 Group  0 Pid   9591 on ip-172-31-4-37 device  0 [0x10] NVIDIA A100-SXM4-40GB
#  Rank  1 Group  0 Pid   9592 on ip-172-31-4-37 device  1 [0x10] NVIDIA A100-SXM4-40GB
#  Rank  2 Group  0 Pid   9593 on ip-172-31-4-37 device  2 [0x20] NVIDIA A100-SXM4-40GB
#  Rank  3 Group  0 Pid   9594 on ip-172-31-4-37 device  3 [0x20] NVIDIA A100-SXM4-40GB
#  Rank  4 Group  0 Pid   9595 on ip-172-31-4-37 device  4 [0x90] NVIDIA A100-SXM4-40GB
#  Rank  5 Group  0 Pid   9596 on ip-172-31-4-37 device  5 [0x90] NVIDIA A100-SXM4-40GB
#  Rank  6 Group  0 Pid   9597 on ip-172-31-4-37 device  6 [0xa0] NVIDIA A100-SXM4-40GB
#  Rank  7 Group  0 Pid   9598 on ip-172-31-4-37 device  7 [0xa0] NVIDIA A100-SXM4-40GB
#  Rank  8 Group  0 Pid  10216 on ip-172-31-13-179 device  0 [0x10] NVIDIA A100-
SXM4-40GB
#  Rank  9 Group  0 Pid  10217 on ip-172-31-13-179 device  1 [0x10] NVIDIA A100-
SXM4-40GB
#  Rank 10 Group  0 Pid  10218 on ip-172-31-13-179 device  2 [0x20] NVIDIA A100-
SXM4-40GB
#  Rank 11 Group  0 Pid  10219 on ip-172-31-13-179 device  3 [0x20] NVIDIA A100-
SXM4-40GB
#  Rank 12 Group  0 Pid  10220 on ip-172-31-13-179 device  4 [0x90] NVIDIA A100-
SXM4-40GB
#  Rank 13 Group  0 Pid  10221 on ip-172-31-13-179 device  5 [0x90] NVIDIA A100-
SXM4-40GB
```

```
#  Rank 14 Group  0 Pid  10222 on ip-172-31-13-179 device  6 [0xa0] NVIDIA A100-
SXM4-40GB
#  Rank 15 Group  0 Pid  10223 on ip-172-31-13-179 device  7 [0xa0] NVIDIA A100-
SXM4-40GB
ip-172-31-4-37:9591:9591 [0] NCCL INFO Bootstrap : Using ens32:172.31.4.37
ip-172-31-4-37:9591:9591 [0] NCCL INFO NET/Plugin: Failed to find ncclCollNetPlugin_v6
 symbol.
ip-172-31-4-37:9591:9591 [0] NCCL INFO NET/Plugin: Failed to find ncclCollNetPlugin
 symbol (v4 or v5).
ip-172-31-4-37:9591:9591 [0] NCCL INFO cudaDriverVersion 12020
NCCL version 2.18.5+cuda12.2
...
ip-172-31-4-37:9024:9062 [6] NCCL INFO NET/OFI Initializing aws-ofi-nccl 1.7.4-aws
ip-172-31-4-37:9020:9063 [2] NCCL INFO NET/OFI Using CUDA runtime version 11070
ip-172-31-4-37:9020:9063 [2] NCCL INFO NET/OFI Configuring AWS-specific options
ip-172-31-4-37:9024:9062 [6] NCCL INFO NET/OFI Using CUDA runtime version 11070
ip-172-31-4-37:9024:9062 [6] NCCL INFO NET/OFI Configuring AWS-specific options
ip-172-31-4-37:9024:9062 [6] NCCL INFO NET/OFI Setting provider_filter to efa
ip-172-31-4-37:9024:9062 [6] NCCL INFO NET/OFI Setting FI_EFA_FORK_SAFE environment
 variable to 1
ip-172-31-4-37:9024:9062 [6] NCCL INFO NET/OFI Disabling NVLS support due to NCCL
 version 21602
ip-172-31-4-37:9020:9063 [2] NCCL INFO NET/OFI Setting provider_filter to efa
ip-172-31-4-37:9020:9063 [2] NCCL INFO NET/OFI Setting FI_EFA_FORK_SAFE environment
 variable to 1
ip-172-31-4-37:9020:9063 [2] NCCL INFO NET/OFI Disabling NVLS support due to NCCL
 version 21602
ip-172-31-4-37:9020:9063 [2] NCCL INFO NET/OFI Running on p4d.24xlarge platform,
 Setting NCCL_TOPO_FILE environment variable to /opt/aws-ofi-nccl/share/aws-ofi-nccl/
xml/p4d-24xl-topo.xml
...
------------------------------some output truncated----------------------------------
#                                                              out-of-place
          in-place
#       size         count      type   redop    root     time   algbw   busbw #wrong
  time   algbw   busbw #wrong
#        (B)    (elements)                               (us)  (GB/s)  (GB/s)
  (us)  (GB/s)  (GB/s)
           0             0     float     sum      -1    11.02    0.00    0.00      0
 11.04    0.00    0.00      0
           0             0     float     sum      -1    11.01    0.00    0.00      0
 11.00    0.00    0.00      0
           0             0     float     sum      -1    11.02    0.00    0.00      0
 11.02    0.00    0.00      0
```

```
           0             0     float     sum      -1    11.01    0.00    0.00       0
11.00    0.00    0.00       0
           0             0     float     sum      -1    11.02    0.00    0.00       0
11.02    0.00    0.00       0
         256             4     float     sum      -1    632.7    0.00    0.00       0
628.2    0.00    0.00       0
         512             8     float     sum      -1    627.4    0.00    0.00       0
629.6    0.00    0.00       0
        1024            16     float     sum      -1    632.2    0.00    0.00       0
631.7    0.00    0.00       0
        2048            32     float     sum      -1    631.0    0.00    0.00       0
634.2    0.00    0.00       0
        4096            64     float     sum      -1    623.3    0.01    0.01       0
633.6    0.01    0.01       0
        8192           128     float     sum      -1    635.1    0.01    0.01       0
633.5    0.01    0.01       0
       16384           256     float     sum      -1    634.8    0.03    0.02       0
637.0    0.03    0.02       0
       32768           512     float     sum      -1    647.9    0.05    0.05       0
636.8    0.05    0.05       0
       65536          1024     float     sum      -1    658.9    0.10    0.09       0
667.0    0.10    0.09       0
      131072          2048     float     sum      -1    671.9    0.20    0.18       0
662.9    0.20    0.19       0
      262144          4096     float     sum      -1    692.1    0.38    0.36       0
685.1    0.38    0.36       0
      524288          8192     float     sum      -1    715.3    0.73    0.69       0
696.6    0.75    0.71       0
     1048576         16384     float     sum      -1    734.6    1.43    1.34       0
729.2    1.44    1.35       0
     2097152         32768     float     sum      -1    785.9    2.67    2.50       0
794.5    2.64    2.47       0
     4194304         65536     float     sum      -1    837.2    5.01    4.70       0
837.6    5.01    4.69       0
     8388608        131072     float     sum      -1    929.2    9.03    8.46       0
931.4    9.01    8.44       0
    16777216        262144     float     sum      -1   1773.6    9.46    8.87       0
1772.8    9.46    8.87       0
    33554432        524288     float     sum      -1   2110.2   15.90   14.91       0
2116.1   15.86   14.87       0
    67108864       1048576     float     sum      -1   2650.9   25.32   23.73       0
2658.1   25.25   23.67       0
   134217728       2097152     float     sum      -1   3943.1   34.04   31.91       0
3945.9   34.01   31.89       0
```

```
   268435456       4194304     float     sum      -1   7216.5   37.20   34.87      0
 7178.6   37.39   35.06      0
   536870912       8388608     float     sum      -1    13680   39.24   36.79      0
 13676   39.26   36.80      0
[  1073741824      16777216     float     sum      -1    25645   41.87   39.25      0
 25497   42.11   39.48      0 ] <- Used For Benchmark
...
# Out of bounds values : 0 OK
# Avg bus bandwidth    : 7.46044
```

Test di convalida

Per verificare che i EFA test abbiano restituito un risultato valido, utilizza i seguenti test per confermare:

- Ottieni il tipo di istanza utilizzando EC2 Instance Metadata:

```
TOKEN=$(curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-
token-ttl-seconds: 21600")
INSTANCE_TYPE=$(curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/
latest/meta-data/instance-type)
```

- Eseguire Test delle prestazioni
- Imposta i seguenti parametri

```
CUDA_VERSION
CUDA_RUNTIME_VERSION
NCCL_VERSION
```

- Convalida i risultati come mostrato:

```
RETURN_VAL=`echo $?`
if [ ${RETURN_VAL} -eq 0 ]; then

    # Information on how the version come from logs
    #
    # ip-172-31-27-205:6427:6427 [0] NCCL INFO cudaDriverVersion 12020
    # NCCL version 2.16.2+cuda11.8
    # ip-172-31-27-205:6427:6820 [0] NCCL INFO NET/OFI Initializing aws-ofi-nccl
 1.7.1-aws
    # ip-172-31-27-205:6427:6820 [0] NCCL INFO NET/OFI Using CUDA runtime version
 11060
```

```
    # cudaDriverVersion 12020  --> This is max supported cuda version by nvidia
 driver
    # NCCL version 2.16.2+cuda11.8 --> This is NCCL version compiled with cuda
 version
    # Using CUDA runtime version 11060 --> This is selected cuda version

    # Validation of logs
    grep "NET/OFI Using CUDA runtime version ${CUDA_RUNTIME_VERSION}" ${TRAINING_LOG}
 || { echo "Runtime cuda text not found"; exit 1; }
    grep "NET/OFI Initializing aws-ofi-nccl" ${TRAINING_LOG} || { echo "aws-ofi-nccl
 is not working, please check if it is installed correctly"; exit 1; }
    grep "NET/OFI Configuring AWS-specific options" ${TRAINING_LOG} || { echo "AWS-
specific options text not found"; exit 1; }
    grep "Using network AWS Libfabric" ${TRAINING_LOG} || { echo "AWS Libfabric text
 not found"; exit 1; }
    grep "busbw" ${TRAINING_LOG} || { echo "busbw text not found"; exit 1; }
    grep "Avg bus bandwidth " ${TRAINING_LOG} || { echo "Avg bus bandwidth text not
 found"; exit 1; }
    grep "NCCL version $NCCL_VERSION" ${TRAINING_LOG} || { echo "Text not found: NCCL
 version $NCCL_VERSION"; exit 1; }

    if [[ ${INSTANCE_TYPE} == "p4d.24xlarge" ]]; then
        grep "NET/AWS Libfabric/0/GDRDMA" ${TRAINING_LOG} || { echo "Text not found:
 NET/AWS Libfabric/0/GDRDMA"; exit 1; }
        grep "NET/OFI Selected Provider is efa (found 4 nics)" ${TRAINING_LOG} ||
 { echo "Selected Provider is efa text not found"; exit 1; }
        grep "aws-ofi-nccl/xml/p4d-24xl-topo.xml" ${TRAINING_LOG} || { echo "Topology
 file not found"; exit 1; }
    elif [[ ${INSTANCE_TYPE} == "p4de.24xlarge" ]]; then
        grep "NET/AWS Libfabric/0/GDRDMA" ${TRAINING_LOG} || { echo "Avg bus
 bandwidth text not found"; exit 1; }
        grep "NET/OFI Selected Provider is efa (found 4 nics)" ${TRAINING_LOG} ||
 { echo "Avg bus bandwidth text not found"; exit 1; }
        grep "aws-ofi-nccl/xml/p4de-24xl-topo.xml" ${TRAINING_LOG} || { echo
 "Topology file not found"; exit 1; }
    elif [[ ${INSTANCE_TYPE} == "p5.48xlarge" ]]; then
        grep "NET/AWS Libfabric/0/GDRDMA" ${TRAINING_LOG} || { echo "Avg bus
 bandwidth text not found"; exit 1; }
        grep "NET/OFI Selected Provider is efa (found 32 nics)" ${TRAINING_LOG} ||
 { echo "Avg bus bandwidth text not found"; exit 1; }
        grep "aws-ofi-nccl/xml/p5.48xl-topo.xml" ${TRAINING_LOG} || { echo "Topology
 file not found"; exit 1; }
    elif [[ ${INSTANCE_TYPE} == "p3dn.24xlarge" ]]; then
```

```
        grep "NET/OFI Selected Provider is efa (found 4 nics)" ${TRAINING_LOG} ||
 { echo "Selected Provider is efa text not found"; exit 1; }
    fi
    echo "*************************** check_efa_nccl_all_reduce passed for cuda
 version ${CUDA_VERSION} ***************************"
else
    echo "*************************** check_efa_nccl_all_reduce failed for cuda
 version ${CUDA_VERSION} ***************************"
fi
```

- Per accedere ai dati del benchmark, possiamo analizzare l'ultima riga della tabella in uscita dal test multinodo all_reduce:

```
benchmark=$(sudo cat ${TRAINING_LOG} | grep '1073741824' | tail -n1 | awk -F " "
 '{{print $12}}' | sed 's/ //' | sed  's/  5e-07//')
if [[ -z "${benchmark}" ]]; then
  echo "benchmark variable is empty"
  exit 1
fi

echo "Benchmark throughput: ${benchmark}"
```

GPUMonitoraggio e ottimizzazione

La sezione seguente ti guiderà attraverso le opzioni di GPU ottimizzazione e monitoraggio. Questa sezione è organizzata come un flusso di lavoro tipico in cui il monitoraggio supervisiona la pre-elaborazione e il training.

Monitoraggio

Il tuo DLAMI viene fornito preinstallato con diversi strumenti GPU di monitoraggio. Questa guida fa anche riferimento a strumenti disponibili per scaricare e installare.

- GPUsMonitora con CloudWatch- un'utilità preinstallata che riporta le statistiche GPU di utilizzo ad Amazon CloudWatch.
- nvidia-smi CLI: un'utilità per monitorare l'utilizzo complessivo del calcolo e della memoria. GPU È preinstallato sul tuo AWS Deep Learning AMIs (DLAMI).
- NVMLLibreria C: basata su C per accedere direttamente API alle funzioni di GPU monitoraggio e gestione. Viene utilizzata da nvidia-smi CLI sotto il cofano ed è preinstallata sul tuo. DLAMI Dispone anche di associazioni Python e Perl per facilitare lo sviluppo in tali lingue. L'utilità gpumon.py preinstallata sul tuo computer utilizza il pacchetto pynvml di. DLAMI nvidia-ml-py
- NVIDIADCGM- Uno strumento di gestione dei cluster. Per informazioni su come installare e configurare questo strumento, visita la pagina per gli sviluppatori.

> **Tip**
>
> Consulta NVIDIA il blog degli sviluppatori per le ultime informazioni sull'utilizzo degli CUDA strumenti installati suDLAMI:
>
> - Monitoraggio TensorCore dell'utilizzo tramite Nsight IDE e nvprof.

GPUsMonitora con CloudWatch

Quando utilizzi your DLAMI with a, GPU potresti scoprire che stai cercando modi per monitorarne l'utilizzo durante l'addestramento o l'inferenza. Questo può essere utile per ottimizzare la data pipeline e regolare la rete di deep learning.

Esistono due modi per configurare le GPU metriche con: CloudWatch

- Configura le metriche con AWS CloudWatch agente (consigliato)
- Configura le metriche con lo script preinstallato `gpumon.py`

Configura le metriche con AWS CloudWatch agente (consigliato)

Integra il tuo DLAMI con l' CloudWatch agente unificato per configurare i GPU parametri e monitorare l'utilizzo dei GPU coprocessi nelle istanze accelerate di AmazonEC2.

Esistono quattro modi per configurare le metriche con: GPU DLAMI

- Configura GPU metriche minime

- Configura GPU metriche parziali
- Configura tutte le GPU metriche disponibili
- Configura GPU metriche personalizzate

Per informazioni sugli aggiornamenti e le patch di sicurezza, consulta Patch di sicurezza per AWS CloudWatch agente

Prerequisiti

Per iniziare, devi configurare le IAM autorizzazioni dell'EC2istanza Amazon che consentano all'istanza di inviare parametri a. CloudWatch Per i passaggi dettagliati, consulta Creare IAM ruoli e utenti da utilizzare con l' CloudWatch agente.

Configura GPU metriche minime

Configura GPU metriche minime utilizzando il `dlami-cloudwatch-agent@minimal` `systemd` servizio. Questo servizio configura le seguenti metriche:

- `utilization_gpu`
- `utilization_memory`

Puoi trovare il `systemd` servizio per le GPU metriche minime preconfigurate nella seguente posizione:

```
/opt/aws/amazon-cloudwatch-agent/etc/dlami-amazon-cloudwatch-agent-minimal.json
```

Abilita e avvia il `systemd` servizio con i seguenti comandi:

```
sudo systemctl enable dlami-cloudwatch-agent@minimal
sudo systemctl start dlami-cloudwatch-agent@minimal
```

Configura GPU metriche parziali

Configura le GPU metriche parziali utilizzando il `dlami-cloudwatch-agent@partial` `systemd` servizio. Questo servizio configura le seguenti metriche:

- `utilization_gpu`
- `utilization_memory`

- memory_total
- memory_used
- memory_free

Puoi trovare il systemd servizio per le GPU metriche preconfigurate parziali nella seguente posizione:

```
/opt/aws/amazon-cloudwatch-agent/etc/dlami-amazon-cloudwatch-agent-partial.json
```

Abilita e avvia il systemd servizio con i seguenti comandi:

```
sudo systemctl enable dlami-cloudwatch-agent@partial
sudo systemctl start dlami-cloudwatch-agent@partial
```

Configura tutte le GPU metriche disponibili

Configura tutte le GPU metriche disponibili utilizzando il dlami-cloudwatch-agent@all systemd servizio. Questo servizio configura le seguenti metriche:

- utilization_gpu
- utilization_memory
- memory_total
- memory_used
- memory_free
- temperature_gpu
- power_draw
- fan_speed
- pcie_link_gen_current
- pcie_link_width_current
- encoder_stats_session_count
- encoder_stats_average_fps
- encoder_stats_average_latency
- clocks_current_graphics
- clocks_current_sm

- `clocks_current_memory`
- `clocks_current_video`

Puoi trovare il `systemd` servizio per tutte le GPU metriche preconfigurate disponibili nella seguente posizione:

```
/opt/aws/amazon-cloudwatch-agent/etc/dlami-amazon-cloudwatch-agent-all.json
```

Abilita e avvia il `systemd` servizio con i seguenti comandi:

```
sudo systemctl enable dlami-cloudwatch-agent@all
sudo systemctl start dlami-cloudwatch-agent@all
```

Configura GPU metriche personalizzate

Se le metriche preconfigurate non soddisfano i requisiti, puoi creare un file di configurazione CloudWatch dell'agente personalizzato.

Crea un file di configurazione personalizzato

Per creare un file di configurazione personalizzato, consulta i passaggi dettagliati in Creare o modificare manualmente il file di configurazione dell' CloudWatch agente.

Per questo esempio, supponiamo che la definizione dello schema si trovi in`/opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.json`.

Configura le metriche con il tuo file personalizzato

Esegui il comando seguente per configurare l' CloudWatch agente in base al tuo file personalizzato:

```
sudo /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl \
-a fetch-config -m ec2 -s -c \
file:/opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.json
```

Patch di sicurezza per AWS CloudWatch agente

Le nuove versioni DLAMIs sono configurate con le ultime versioni disponibili AWS CloudWatch patch di sicurezza per agenti. Consultate le seguenti sezioni per aggiornare la versione corrente DLAMI con le patch di sicurezza più recenti, a seconda del sistema operativo scelto.

Amazon Linux 2

yumUsalo per ricevere le ultime novità AWS CloudWatch patch di sicurezza per agenti per Amazon Linux 2DLAMI.

```
sudo yum update
```

Ubuntu

Per ricevere le ultime novità AWS CloudWatch patch di sicurezza per un DLAMI con Ubuntu, è necessario reinstallare il AWS CloudWatch agente che utilizza un link per il download di Amazon S3.

```
wget https://s3.region.amazonaws.com/amazoncloudwatch-agent-region/ubuntu/arm64/latest/
amazon-cloudwatch-agent.deb
```

Per ulteriori informazioni sull'installazione di AWS CloudWatch agente che utilizza i link per il download di Amazon S3, consulta Installazione ed esecuzione dell' CloudWatch agente sui server.

Configura le metriche con lo script preinstallato **`gpumon.py`**

Un'utilità chiamata gpumon.py è preinstallata sul tuo. DLAMI Si integra CloudWatch e supporta il monitoraggio del periodo di GPU utilizzo: GPU memoria, GPU temperatura e GPU alimentazione. Lo script invia periodicamente i dati monitorati a CloudWatch. È possibile configurare il livello di granularità dei dati a cui vengono inviati CloudWatch modificando alcune impostazioni nello script. Prima di avviare lo script, tuttavia, è necessario configurarlo per CloudWatch ricevere le metriche.

Come configurare ed eseguire il GPU monitoraggio con CloudWatch

1. Crea un IAM utente o modificane uno esistente per avere una politica su cui pubblicare la metrica. CloudWatch Se crei un nuovo utente, prendi nota delle credenziali poiché saranno necessarie nella fase successiva.

   La IAM policy da cercare è «cloudwatch:». PutMetricData La policy che viene aggiunta è la seguente:

   ```
   {
      "Version": "2012-10-17",
      "Statement": [
           {
               "Action": [
   ```

```
                "cloudwatch:PutMetricData"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

> **Tip**
>
> Per ulteriori informazioni sulla creazione di un IAM utente e sull'aggiunta di policy per CloudWatch, consulta la CloudWatch documentazione.

2. Sul tuoDLAMI, corri AWS configura e specifica le credenziali IAM dell'utente.

```
$ aws configure
```

3. Potrebbe essere necessario apportare alcune modifiche all'utilità gpumon prima di eseguirla. È possibile trovare l'utilità gpumon e README nella posizione definita nel seguente blocco di codice. Per ulteriori informazioni sullo `gpumon.py` script, consulta la posizione dello script in Amazon S3.

```
Folder: ~/tools/GPUCloudWatchMonitor
Files:  ~/tools/GPUCloudWatchMonitor/gpumon.py
       ~/tools/GPUCloudWatchMonitor/README
```

Opzioni:

- Cambia la regione in gpumon.py se la tua istanza è NOT in us-east-1.
- Modifica altri parametri come il periodo CloudWatch `namespace` di riferimento con. `store_reso`

4. Attualmente lo script supporta solo Python 3. Attiva l'ambiente Python 3 del tuo framework preferito o attiva l'ambiente DLAMI Python 3 generale.

```
$ source activate python3
```

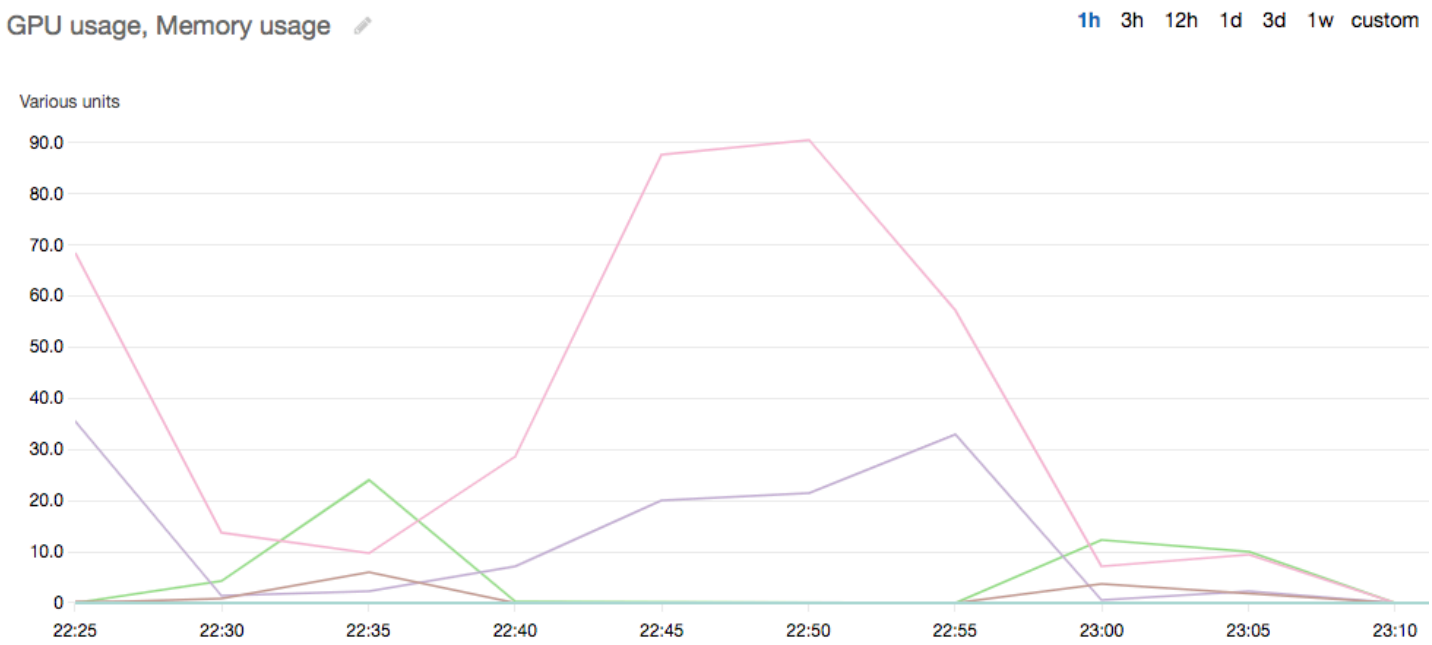
5. Esegui l'utilità gpumon in background.

```
(python3)$ python gpumon.py &
```

6. Apri il browser e https://console.aws.amazon.com/cloudwatch/seleziona la metrica. Avrà un namespace ". DeepLearningTrain

> **Tip**
>
> Puoi cambiare lo spazio dei nomi modificando gpumon.py. Puoi anche modificare l'intervallo di reporting regolando `store_reso`.

Di seguito è riportato un esempio di CloudWatch grafico che riporta un'esecuzione di gpumon.py che monitora un processo di formazione sull'istanza p2.8xlarge.



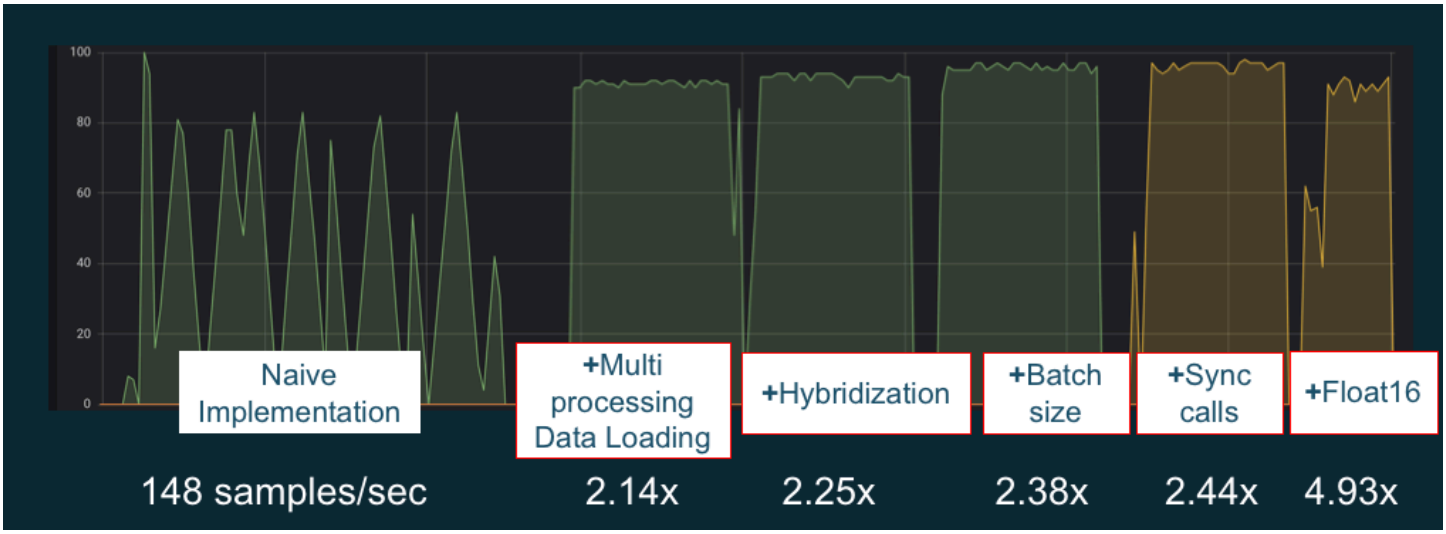
Potrebbero interessarti questi altri argomenti sul GPU monitoraggio e l'ottimizzazione:

- Monitoraggio
  - GPUsMonitora con CloudWatch
- Ottimizzazione
  - Pre-elaborazione
  - Addestramento

Ottimizzazione

Per sfruttare al meglio le tue potenzialitàGPUs, puoi ottimizzare la pipeline di dati e ottimizzare la tua rete di deep learning. Come descritto nella tabella seguente, un'implementazione ingenua o di base di una rete neurale potrebbe sfruttarne le potenzialità in GPU modo incoerente e non sfruttarne appieno le potenzialità. Ottimizzando la preelaborazione e il caricamento dei dati, è possibile ridurre il problema che comporta. CPU GPU Puoi regolare la rete neurale stessa, utilizzando l'ibridazione (quando supportata dal framework), modificando le dimensioni batch e sincronizzando le chiamate. Puoi anche utilizzare training a precisione multipla (float16 o int8) nella maggior parte dei framework, che può avere un effetto significativo sul miglioramento del throughput.

Il grafico seguente mostra i miglioramenti delle prestazioni cumulativi quando si applicano ottimizzazioni differenti. I risultati dipenderanno dai dati in corso di elaborazione e dalla rete che si sta ottimizzando.



Esempi di GPU ottimizzazioni delle prestazioni. Fonte del grafico: Performance Tricks with MXNet Gluon

Le seguenti guide introducono opzioni che funzioneranno con te DLAMI e ti aiuteranno a migliorare GPU le prestazioni.

Argomenti

- Pre-elaborazione
- Addestramento

Pre-elaborazione

La preelaborazione dei dati mediante trasformazioni o aumenti può spesso essere un processo CPU vincolato e questo può essere il collo di bottiglia nell'intera pipeline. I framework dispongono di operatori integrati per l'elaborazione delle immagini, ma DALI (Data Augmentation Library) dimostra prestazioni migliori rispetto alle opzioni integrate dei framework.

- NVIDIAData Augmentation Library ()DALI: trasferisce l'aumento dei dati su. DALI GPU Non è preinstallato suDLAMI, ma puoi accedervi installandolo o caricando un contenitore di framework supportato sulla tua DLAMI o su un'altra istanza Amazon Elastic Compute Cloud. Per i dettagli, consulta la pagina del DALI progetto sul NVIDIA sito Web. Per un esempio di caso d'uso e per scaricare esempi di codice, consultate l'esempio SageMaker Preprocessing Training Performance.
- nvJPEG: una libreria di decoder con GPU accelerazione per programmatori C. JPEG Supporta la decodifica di singole immagini o batch, nonché le successive operazioni di trasformazione comuni nell'apprendimento approfondito. nv è integrato oppure è possibile scaricarlo dalla pagina JPEG nvjpeg DALI del sito Web e utilizzarlo separatamente. NVIDIA

Potrebbero interessarti questi altri argomenti sul monitoraggio e l'ottimizzazione: GPU

- Monitoraggio
  - GPUsMonitora con CloudWatch
- Ottimizzazione
  - Pre-elaborazione
  - Addestramento

Addestramento

Grazie al training a precisione mista puoi distribuire reti più grandi con la stessa quantità di memoria o ridurre l'utilizzo della memoria rispetto alla rete a precisione singola o doppia, registrando al contempo un incremento delle prestazioni di calcolo. Hai anche il vantaggio di trasferimenti dati più piccoli e rapidi, un fattore importante nel training distribuito a più nodi. Per sfruttare il training a precisione mista occorre regolare casting dei dati e perdita di scaling. Le guide seguenti descrivono come eseguire questa operazione per i framework che supportano la precisione mista.

- NVIDIADeep Learning SDK: documenti sul NVIDIA sito Web che descrivono l'implementazione a precisione mista perMXNet, e. PyTorch TensorFlow

> **Tip**
>
> Assicurati di controllare il sito Web per il framework scelto e cerca "mixed precision" o "fp16" per le tecniche di ottimizzazione più recenti. Di seguito sono elencate alcune guide a precisione mista che possono essere utili:
>
> - Formazione a precisione mista con TensorFlow (video) - sul sito del blog. NVIDIA
> - Allenamento a precisione mista con float16 con MXNet - un articolo sul sito web. FAQ MXNet
> - NVIDIAApex: uno strumento per un facile allenamento a precisione mista con PyTorch - un articolo di blog sul sito web. NVIDIA

Potrebbero interessarti questi altri argomenti sul GPU monitoraggio e l'ottimizzazione:

- Monitoraggio
  - GPUsMonitora con CloudWatch
- Ottimizzazione
  - Pre-elaborazione
  - Addestramento

Il AWS Chip Inferentia con DLAMI

AWS Inferentia è un chip di apprendimento automatico personalizzato progettato da AWS che puoi utilizzare per previsioni di inferenza ad alte prestazioni. Per utilizzare il chip, configura un'istanza Amazon Elastic Compute Cloud e utilizza il AWS Kit di sviluppo software Neuron (SDK) per richiamare il chip Inferentia. Per offrire ai clienti la migliore esperienza con Inferentia, Neuron è stato integrato nel AWS Deep Learning AMIs (DLAMI).

I seguenti argomenti mostrano come iniziare a utilizzare Inferentia con. DLAMI

Indice

Avvio di un'istanza con DLAMI AWS Neuron

L'ultima DLAMI è pronta per l'uso con AWS Inferentia e viene fornito con AWS Pacchetto NeuronAPI. Per avviare un'DLAMIistanza, vedi Avvio e configurazione di un. DLAMI Dopo aver installato unDLAMI, segui i passaggi seguenti per assicurarti che AWS Chip Inferentia e AWS Le risorse neuronali sono attive.

Indice

Verifica la tua istanza

Prima di utilizzare l'istanza, verifica che sia correttamente configurata e configurata con Neuron.

Identificazione AWS Dispositivi di inferenza

Per identificare il numero di dispositivi Inferentia sulla tua istanza, usa il seguente comando:

```
neuron-ls
```

Se all'istanza sono collegati dispositivi Inferentia, l'output sarà simile al seguente:

```
+--------+--------+--------+-----------+--------------+
| NEURON | NEURON | NEURON | CONNECTED |     PCI      |
| DEVICE | CORES  | MEMORY |  DEVICES  |     BDF      |
+--------+--------+--------+-----------+--------------+
| 0      | 4      | 8 GB   | 1         | 0000:00:1c.0 |
| 1      | 4      | 8 GB   | 2, 0      | 0000:00:1d.0 |
| 2      | 4      | 8 GB   | 3, 1      | 0000:00:1e.0 |
| 3      | 4      | 8 GB   | 2         | 0000:00:1f.0 |
+--------+--------+--------+-----------+--------------+
```

L'output fornito è tratto da un'istanza INF1.6xLarge e include le seguenti colonne:

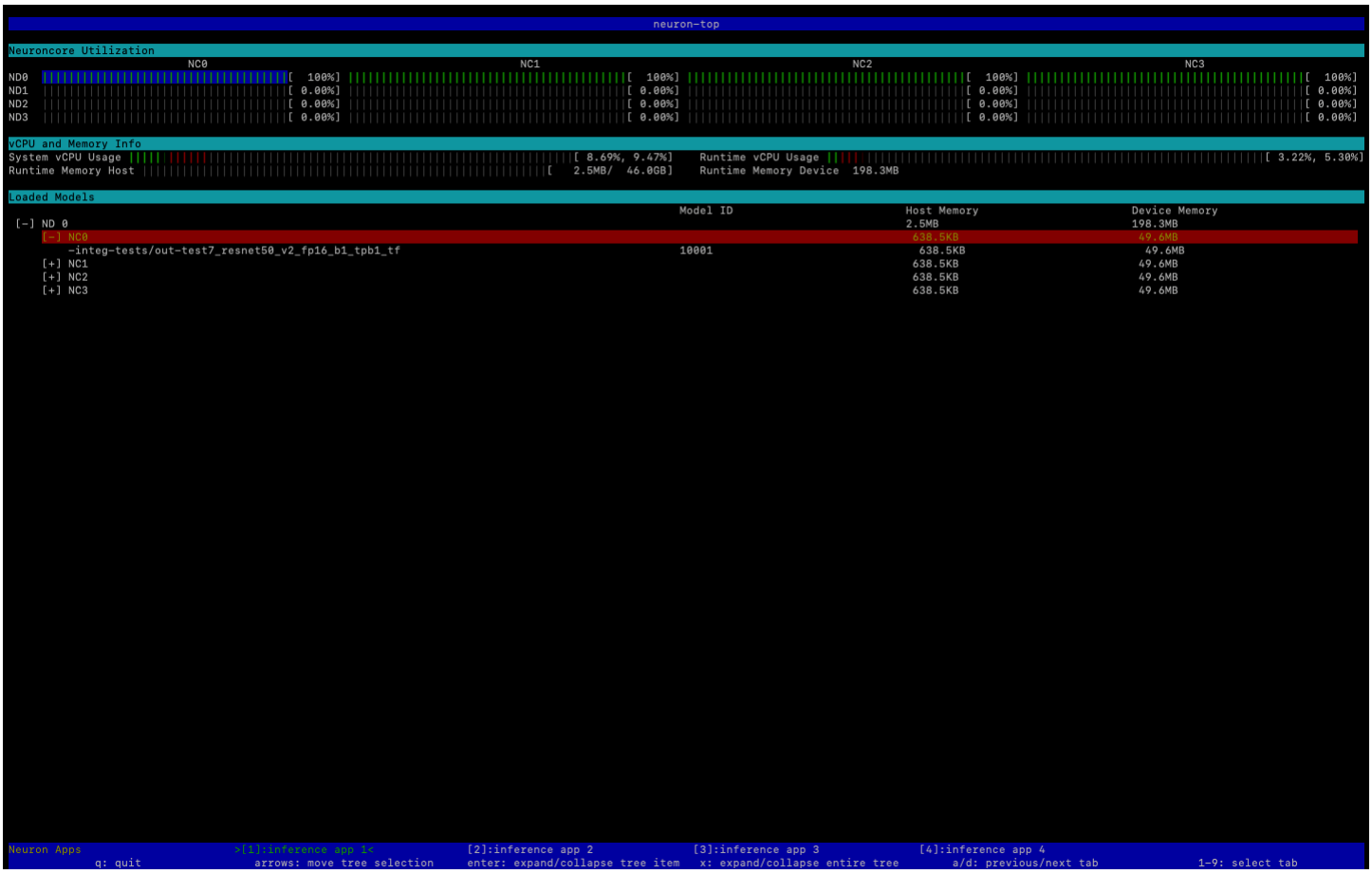
- NEURONDEVICE: L'ID logico assegnato a. NeuronDevice Questo ID viene utilizzato quando si configurano più runtime per utilizzarne diversi. NeuronDevices
- NEURONCORES: Il numero di NeuronCores presenti in. NeuronDevice
- NEURONMEMORY: La quantità di DRAM memoria contenuta in NeuronDevice.
- CONNECTEDDEVICES: Altro NeuronDevices collegato a NeuronDevice.
- PCIBDF: L'ID della funzione del dispositivo PCI bus (BDF) di NeuronDevice.

Visualizza l'utilizzo delle risorse

Visualizza informazioni utili sull'CPUutilizzo di NeuronCore and v, sull'utilizzo della memoria, sui modelli caricati e sulle applicazioni Neuron con il `neuron-top` comando. L'avvio `neuron-top` senza argomenti mostrerà i dati per tutte le applicazioni di machine learning che utilizzano. NeuronCores

```
neuron-top
```

Quando un'applicazione ne utilizza quattro NeuronCores, l'output dovrebbe essere simile all'immagine seguente:



Per ulteriori informazioni sulle risorse per monitorare e ottimizzare le applicazioni di inferenza basate su Neuron, consulta Neuron Tools.

Utilizzo di Neuron Monitor (neuron-monitor)

Neuron Monitor raccoglie le metriche dai runtime di Neuron in esecuzione sul sistema e trasmette i dati raccolti a stdout in formato. JSON Queste metriche sono organizzate in gruppi di metriche che puoi configurare fornendo un file di configurazione. Per ulteriori informazioni su Neuron Monitor, consulta la Guida per l'utente di Neuron Monitor.

Aggiornamento del software Neuron

Per informazioni su come aggiornare il SDK software Neuron all'interno, consulta il DLAMI AWS Guida alla configurazione di Neuron.

Fase succcessiva

Usare il con DLAMI AWS Neuron

Usare il con DLAMI AWS Neuron

Un tipico flusso di lavoro con AWS Neuron SDK consiste nel compilare un modello di machine learning precedentemente addestrato su un server di compilazione. Successivamente, distribuisci gli artefatti alle istanze Inf1 per l'esecuzione. AWS Deep Learning AMIs (DLAMI) viene preinstallato con tutto il necessario per compilare ed eseguire l'inferenza in un'istanza Inf1 che utilizza Inferentia.

Le sezioni seguenti descrivono come utilizzarlo con Inferentia. DLAMI

Indice

Usando TensorFlow -Neuron e il AWS Compilatore Neuron

Questo tutorial mostra come usare il AWS Compilatore Neuron per compilare il modello Keras ResNet -50 ed esportarlo come modello salvato in formato. SavedModel Questo formato è un tipico

formato intercambiabile tra modelli. TensorFlow Il tutorial illustra anche come eseguire l'inferenza su un'istanza di Inf1 con input di esempio.

Per ulteriori informazioni sul Neuron, vedere SDK il AWS Documentazione Neuron. SDK

Indice

Prerequisiti

Prima di utilizzare questo tutorial, è necessario aver completato la procedura di configurazione in Avvio di un'istanza con DLAMI AWS Neuron. Dovresti anche avere familiarità con il deep learning e l'uso di. DLAMI

Attivare l'ambiente Conda

Attiva l'ambiente TensorFlow -Neuron conda usando il seguente comando:

```
source activate aws_neuron_tensorflow_p36
```

Per uscire dall'ambiente Conda corrente, eseguire il comando seguente:

```
source deactivate
```

Compilazione Resnet50

Creare uno script Python chiamato **`tensorflow_compile_resnet50.py`** che abbia il seguente contenuto. Questo script Python compila il modello Keras ResNet 50 e lo esporta come modello salvato.

```
import os
import time
```

```
import shutil
import tensorflow as tf
import tensorflow.neuron as tfn
import tensorflow.compat.v1.keras as keras
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.applications.resnet50 import preprocess_input

# Create a workspace
WORKSPACE = './ws_resnet50'
os.makedirs(WORKSPACE, exist_ok=True)

# Prepare export directory (old one removed)
model_dir = os.path.join(WORKSPACE, 'resnet50')
compiled_model_dir = os.path.join(WORKSPACE, 'resnet50_neuron')
shutil.rmtree(model_dir, ignore_errors=True)
shutil.rmtree(compiled_model_dir, ignore_errors=True)

# Instantiate Keras ResNet50 model
keras.backend.set_learning_phase(0)
model = ResNet50(weights='imagenet')

# Export SavedModel
tf.saved_model.simple_save(
 session            = keras.backend.get_session(),
 export_dir         = model_dir,
 inputs             = {'input': model.inputs[0]},
 outputs            = {'output': model.outputs[0]})

# Compile using Neuron
tfn.saved_model.compile(model_dir, compiled_model_dir)

# Prepare SavedModel for uploading to Inf1 instance
shutil.make_archive(compiled_model_dir, 'zip', WORKSPACE, 'resnet50_neuron')
```

Compilare il modello utilizzando il seguente comando:

```
python tensorflow_compile_resnet50.py
```

Il processo di compilazione richiederà alcuni minuti. Al termine, l'output dovrebbe essere simile al seguente:

```
...
INFO:tensorflow:fusing subgraph neuron_op_d6f098c01c780733 with neuron-cc
INFO:tensorflow:Number of operations in TensorFlow session: 4638
INFO:tensorflow:Number of operations after tf.neuron optimizations: 556
INFO:tensorflow:Number of operations placed on Neuron runtime: 554
INFO:tensorflow:Successfully converted ./ws_resnet50/resnet50 to ./ws_resnet50/
resnet50_neuron
...
```

Dopo la compilazione, il modello salvato viene compresso a **`ws_resnet50/resnet50_neuron.zip`**. Decomprimere il modello e scaricare l'immagine di esempio per l'inferenza utilizzando i seguenti comandi:

```
unzip ws_resnet50/resnet50_neuron.zip -d .
curl -O https://raw.githubusercontent.com/awslabs/mxnet-model-server/master/docs/
images/kitten_small.jpg
```

ResNet50 Inferenza

Creare uno script Python chiamato **`tensorflow_infer_resnet50.py`** che abbia il seguente contenuto. Questo script esegue l'inferenza sul modello scaricato utilizzando un modello di inferenza precedentemente compilato.

```
import os
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications import resnet50

# Create input from image
img_sgl = image.load_img('kitten_small.jpg', target_size=(224, 224))
img_arr = image.img_to_array(img_sgl)
img_arr2 = np.expand_dims(img_arr, axis=0)
img_arr3 = resnet50.preprocess_input(img_arr2)
# Load model
COMPILED_MODEL_DIR = './ws_resnet50/resnet50_neuron/'
predictor_inferentia = tf.contrib.predictor.from_saved_model(COMPILED_MODEL_DIR)
# Run inference
model_feed_dict={'input': img_arr3}
```

```
infa_rslts = predictor_inferentia(model_feed_dict);
# Display results
print(resnet50.decode_predictions(infa_rslts["output"], top=5)[0])
```

Eseguire l'inferenza sul modello utilizzando il seguente comando:

```
python tensorflow_infer_resnet50.py
```

L'aspetto dell'output deve essere simile al seguente:

```
...
[('n02123045', 'tabby', 0.6918919), ('n02127052', 'lynx', 0.12770271), ('n02123159',
 'tiger_cat', 0.08277027), ('n02124075', 'Egyptian_cat', 0.06418919), ('n02128757',
 'snow_leopard', 0.009290541)]
```

Fase succcessiva

Utilizzo AWS Servizio di neuroni TensorFlow

Utilizzo AWS Servizio di neuroni TensorFlow

Questo tutorial mostra come costruire un grafico e aggiungere un AWS Fase di compilazione di Neuron prima di esportare il modello salvato da utilizzare con Serving. TensorFlow TensorFlow Serving è un sistema di servizio che consente di ampliare l'inferenza su una rete. Neuron TensorFlow Serving utilizza lo stesso API del normale Serving. TensorFlow L'unica differenza è che un modello salvato deve essere compilato per AWS Inferentia e il punto di ingresso hanno un nome binario diverso. `tensorflow_model_server_neuron` Il file binario si trova `/usr/local/bin/tensorflow_model_server_neuron` ed è preinstallato in. DLAMI

Per ulteriori informazioni sul NeuronSDK, vedere il AWS Documentazione Neuron. SDK

Indice

- Generare richieste di inferenza al server del modello

Prerequisiti

Prima di utilizzare questo tutorial, è necessario aver completato la procedura di configurazione in Avvio di un'istanza con DLAMI AWS Neuron. Dovresti anche avere familiarità con il deep learning e l'uso di. DLAMI

Attivare l'ambiente Conda

Attiva l'ambiente TensorFlow -Neuron conda usando il seguente comando:

```
source activate aws_neuron_tensorflow_p36
```

Se è necessario uscire dall'ambiente Conda corrente, eseguire:

```
source deactivate
```

Compilare ed esportare il modello salvato

Crea uno script Python chiamato `tensorflow-model-server-compile.py` con il seguente contenuto. Questo script costruisce un grafico e lo compila usando Neuron. Esporta quindi il grafico compilato come modello salvato.

```
import tensorflow as tf
import tensorflow.neuron
import os

tf.keras.backend.set_learning_phase(0)
model = tf.keras.applications.ResNet50(weights='imagenet')
sess = tf.keras.backend.get_session()
inputs = {'input': model.inputs[0]}
outputs = {'output': model.outputs[0]}

# save the model using tf.saved_model.simple_save
modeldir = "./resnet50/1"
tf.saved_model.simple_save(sess, modeldir, inputs, outputs)
```

```
# compile the model for Inferentia
neuron_modeldir = os.path.join(os.path.expanduser('~'), 'resnet50_inf1', '1')
tf.neuron.saved_model.compile(modeldir, neuron_modeldir, batch_size=1)
```

Compilare il modello utilizzando il seguente comando:

```
python tensorflow-model-server-compile.py
```

L'aspetto dell'output deve essere simile al seguente:

```
...
INFO:tensorflow:fusing subgraph neuron_op_d6f098c01c780733 with neuron-cc
INFO:tensorflow:Number of operations in TensorFlow session: 4638
INFO:tensorflow:Number of operations after tf.neuron optimizations: 556
INFO:tensorflow:Number of operations placed on Neuron runtime: 554
INFO:tensorflow:Successfully converted ./resnet50/1 to /home/ubuntu/resnet50_inf1/1
```

Servire il modello salvato

Una volta compilato il modello, è possibile utilizzare il seguente comando per servire il modello salvato con il binario tensorflow_model_server_neuron:

```
tensorflow_model_server_neuron --model_name=resnet50_inf1 \
    --model_base_path=$HOME/resnet50_inf1/ --port=8500 &
```

L'aspetto dell'output sarà simile al seguente. Il modello compilato viene inserito nel dispositivo Inferentia DRAM dal server per prepararsi all'inferenza.

```
...
2019-11-22 01:20:32.075856: I external/org_tensorflow/tensorflow/cc/saved_model/
loader.cc:311] SavedModel load for tags { serve }; Status: success. Took 40764
 microseconds.
2019-11-22 01:20:32.075888: I tensorflow_serving/servables/tensorflow/
saved_model_warmup.cc:105] No warmup data file found at /home/ubuntu/resnet50_inf1/1/
assets.extra/tf_serving_warmup_requests
2019-11-22 01:20:32.075950: I tensorflow_serving/core/loader_harness.cc:87]
 Successfully loaded servable version {name: resnet50_inf1 version: 1}
```

```
2019-11-22 01:20:32.077859: I tensorflow_serving/model_servers/
server.cc:353] Running gRPC ModelServer at 0.0.0.0:8500 ...
```

Generare richieste di inferenza al server del modello

Creare uno script Python chiamato `tensorflow-model-server-infer.py` con il seguente contenuto. Questo script esegue l'inferenza tramite gRPC, che è il framework di servizio.

```
import numpy as np
import grpc
import tensorflow as tf
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.resnet50 import preprocess_input
from tensorflow_serving.apis import predict_pb2
from tensorflow_serving.apis import prediction_service_pb2_grpc
from tensorflow.keras.applications.resnet50 import decode_predictions

if __name__ == '__main__':
    channel = grpc.insecure_channel('localhost:8500')
    stub = prediction_service_pb2_grpc.PredictionServiceStub(channel)
    img_file = tf.keras.utils.get_file(
        "./kitten_small.jpg",
        "https://raw.githubusercontent.com/awslabs/mxnet-model-server/master/docs/
images/kitten_small.jpg")
    img = image.load_img(img_file, target_size=(224, 224))
    img_array = preprocess_input(image.img_to_array(img)[None, ...])
    request = predict_pb2.PredictRequest()
    request.model_spec.name = 'resnet50_inf1'
    request.inputs['input'].CopyFrom(
        tf.contrib.util.make_tensor_proto(img_array, shape=img_array.shape))
    result = stub.Predict(request)
    prediction = tf.make_ndarray(result.outputs['output'])
    print(decode_predictions(prediction))
```

Esegui l'inferenza sul modello usando g RPC con il seguente comando:

```
python tensorflow-model-server-infer.py
```

L'aspetto dell'output deve essere simile al seguente:

```
[[('n02123045', 'tabby', 0.6918919), ('n02127052', 'lynx', 0.12770271), ('n02123159',
 'tiger_cat', 0.08277027), ('n02124075', 'Egyptian_cat', 0.06418919), ('n02128757',
 'snow_leopard', 0.009290541)]]
```

Usando MXNet -Neuron e il AWS Compilatore Neuron

La compilazione MXNet -Neuron API fornisce un metodo per compilare un grafico modello che è possibile eseguire su un AWS Dispositivo Inferentia.

In questo esempio, si utilizza API per compilare un modello ResNet -50 e lo si utilizza per eseguire l'inferenza.

Per ulteriori informazioni sul SDK Neuron, vedere il AWS Documentazione Neuron. SDK

Indice

Prerequisiti

Prima di utilizzare questo tutorial, è necessario aver completato la procedura di configurazione in Avvio di un'istanza con DLAMI AWS Neuron. Dovresti anche avere familiarità con il deep learning e l'uso di. DLAMI

Attivare l'ambiente Conda

Attiva l'ambiente MXNet -Neuron conda usando il seguente comando:

```
source activate aws_neuron_mxnet_p36
```

Per uscire dall'ambiente conda corrente, eseguire:

```
source deactivate
```

Compilazione Resnet50

Creare uno script Python chiamato **`mxnet_compile_resnet50.py`** con il seguente contenuto. Questo script utilizza la compilazione MXNet -Neuron API Python per compilare un modello -50. ResNet

```
import mxnet as mx
import numpy as np

print("downloading...")
path='http://data.mxnet.io/models/imagenet/'
mx.test_utils.download(path+'resnet/50-layers/resnet-50-0000.params')
mx.test_utils.download(path+'resnet/50-layers/resnet-50-symbol.json')
print("download finished.")

sym, args, aux = mx.model.load_checkpoint('resnet-50', 0)

print("compile for inferentia using neuron... this will take a few minutes...")
inputs = { "data" : mx.nd.ones([1,3,224,224], name='data', dtype='float32') }

sym, args, aux = mx.contrib.neuron.compile(sym, args, aux, inputs)

print("save compiled model...")
mx.model.save_checkpoint("compiled_resnet50", 0, sym, args, aux)
```

Compilare il modello utilizzando il seguente comando:

```
python mxnet_compile_resnet50.py
```

La compilazione richiederà alcuni minuti. Al termine della compilazione, i seguenti file si troveranno nella directory corrente:

```
resnet-50-0000.params
resnet-50-symbol.json
compiled_resnet50-0000.params
compiled_resnet50-symbol.json
```

ResNet50 Inferenza

Creare uno script Python chiamato **`mxnet_infer_resnet50.py`** con il seguente contenuto. Questo script scarica un'immagine di esempio e la usa per eseguire l'inferenza con il modello compilato.

```
import mxnet as mx
import numpy as np

path='http://data.mxnet.io/models/imagenet/'
mx.test_utils.download(path+'synset.txt')

fname = mx.test_utils.download('https://raw.githubusercontent.com/awslabs/mxnet-model-
server/master/docs/images/kitten_small.jpg')
img = mx.image.imread(fname)

# convert into format (batch, RGB, width, height)
img = mx.image.imresize(img, 224, 224)
# resize
img = img.transpose((2, 0, 1))
# Channel first
img = img.expand_dims(axis=0)
# batchify
img = img.astype(dtype='float32')

sym, args, aux = mx.model.load_checkpoint('compiled_resnet50', 0)
softmax = mx.nd.random_normal(shape=(1,))
args['softmax_label'] = softmax
args['data'] = img
# Inferentia context
ctx = mx.neuron()

exe = sym.bind(ctx=ctx, args=args, aux_states=aux, grad_req='null')
with open('synset.txt', 'r') as f:
    labels = [l.rstrip() for l in f]

exe.forward(data=img)
prob = exe.outputs[0].asnumpy()
# print the top-5
prob = np.squeeze(prob)
a = np.argsort(prob)[::-1]
for i in a[0:5]:
    print('probability=%f, class=%s' %(prob[i], labels[i]))
```

Eseguire l'inferenza con il modello compilato utilizzando il seguente comando:

```
python mxnet_infer_resnet50.py
```

L'aspetto dell'output deve essere simile al seguente:

```
probability=0.642454, class=n02123045 tabby, tabby cat
probability=0.189407, class=n02123159 tiger cat
probability=0.100798, class=n02124075 Egyptian cat
probability=0.030649, class=n02127052 lynx, catamount
probability=0.016278, class=n02129604 tiger, Panthera tigris
```

Fase succcessiva

Utilizzo di MXNet -Neuron Model Serving

Utilizzo di MXNet -Neuron Model Serving

In questo tutorial imparerai a utilizzare un MXNet modello pre-addestrato per eseguire la classificazione delle immagini in tempo reale con Multi Model Server (). MMS MMSè uno easy-to-use strumento flessibile per servire modelli di deep learning addestrati utilizzando qualsiasi framework di machine learning o deep learning. Questo tutorial include una fase di compilazione utilizzando AWS Neuron e un'implementazione dell'utilizzo. MMS MXNet

Per ulteriori informazioni sul NeuronSDK, vedere il AWS Documentazione Neuron. SDK

Indice

Prerequisiti

Prima di utilizzare questo tutorial, è necessario aver completato la procedura di configurazione in Avvio di un'istanza con DLAMI AWS Neuron. Dovresti anche avere familiarità con il deep learning e l'uso di. DLAMI

Attivare l'ambiente Conda

Attiva l'ambiente MXNet -Neuron conda utilizzando il seguente comando:

```
source activate aws_neuron_mxnet_p36
```

Per uscire dall'ambiente conda corrente, eseguire:

```
source deactivate
```

Scarica il codice di esempio

Per eseguire questo esempio, scaricare il codice di esempio utilizzando i seguenti comandi:

```
git clone https://github.com/awslabs/multi-model-server
cd multi-model-server/examples/mxnet_vision
```

Compila il modello

Creare uno script Python chiamato `multi-model-server-compile.py` con il seguente contenuto. Questo script compila il modello ResNet 50 nella destinazione del dispositivo Inferentia.

```
import mxnet as mx
from mxnet.contrib import neuron
import numpy as np

path='http://data.mxnet.io/models/imagenet/'
mx.test_utils.download(path+'resnet/50-layers/resnet-50-0000.params')
mx.test_utils.download(path+'resnet/50-layers/resnet-50-symbol.json')
mx.test_utils.download(path+'synset.txt')

nn_name = "resnet-50"

#Load a model
```

```
sym, args, auxs = mx.model.load_checkpoint(nn_name, 0)

#Define compilation parameters#  - input shape and dtype
inputs = {'data' : mx.nd.zeros([1,3,224,224], dtype='float32') }

# compile graph to inferentia target
csym, cargs, cauxs = neuron.compile(sym, args, auxs, inputs)

# save compiled model
mx.model.save_checkpoint(nn_name + "_compiled", 0, csym, cargs, cauxs)
```

Per compilare il modello, utilizzare il seguente comando:

```
python multi-model-server-compile.py
```

L'aspetto dell'output deve essere simile al seguente:

```
...
[21:18:40] src/nnvm/legacy_json_util.cc:209: Loading symbol saved by previous version
 v0.8.0. Attempting to upgrade...
[21:18:40] src/nnvm/legacy_json_util.cc:217: Symbol successfully upgraded!
[21:19:00] src/operator/subgraph/build_subgraph.cc:698: start to execute partition
 graph.
[21:19:00] src/nnvm/legacy_json_util.cc:209: Loading symbol saved by previous version
 v0.8.0. Attempting to upgrade...
[21:19:00] src/nnvm/legacy_json_util.cc:217: Symbol successfully upgraded!
```

Creare un file denominato `signature.json` con il seguente contenuto per configurare il nome e la forma di input:

```
{
  "inputs": [
    {
      "data_name": "data",
      "data_shape": [
        1,
        3,
        224,
        224
      ]
    }
  ]
```

```
}
```

Scaricare il file `synset.txt` utilizzando il comando seguente: Questo file è un elenco di nomi per ImageNet le classi di previsione.

```
curl -O https://s3.amazonaws.com/model-server/model_archive_1.0/examples/
squeezenet_v1.1/synset.txt
```

Creare una classe di servizio personalizzata seguendo il modello nella cartella `model_server_template`. Copiare il modello nella directory di lavoro corrente utilizzando il seguente comando:

```
cp -r ../model_service_template/* .
```

Modificare il modulo `mxnet_model_service.py` per sostituire il contesto `mx.cpu()` con il contesto `mx.neuron()` come segue. È inoltre necessario commentare la copia dei dati non necessaria `model_input` perché MXNet -Neuron non supporta and Gluon. NDArray APIs

```
...
self.mxnet_ctx = mx.neuron() if gpu_id is None else mx.gpu(gpu_id)
...
#model_input = [item.as_in_context(self.mxnet_ctx) for item in model_input]
```

Comprimere il modello con model-archiver utilizzando i seguenti comandi:

```
cd ~/multi-model-server/examples
model-archiver --force --model-name resnet-50_compiled --model-path mxnet_vision --
handler mxnet_vision_service:handle
```

Eseguire l’inferenza

Avvia il Multi Model Server e carica il modello che utilizza il RESTful API utilizzando i seguenti comandi. Assicurarsi che neuron-rtd sia in esecuzione con le impostazioni predefinite.

```
cd ~/multi-model-server/
multi-model-server --start --model-store examples > /dev/null # Pipe to log file if you
 want to keep a log of MMS
curl -v -X POST "http://localhost:8081/models?
initial_workers=1&max_workers=4&synchronous=true&url=resnet-50_compiled.mar"
```

```
sleep 10 # allow sufficient time to load model
```

Eseguire l'inferenza utilizzando un'immagine di esempio con i seguenti comandi:

```
curl -O https://raw.githubusercontent.com/awslabs/multi-model-server/master/docs/
images/kitten_small.jpg
curl -X POST http://127.0.0.1:8080/predictions/resnet-50_compiled -T kitten_small.jpg
```

L'aspetto dell'output deve essere simile al seguente:

```
[
  {
    "probability": 0.6388034820556641,
    "class": "n02123045 tabby, tabby cat"
  },
  {
    "probability": 0.16900072991847992,
    "class": "n02123159 tiger cat"
  },
  {
    "probability": 0.12221276015043259,
    "class": "n02124075 Egyptian cat"
  },
  {
    "probability": 0.028706775978207588,
    "class": "n02127052 lynx, catamount"
  },
  {
    "probability": 0.01915954425930977,
    "class": "n02129604 tiger, Panthera tigris"
  }
]
```

Per eseguire la pulizia dopo il test, eseguite un comando di eliminazione tramite RESTful API e arrestate il server del modello utilizzando i seguenti comandi:

```
curl -X DELETE http://127.0.0.1:8081/models/resnet-50_compiled

multi-model-server --stop
```

Verrà visualizzato l'output seguente:

```
{
  "status": "Model \"resnet-50_compiled\" unregistered"
}
Model server stopped.
Found 1 models and 1 NCGs.
Unloading 10001 (MODEL_STATUS_STARTED) :: success
Destroying NCG 1 :: success
```

Usando PyTorch -Neuron e AWS Compilatore Neuron

La compilazione PyTorch -Neuron API fornisce un metodo per compilare un grafico modello che è possibile eseguire su un AWS Dispositivo Inferentia.

Un modello addestrato deve essere compilato in un target Inferentia prima di poter essere distribuito nelle istanze di Inf1. Il seguente tutorial compila il modello torchvision ResNet 50 e lo esporta come modulo salvato. TorchScript Questo modello viene quindi utilizzato per eseguire l'inferenza.

Per comodità, questa esercitazione utilizza un'istanza di Inf1 sia per la compilazione sia per l'inferenza. In pratica, è possibile compilare il modello utilizzando un altro tipo di istanza, ad esempio la famiglia di istanze c5. È quindi necessario distribuire il modello compilato al server di inferenza Inf1. Per ulteriori informazioni, consultare la .AWS Documentazione Neuron. PyTorch SDK

Indice

Prerequisiti

Prima di utilizzare questo tutorial, è necessario aver completato la procedura di configurazione in Avvio di un'istanza con DLAMI AWS Neuron. Dovresti anche avere familiarità con il deep learning e l'uso di. DLAMI

Attivare l'ambiente Conda

Attiva l'ambiente PyTorch -Neuron conda usando il seguente comando:

```
source activate aws_neuron_pytorch_p36
```

Per uscire dall'ambiente conda corrente, eseguire:

```
source deactivate
```

Compilazione Resnet50

Creare uno script Python chiamato **`pytorch_trace_resnet50.py`** con il seguente contenuto. Questo script utilizza la compilazione PyTorch -Neuron API Python per compilare un modello -50. ResNet

> **Note**
>
> Esiste una dipendenza tra le versioni di torchvision e il pacchetto torch di cui dovresti essere a conoscenza durante la compilazione dei modelli torchvision. Queste regole di dipendenza possono essere gestite tramite pip. Torchvision==0.6.1 corrisponde alla versione torch==1.5.1, mentre torchvision==0.8.2 corrisponde alla versione torch==1.7.1.

```
import torch
import numpy as np
import os
import torch_neuron
from torchvision import models

image = torch.zeros([1, 3, 224, 224], dtype=torch.float32)

## Load a pretrained ResNet50 model
model = models.resnet50(pretrained=True)

## Tell the model we are using it for evaluation (not training)
model.eval()
model_neuron = torch.neuron.trace(model, example_inputs=[image])

## Export to saved model
model_neuron.save("resnet50_neuron.pt")
```

Eseguire lo script di compilazione.

```
python pytorch_trace_resnet50.py
```

La compilazione richiederà alcuni minuti. Al termine della compilazione, il modello compilato viene salvato come `resnet50_neuron.pt` nella directory locale.

ResNet50 Inferenza

Creare uno script Python chiamato **`pytorch_infer_resnet50.py`** con il seguente contenuto. Questo script scarica un'immagine di esempio e la usa per eseguire l'inferenza con il modello compilato.

```
import os
import time
import torch
import torch_neuron
import json
import numpy as np

from urllib import request

from torchvision import models, transforms, datasets

## Create an image directory containing a small kitten
os.makedirs("./torch_neuron_test/images", exist_ok=True)
request.urlretrieve("https://raw.githubusercontent.com/awslabs/mxnet-model-server/
master/docs/images/kitten_small.jpg",
                    "./torch_neuron_test/images/kitten_small.jpg")


## Fetch labels to output the top classifications
request.urlretrieve("https://s3.amazonaws.com/deep-learning-models/image-models/
imagenet_class_index.json","imagenet_class_index.json")
idx2label = []

with open("imagenet_class_index.json", "r") as read_file:
    class_idx = json.load(read_file)
    idx2label = [class_idx[str(k)][1] for k in range(len(class_idx))]

## Import a sample image and normalize it into a tensor
normalize = transforms.Normalize(
    mean=[0.485, 0.456, 0.406],
    std=[0.229, 0.224, 0.225])
```

```
eval_dataset = datasets.ImageFolder(
    os.path.dirname("./torch_neuron_test/"),
    transforms.Compose([
    transforms.Resize([224, 224]),
    transforms.ToTensor(),
    normalize,
    ])
)

image, _ = eval_dataset[0]
image = torch.tensor(image.numpy()[np.newaxis, ...])

## Load model
model_neuron = torch.jit.load( 'resnet50_neuron.pt' )

## Predict
results = model_neuron( image )

# Get the top 5 results
top5_idx = results[0].sort()[1][-5:]

# Lookup and print the top 5 labels
top5_labels = [idx2label[idx] for idx in top5_idx]

print("Top 5 labels:\n {}".format(top5_labels) )
```

Eseguire l'inferenza con il modello compilato utilizzando il seguente comando:

```
python pytorch_infer_resnet50.py
```

L'aspetto dell'output deve essere simile al seguente:

```
Top 5 labels:
 ['tiger', 'lynx', 'tiger_cat', 'Egyptian_cat', 'tabby']
```

La ARM64 DLAMI

AWS ARM64GPUDLAMIssono progettati per fornire prestazioni elevate ed efficienza in termini di costi per carichi di lavoro di deep learning. In particolare, il tipo di istanza g5G presenta quella basata su ARM64 AWS Processore Graviton2, costruito da zero da AWS e ottimizzato per il modo in cui i

clienti gestiscono i propri carichi di lavoro nel cloud. AWS ARM64GPUDLAMIssono preconfigurati con Docker, NVIDIA Docker, NVIDIA Driver, Cu CUDADNN,, oltre ai più diffusi framework di machine learning come e. NCCL TensorFlow PyTorch

Con il tipo di istanza g5G, puoi sfruttare i vantaggi in termini di prezzo e prestazioni di Graviton2 per implementare modelli di deep learning accelerati a un costo notevolmente inferiore rispetto alle istanze GPU basate su x86 con accelerazione. GPU

Seleziona un ARM64 DLAMI

Avvia un'istanza G5g con ARM64 DLAMI quella che preferisci.

Per step-by-step istruzioni sull'avvio di unDLAMI, consulta Avvio e configurazione di un. DLAMI

Per un elenco delle più recenti ARM64DLAMIs, consulta le note di rilascio per. DLAMI

Inizia

Negli argomenti seguenti viene illustrato come iniziare a utilizzare ARM64DLAMI.

Indice

Utilizzando il ARM64 GPU PyTorch DLAMI

Il AWS Deep Learning AMIs è pronto per l'uso con processori Arm64 ed è GPUs ottimizzato per. PyTorch ARM64GPU PyTorch DLAMIInclude un ambiente Python preconfigurato con e TorchServeper casi PyTorchd'TorchVisionuso di inferenza e formazione di deep learning.

Indice

Verifica dell' PyTorch ambiente Python

Connettiti alla tua istanza G5g e attiva l'ambiente Conda di base con il seguente comando:

```
source activate base
```

Il prompt dei comandi dovrebbe indicare che stai lavorando nell'ambiente Conda di base, che contiene e altre PyTorch librerie TorchVision.

```
(base) $
```

Verificate i percorsi utensile predefiniti dell' PyTorch ambiente:

```
(base) $ which python
(base) $ which pip
(base) $ which conda
(base) $ which mamba
>>> import torch, torchvision
>>> torch.__version__
>>> torchvision.__version__
>>> v = torch.autograd.Variable(torch.randn(10, 3, 224, 224))
>>> v = torch.autograd.Variable(torch.randn(10, 3, 224, 224)).cuda()
>>> assert isinstance(v, torch.Tensor)
```

Esegui Training Sample con PyTorch

Esegui un esempio di lavoro di MNIST formazione:

```
git clone https://github.com/pytorch/examples.git
cd examples/mnist
python main.py
```

L'aspetto dell'output sarà simile al seguente:

```
...
Train Epoch: 14 [56320/60000 (94%)]    Loss: 0.021424
Train Epoch: 14 [56960/60000 (95%)]    Loss: 0.023695
Train Epoch: 14 [57600/60000 (96%)]    Loss: 0.001973
Train Epoch: 14 [58240/60000 (97%)]    Loss: 0.007121
Train Epoch: 14 [58880/60000 (98%)]    Loss: 0.003717
Train Epoch: 14 [59520/60000 (99%)]    Loss: 0.001729
Test set: Average loss: 0.0275, Accuracy: 9916/10000 (99%)
```

Esegui Inference Sample con PyTorch

Usa i seguenti comandi per scaricare un modello densenet161 pre-addestrato ed eseguire l'inferenza utilizzando: TorchServe

```
# Set up TorchServe
cd $HOME
git clone https://github.com/pytorch/serve.git
mkdir -p serve/model_store
cd serve

# Download a pre-trained densenet161 model
wget https://download.pytorch.org/models/densenet161-8d451a50.pth >/dev/null

# Save the model using torch-model-archiver
torch-model-archiver --model-name densenet161 \
    --version 1.0 \
    --model-file examples/image_classifier/densenet_161/model.py \
    --serialized-file densenet161-8d451a50.pth \
    --handler image_classifier \
    --extra-files examples/image_classifier/index_to_name.json  \
    --export-path model_store

# Start the model server
torchserve --start --no-config-snapshots \
    --model-store model_store \
    --models densenet161=densenet161.mar &> torchserve.log

# Wait for the model server to start
sleep 30

# Run a prediction request
curl http://127.0.0.1:8080/predictions/densenet161 -T examples/image_classifier/
kitten.jpg
```

L'aspetto dell'output sarà simile al seguente:

```
{
  "tiger_cat": 0.4693363308906555,
  "tabby": 0.4633873701095581,
  "Egyptian_cat": 0.06456123292446136,
  "lynx": 0.0012828150065615773,
  "plastic_bag": 0.00023322898778133094
}
```

Utilizzate i seguenti comandi per annullare la registrazione del modello densenet161 e arrestare il server:

```
curl -X DELETE http://localhost:8081/models/densenet161/1.0
torchserve --stop
```

L'aspetto dell'output sarà simile al seguente:

```
{
  "status": "Model \"densenet161\" unregistered"
}
TorchServe has stopped.
```

Inferenza

Questa sezione fornisce tutorial su come eseguire l'inferenza utilizzando i framework e gli strumentiDLAMI.

Strumenti di inferenza

- TensorFlow Servire

Model serving

Di seguito sono riportate le opzioni di servizio dei modelli installate su Deep Learning AMI con Conda. Fai clic su quella desiderata per informazioni su come utilizzarla.

Argomenti

- TensorFlow Servire
- TorchServe

TensorFlow Servire

TensorFlow Serving è un sistema di servizio flessibile e ad alte prestazioni per modelli di apprendimento automatico.

`tensorflow-serving-api`È preinstallato con Deep Learning AMI with Conda! Troverai uno script di esempio in cui addestrare, esportare e servire un MNIST modello. `~/examples/tensorflow-serving/`

Per eseguire uno di questi esempi, connettiti prima al tuo Deep Learning AMI con Conda e attiva l' TensorFlow ambiente.

```
$ source activate tensorflow2_p310
```

Ora accedi alla cartella con gli esempi di script.

```
$ cd ~/examples/tensorflow-serving/
```

Distribuzione di un modello Inception preformato

Di seguito è riportato un esempio che può essere provato per distribuire diversi modelli come Inception. Come regola generale, è necessario un modello utilizzabile e gli script client siano già scaricati sul proprio. DLAMI

Distribuzione e test dell'inferenza con un modello Inception

1. Scarica il modello.

   ```
   $ curl -O https://s3-us-west-2.amazonaws.com/tf-test-models/INCEPTION.zip
   ```

2. Estrai il modello.

   ```
   $ unzip INCEPTION.zip
   ```

3. Scaricare un'immagine di un husky.

   ```
   $ curl -O https://upload.wikimedia.org/wikipedia/commons/b/b5/Siberian_Husky_bi-
   eyed_Flickr.jpg
   ```

4. Avvia il server. Per Amazon Linux è necessario modificare la directory utilizzata per `model_base_path` da `/home/ubuntu` a `/home/ec2-user`.

   ```
   $ tensorflow_model_server --model_name=INCEPTION --model_base_path=/home/ubuntu/
   examples/tensorflow-serving/INCEPTION/INCEPTION --port=9000
   ```

5. Con il server in esecuzione in primo piano, è necessario avviare un'altra sessione di terminale per continuare. Apri un nuovo terminale e attiva TensorFlow con. `source activate tensorflow2_p310` Quindi utilizza l'editor di testo preferito per creare uno script che ha i

seguenti contenuti. Denominalo `inception_client.py`. Lo script prenderà un nome file immagine come parametro e otterrà un risultato di previsione dal modello preformato.

```
from __future__ import print_function

import grpc
import tensorflow as tf
import argparse

from tensorflow_serving.apis import predict_pb2
from tensorflow_serving.apis import prediction_service_pb2_grpc

parser = argparse.ArgumentParser(
    description='TF Serving Test',
    formatter_class=argparse.ArgumentDefaultsHelpFormatter
)
parser.add_argument('--server_address', default='localhost:9000',
                    help='Tenforflow Model Server Address')
parser.add_argument('--image', default='Siberian_Husky_bi-eyed_Flickr.jpg',
                    help='Path to the image')
args = parser.parse_args()


def main():
  channel = grpc.insecure_channel(args.server_address)
  stub = prediction_service_pb2_grpc.PredictionServiceStub(channel)
  # Send request
  with open(args.image, 'rb') as f:
    # See prediction_service.proto for gRPC request/response details.
    request = predict_pb2.PredictRequest()
    request.model_spec.name = 'INCEPTION'
    request.model_spec.signature_name = 'predict_images'

    input_name = 'images'
    input_shape = [1]
    input_data = f.read()
    request.inputs[input_name].CopyFrom(
      tf.make_tensor_proto(input_data, shape=input_shape))

    result = stub.Predict(request, 10.0)  # 10 secs timeout
    print(result)

  print("Inception Client Passed")
```

```
if __name__ == '__main__':
  main()
```

6. Ora esegui lo script fornendo la posizione e la porta del server e il nome della foto dell'husky come parametri.

```
$ python3 inception_client.py --server=localhost:9000 --image Siberian_Husky_bi-
eyed_Flickr.jpg
```

Addestra e servi un MNIST modello

Per questo tutorial, esporteremo un modello, quindi lo serviremo con l'applicazione `tensorflow_model_server`. Infine, testeremo il server di modelli con un esempio di script client.

Esegui lo script che addestrerà ed esporterà un MNIST modello. Come unico argomento dello script, è necessario fornire il percorso di una cartella in cui salvare il modello. Per il momento, possiamo metterla in `mnist_model`. Lo script creerà la cartella.

```
$ python mnist_saved_model.py /tmp/mnist_model
```

L'esecuzione dello script può durare alcuni minuti. Al termine del training e dell'esportazione del modello, viene visualizzato quanto segue:

```
Done training!
Exporting trained model to mnist_model/1
Done exporting!
```

L'operazione successiva consiste nell'eseguire `tensorflow_model_server` per servire il modello esportato.

```
$ tensorflow_model_server --port=9000 --model_name=mnist --model_base_path=/tmp/
mnist_model
```

Uno script client viene fornito per testare il server.

Per eseguire il test, devi aprire una nuova finestra del terminale.

```
$ python mnist_client.py --num_tests=1000 --server=localhost:9000
```

Ulteriori funzionalità ed esempi

Se sei interessato a saperne di più su TensorFlow Serving, consulta il TensorFlow sito web.

Puoi anche usare TensorFlow Serving with Amazon Elastic Inference. Consulta la guida su come utilizzare Elastic Inference with TensorFlow Serving per maggiori informazioni.

TorchServe

TorchServe è uno strumento flessibile per servire modelli di deep learning che sono stati esportati da. PyTorch TorchServe viene preinstallato con Deep Learning AMI with Conda.

Per ulteriori informazioni sull'utilizzo TorchServe, vedere Model Server for PyTorch Documentation.

Argomenti

Offri un modello di classificazione delle immagini su TorchServe

Questo tutorial mostra come utilizzare un modello di classificazione delle immagini con TorchServe. Utilizza un modello DenseNet -161 fornito da PyTorch. Una volta che il server è in esecuzione, ascolta le richieste di previsione. Quando carichi un'immagine, in questo caso l'immagine di un gattino, il server restituisce una previsione delle 5 migliori classi corrispondenti tra le classi su cui è stato addestrato il modello.

Per fornire un esempio di modello di classificazione delle immagini su TorchServe

1. Connettiti a un'istanza Amazon Elastic Compute Cloud (AmazonEC2) con Deep Learning AMI with Conda v34 o versione successiva.
2. Attiva l'ambiente. `pytorch_p310`

   ```
   source activate pytorch_p310
   ```

3. Clona il TorchServe repository, quindi crea una directory per archiviare i tuoi modelli.

   ```
   git clone https://github.com/pytorch/serve.git
   mkdir model_store
   ```

4. Archivia il modello utilizzando il model archiver. Il `extra-files` parametro utilizza un file del `TorchServe` repository, quindi aggiorna il percorso se necessario. Per ulteriori informazioni sul model archiver, vedere Torch Model archiver for. TorchServe

```
wget https://download.pytorch.org/models/densenet161-8d451a50.pth
torch-model-archiver --model-name densenet161 --version 1.0 --model-file ./
serve/examples/image_classifier/densenet_161/model.py --serialized-file
 densenet161-8d451a50.pth --export-path model_store --extra-files ./serve/examples/
image_classifier/index_to_name.json --handler image_classifier
```

5. Esegui TorchServe per avviare un endpoint. L'aggiunta > `/dev/null` disattiva l'output del registro.

```
torchserve --start --ncs --model-store model_store --models densenet161.mar > /dev/
null
```

6. Scaricate l'immagine di un gattino e inviatela all'endpoint TorchServe previsto:

```
curl -O https://s3.amazonaws.com/model-server/inputs/kitten.jpg
curl http://127.0.0.1:8080/predictions/densenet161 -T kitten.jpg
```

   L'endpoint di previsione restituisce una previsione JSON simile alle prime cinque previsioni seguenti, in cui l'immagine ha una probabilità del 47% di contenere un gatto egiziano, seguita da una probabilità del 46% che abbia un gatto soriano.

```
{
 "tiger_cat": 0.46933576464653015,
 "tabby": 0.463387668132782,
 "Egyptian_cat": 0.0645613968372345,
 "lynx": 0.0012828196631744504,
 "plastic_bag": 0.00023323058849200606
}
```

7. Al termine del test, arresta il server:

```
torchserve --stop
```

Altri esempi

TorchServe ha una serie di esempi che puoi eseguire sulla tua DLAMI istanza. Puoi visualizzarli nella pagina degli esempi del repository TorchServe del progetto.

Maggiori informazioni

Per ulteriore TorchServe documentazione, incluso come configurare Docker e TorchServe le TorchServe funzionalità più recenti, consulta la pagina del TorchServe progetto su GitHub.

Aggiornamento del tuo DLAMI

Qui troverai informazioni sull'aggiornamento del tuo DLAMI e suggerimenti sull'aggiornamento del software del tuo DLAMI.

Argomenti

Aggiornamento a una nuova versione DLAMI

Le immagini di sistema di DLAMI vengono aggiornate regolarmente per sfruttare le nuove versioni del framework di deep learning, CUDA e altri aggiornamenti software e l'ottimizzazione delle prestazioni. Se utilizzi un DLAMI da qualche tempo e desideri sfruttare un aggiornamento, dovrai avviare una nuova istanza. Devi inoltre trasferire manualmente set di dati, checkpoint o altri dati importanti. Puoi invece utilizzare Amazon EBS per conservare i tuoi dati e collegarli a un nuovo DLAMI. In questo modo, puoi eseguire regolarmente l'upgrade riducendo al minimo il tempo necessario per la transizione dei dati.

> **Note**
>
> Quando colleghi e sposti volumi Amazon EBS tra DLAMI, devi avere sia i DLAMI che il nuovo volume nella stessa zona di disponibilità.

1. Usa la console Amazon EC2 per creare un nuovo volume Amazon EBS. Per istruzioni dettagliate, consulta Creazione di un volume Amazon EBS.
2. Collega il volume Amazon EBS appena creato al tuo DLAMI esistente. Per istruzioni dettagliate, consulta Allegare un volume Amazon EBS.
3. Trasferire i dati, come set di dati, checkpoint e file di configurazione.
4. Avvia un DLAMI. Per istruzioni dettagliate, consulta Avvio e configurazione di una DLAMI.
5. Scollega il volume Amazon EBS dal tuo vecchio DLAMI. Per istruzioni dettagliate, consulta Scollegare un volume Amazon EBS.

6. Collega il volume Amazon EBS al tuo nuovo DLAMI. Seguire le istruzioni dal punto 2 per collegare il volume.
7. Dopo aver verificato che i dati siano disponibili sul nuovo DLAMI, interrompi e chiudi il vecchio DLAMI. Per istruzioni di pulizia dettagliate, consulta Pulizia.

Suggerimenti per gli aggiornamenti software

Di tanto in tanto, potresti voler aggiornare manualmente il software sul tuo DLAMI. In generale, è consigliabile utilizzare `pip` per aggiornare i pacchetti Python. È inoltre necessario utilizzare `pip` per aggiornare i pacchetti all'interno di un ambiente Conda sull'AMI Deep Learning con Conda. Per istruzioni relative all'aggiornamento e all'installazione, visita il sito Web del framework o del software in questione.

> **Note**
>
> Non possiamo garantire che l'aggiornamento di un pacchetto abbia successo. Il tentativo di aggiornare un pacchetto in un ambiente con dipendenze incompatibili può causare un errore. In tal caso, è necessario contattare il responsabile della libreria per vedere se è possibile aggiornare le dipendenze del pacchetto. In alternativa, puoi provare a modificare l'ambiente in modo tale da consentire l'aggiornamento. Tuttavia, questa modifica comporterà probabilmente la rimozione o l'aggiornamento dei pacchetti esistenti, il che significa che non possiamo più garantire la stabilità di questo ambiente.

AWS Deep Learning AMIs Viene fornito con molti ambienti Conda e molti pacchetti preinstallati. A causa del numero di pacchetti preinstallati, è difficile trovare un set di pacchetti la cui compatibilità sia garantita. Potresti visualizzare un avviso «L'ambiente non è coerente, controlla attentamente il piano dei pacchetti». DLAMI assicura che tutti gli ambienti forniti da DLAMI siano corretti, ma non può garantire che i pacchetti installati dall'utente funzionino correttamente.

Ricevi notifiche sui nuovi aggiornamenti

> **Note**
>
> AWS Le AMI di Deep Learning hanno una cadenza di rilascio settimanale per le patch di sicurezza. Le notifiche di rilascio verranno inviate per queste patch di sicurezza incrementali, anche se potrebbero non essere incluse nelle note di rilascio ufficiali.

È possibile ricevere notifiche ogni volta che viene rilasciato un nuovo DLAMI. Le notifiche vengono pubblicate con Amazon SNS utilizzando il seguente argomento.

```
arn:aws:sns:us-west-2:767397762724:dlami-updates
```

I messaggi vengono pubblicati qui quando viene pubblicato un nuovo DLAMI. La versione, i metadati e gli ID AMI regionali dell'AMI verranno inclusi nel messaggio.

Questi messaggi possono essere ricevuti utilizzando diversi metodi. Si consiglia di utilizzare il seguente metodo.

1. Apri la console Amazon SNS.
2. Nella barra di navigazione, cambia la AWS regione in Stati Uniti occidentali (Oregon), se necessario. Devi selezionare la regione in cui è stata creata la notifica SNS a cui ti stai abbonando.
3. Nel pannello di navigazione, scegli Abbonamenti, Crea abbonamento.
4. Nella finestra di dialogo Create subscription (Crea sottoscrizione) eseguire le seguenti operazioni:
   a. Per l'argomento ARN, copia e incolla il seguente Amazon Resource Name (ARN): **`arn:aws:sns:us-west-2:767397762724:dlami-updates`**
   b. Per Protocol, scegline uno tra [Amazon SQS, AWS Lamda, Email, Email-JSON]
   c. Per Endpoint, inserisci l'indirizzo e-mail o Amazon Resource Name (ARN) della risorsa che utilizzerai per ricevere le notifiche.
   d. Scegli Crea sottoscrizione.
5. Riceverai un'e-mail di conferma con oggetto AWS Notifica - Conferma dell'abbonamento. Apri l'e-mail e seleziona Conferma sottoscrizione per completare la sottoscrizione.

Sicurezza in AWS Deep Learning AMIs

Sicurezza nel cloud presso AWS è la massima priorità. Come AWS cliente, trarrai vantaggio da un'architettura di data center e rete progettata per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra AWS e tu. Il modello di responsabilità condivisa descrive questo aspetto come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud — AWS è responsabile della protezione dell'infrastruttura in esecuzione AWS servizi in AWS Cloud. AWS ti offre anche servizi che puoi utilizzare in modo sicuro. I revisori di terze parti testano e verificano regolarmente l'efficacia della nostra sicurezza nell'ambito del AWS Programmi di conformità . Per ulteriori informazioni sui programmi di conformità applicabiliDLAMI, vedere AWS Servizi rientranti nell'ambito del programma di conformità .
- Sicurezza nel cloud: la tua responsabilità è determinata dal AWS servizio che utilizzi. Sei anche responsabile di altri fattori, tra cui la riservatezza dei dati, i requisiti della tua azienda e le leggi e normative vigenti.

Questa documentazione aiuta a capire come applicare il modello di responsabilità condivisa durante l'utilizzoDLAMI. I seguenti argomenti mostrano come eseguire la configurazione DLAMI per soddisfare gli obiettivi di sicurezza e conformità. Imparerai anche a usarne altri AWS servizi che ti aiutano a monitorare e proteggere DLAMI le tue risorse.

Per ulteriori informazioni, consulta la sezione Sicurezza in Amazon EC2.

Argomenti

Protezione dei dati in AWS Deep Learning AMIs

Il AWS modello di responsabilità condivisa modello di di si applica alla protezione dei dati in AWS Apprendimento profondoAMI. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutte le Cloud AWS. L'utente è responsabile del mantenimento del controllo sui contenuti ospitati su questa infrastruttura. L'utente è inoltre responsabile delle attività di configurazione e gestione della sicurezza per Servizi AWS che usi. Per ulteriori informazioni sulla privacy dei dati, consulta la sezione Privacy dei dati FAQ. Per informazioni sulla protezione dei dati in Europa, consulta AWS Modello di responsabilità condivisa e post sul GDPR blog sul AWS Blog sulla sicurezza.

Ai fini della protezione dei dati, ti consigliamo di proteggere Account AWS credenziali e configura singoli utenti con AWS IAM Identity Center oppure AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Usa l'autenticazione a più fattori (MFA) con ogni account.
- UsaSSL/TLSper comunicare con AWS risorse. Richiediamo TLS 1.2 e consigliamo TLS 1.3.
- Configurazione API e registrazione delle attività degli utenti con AWS CloudTrail. Per informazioni sull'utilizzo dei CloudTrail percorsi per l'acquisizione AWS attività, vedi Lavorare con i CloudTrail sentieri in AWS CloudTrail Guida per l'utente.
- Utilizzo AWS soluzioni di crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.
- Utilizza i servizi di sicurezza gestiti avanzati, come Amazon Macie, che aiutano a individuare e proteggere i dati sensibili archiviati in Amazon S3.
- Se sono necessari FIPS 140-3 moduli crittografici convalidati per l'accesso AWS tramite un'interfaccia a riga di comando o unAPI, utilizza un endpoint. FIPS Per ulteriori informazioni sugli FIPS endpoint disponibili, vedere Federal Information Processing Standard (FIPS) 140-3.

Ti consigliamo vivamente di non inserire mai informazioni riservate o sensibili, ad esempio gli indirizzi e-mail dei clienti, nei tag o nei campi di testo in formato libero, ad esempio nel campo Nome. Ciò include quando lavori con DLAMI o altro Servizi AWS utilizzando la consoleAPI, AWS CLI, oppure AWS SDKs. I dati inseriti nei tag o nei campi di testo in formato libero utilizzati per i nomi possono essere utilizzati per i la fatturazione o i log di diagnostica. Se fornisci un URL a un server esterno, ti consigliamo vivamente di non includere le informazioni sulle credenziali URL per convalidare la tua richiesta a quel server.

Identity and Access Management in AWS Deep Learning AMIs

AWS Identity and Access Management (IAM) è un Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso a AWS risorse. IAMgli amministratori controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (dispone delle autorizzazioni) a utilizzare le risorse. DLAMI IAMè un Servizio AWS che puoi utilizzare senza costi aggiuntivi.

Per ulteriori informazioni su Identity and Access Management, consulta Identity and Access Management for Amazon EC2.

Argomenti

- Autenticazione con identità
- Gestione dell'accesso tramite policy
- IAMcon Amazon EMR

Autenticazione con identità

L'autenticazione è il modo in cui si accede a AWS utilizzando le tue credenziali di identità. Devi essere autenticato (aver effettuato l'accesso a AWS) come Utente root dell'account AWS, come IAM utente o assumendo un IAM ruolo.

Puoi accedere a AWS come identità federata utilizzando le credenziali fornite tramite una fonte di identità. AWS IAM Identity Center Gli utenti (IAMIdentity Center), l'autenticazione Single Sign-On della tua azienda e le tue credenziali Google o Facebook sono esempi di identità federate. Quando accedi come identità federata, l'amministratore aveva precedentemente configurato la federazione delle identità utilizzando i ruoli. IAM Quando accedi AWS utilizzando la federazione, assumi indirettamente un ruolo.

A seconda del tipo di utente che sei, puoi accedere a AWS Management Console o il AWS portale di accesso. Per ulteriori informazioni sull'accesso a AWS, vedi Come accedere al tuo Account AWS nella Accedi ad AWS Guida per l'utente.

Se accedi AWS programmaticamente, AWS fornisce un kit di sviluppo software (SDK) e un'interfaccia a riga di comando (CLI) per firmare crittograficamente le richieste utilizzando le credenziali dell'utente. Se non usi AWS strumenti, devi firmare tu stesso le richieste. Per ulteriori informazioni sull'utilizzo del metodo consigliato per firmare autonomamente le richieste, vedi Firma AWS APIrichieste nella Guida IAM per l'utente.

A prescindere dal metodo di autenticazione utilizzato, potrebbe essere necessario specificare ulteriori informazioni sulla sicurezza. Ad esempio, AWS consiglia di utilizzare l'autenticazione a più fattori (MFA) per aumentare la sicurezza del proprio account. Per ulteriori informazioni, consulta Autenticazione a più fattori nel AWS IAM Identity Center Guida per l'utente e utilizzo dell'autenticazione a più fattori () MFA in AWS nella Guida per l'utente di IAM.

Account AWS utente root

Quando crei un Account AWS, inizi con un'unica identità di accesso con accesso completo a tutti Servizi AWS e le risorse presenti nell'account. Questa identità è denominata Account AWS utente root ed è accessibile effettuando l'accesso con l'indirizzo e-mail e la password utilizzati per creare l'account. Si consiglia vivamente di non utilizzare l'utente root per le attività quotidiane. Conserva le credenziali dell'utente root e utilizzale per eseguire le operazioni che solo l'utente root può eseguire. Per l'elenco completo delle attività che richiedono l'accesso come utente root, consulta Attività che richiedono le credenziali dell'utente root nella Guida per l'IAMutente.

Utenti e gruppi IAM

Un IAMutente è un'identità all'interno del tuo Account AWS che dispone di autorizzazioni specifiche per una singola persona o applicazione. Laddove possibile, consigliamo di fare affidamento su credenziali temporanee anziché creare IAM utenti con credenziali a lungo termine come password e chiavi di accesso. Tuttavia, se hai casi d'uso specifici che richiedono credenziali a lungo termine con IAM gli utenti, ti consigliamo di ruotare le chiavi di accesso. Per ulteriori informazioni, consulta Ruotare regolarmente le chiavi di accesso per i casi d'uso che richiedono credenziali a lungo termine nella Guida per l'utente. IAM

Un IAMgruppo è un'identità che specifica un insieme di utenti. IAM Non è possibile eseguire l'accesso come gruppo. È possibile utilizzare gruppi per specificare le autorizzazioni per più utenti alla volta. I gruppi semplificano la gestione delle autorizzazioni per set di utenti di grandi dimensioni. Ad esempio, potresti avere un gruppo denominato IAMAdminse concedere a quel gruppo le autorizzazioni per IAM amministrare le risorse.

Gli utenti sono diversi dai ruoli. Un utente è associato in modo univoco a una persona o un'applicazione, mentre un ruolo è destinato a essere assunto da chiunque ne abbia bisogno. Gli utenti dispongono di credenziali a lungo termine permanenti, mentre i ruoli forniscono credenziali temporanee. Per ulteriori informazioni, consulta Quando creare un IAM utente (anziché un ruolo) nella Guida per l'IAMutente.

IAMRuoli

Un IAMruolo è un'identità all'interno di te Account AWS che dispone di autorizzazioni specifiche. È simile a un IAM utente, ma non è associato a una persona specifica. È possibile assumere temporaneamente un IAM ruolo nel AWS Management Console cambiando ruolo. Puoi assumere un ruolo chiamando un AWS CLI oppure AWS APIoperazione o utilizzando un comando personalizzatoURL. Per ulteriori informazioni sui metodi di utilizzo dei ruoli, vedere Utilizzo IAM dei ruoli nella Guida per l'IAMutente.

IAMi ruoli con credenziali temporanee sono utili nelle seguenti situazioni:

- Accesso utente federato: per assegnare le autorizzazioni a una identità federata, è possibile creare un ruolo e definire le autorizzazioni per il ruolo. Quando un'identità federata viene autenticata, l'identità viene associata al ruolo e ottiene le autorizzazioni da esso definite. Per informazioni sui ruoli per la federazione, vedere Creazione di un ruolo per un provider di identità di terze parti nella Guida per l'IAMutente. Se utilizzi IAM Identity Center, configuri un set di autorizzazioni. Per controllare a cosa possono accedere le identità dopo l'autenticazione, IAM Identity Center correla il set di autorizzazioni a un ruolo in. IAM Per informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nel AWS IAM Identity Center Guida per l'utente.
- Autorizzazioni IAM utente temporanee: un IAM utente o un ruolo può assumere un IAM ruolo per assumere temporaneamente autorizzazioni diverse per un'attività specifica.
- Accesso su più account: puoi utilizzare un IAM ruolo per consentire a qualcuno (un responsabile fidato) di un altro account di accedere alle risorse del tuo account. I ruoli sono lo strumento principale per concedere l'accesso multi-account. Tuttavia, con alcuni Servizi AWS, è possibile allegare una policy direttamente a una risorsa (anziché utilizzare un ruolo come proxy). Per conoscere la differenza tra i ruoli e le politiche basate sulle risorse per l'accesso tra più account, consulta l'accesso alle risorse tra account IAM nella Guida per l'utente. IAM
- Accesso a più servizi: alcuni Servizi AWS usa le funzionalità in altri Servizi AWS. Ad esempio, quando effettui una chiamata in un servizio, è normale che quel servizio esegua applicazioni in Amazon EC2 o archivi oggetti in Amazon S3. Un servizio può eseguire questa operazione utilizzando le autorizzazioni dell'entità chiamante, utilizzando un ruolo di servizio o utilizzando un ruolo collegato al servizio.
  - Sessioni di accesso diretto (FAS): quando utilizzi un IAM utente o un ruolo per eseguire azioni in AWS, sei considerato un preside. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra operazione in un servizio diverso. FASutilizza le autorizzazioni del principale che chiama un Servizio AWS, in combinazione con la richiesta Servizio AWS per effettuare richieste ai servizi a valle. FASle richieste vengono effettuate solo quando un servizio

riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse da completare. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le azioni. Per i dettagli FAS delle politiche relative alle richieste, consulta Forward access sessions.

- Ruolo di servizio: un ruolo di servizio è un IAMruolo che un servizio assume per eseguire azioni per conto dell'utente. Un IAM amministratore può creare, modificare ed eliminare un ruolo di servizio dall'internoIAM. Per ulteriori informazioni, vedere Creazione di un ruolo per delegare le autorizzazioni a un Servizio AWS nella Guida per l'utente di IAM.
- Ruolo collegato al servizio: un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un Servizio AWS. Il servizio può assumere il ruolo di eseguire un'azione per conto dell'utente. I ruoli collegati ai servizi vengono visualizzati nel tuo Account AWS e sono di proprietà del servizio. Un IAM amministratore può visualizzare, ma non modificare le autorizzazioni per i ruoli collegati al servizio.

- Applicazioni in esecuzione su Amazon EC2: puoi utilizzare un IAM ruolo per gestire le credenziali temporanee per le applicazioni in esecuzione su un'EC2istanza e in fase di creazione AWS CLI oppure AWS APIrichieste. Ciò è preferibile alla memorizzazione delle chiavi di accesso all'interno dell'EC2istanza. Per assegnare un AWS assegnare un ruolo a un'EC2istanza e renderlo disponibile a tutte le relative applicazioni, è necessario creare un profilo di istanza collegato all'istanza. Un profilo di istanza contiene il ruolo e consente ai programmi in esecuzione sull'EC2istanza di ottenere credenziali temporanee. Per ulteriori informazioni, consulta Usare un IAM ruolo per concedere le autorizzazioni alle applicazioni in esecuzione su EC2 istanze Amazon nella Guida per l'IAMutente.

Per sapere se utilizzare IAM ruoli o IAM utenti, consulta Quando creare un IAM ruolo (anziché un utente) nella Guida per l'IAMutente.

Gestione dell'accesso tramite policy

Puoi controllare l'accesso in AWS creando politiche e allegandole a AWS identità o risorse. Una politica è un oggetto in AWS che, se associato a un'identità o a una risorsa, ne definisce le autorizzazioni. AWS valuta queste politiche quando un principale (utente, utente root o sessione di ruolo) effettua una richiesta. Le autorizzazioni nelle policy determinano l'approvazione o il rifiuto della richiesta. La maggior parte delle politiche viene archiviata in AWS come JSON documenti. Per ulteriori informazioni sulla struttura e il contenuto dei documenti relativi alle JSON politiche, vedere Panoramica delle JSON politiche nella Guida per l'IAMutente.

Gli amministratori possono utilizzare AWS JSONpolitiche per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Per concedere agli utenti il permesso di eseguire azioni sulle risorse di cui hanno bisogno, un IAM amministratore può creare IAM politiche. L'amministratore può quindi aggiungere le IAM politiche ai ruoli e gli utenti possono assumerli.

IAMle politiche definiscono le autorizzazioni per un'azione indipendentemente dal metodo utilizzato per eseguire l'operazione. Ad esempio, supponiamo di disporre di una policy che consente l'operazione `iam:GetRole`. Un utente con tale criterio può ottenere informazioni sul ruolo da AWS Management Console, il AWS CLI, o AWS API.

Policy basate sulle identità

I criteri basati sull'identità sono documenti relativi ai criteri di JSON autorizzazione che è possibile allegare a un'identità, ad esempio un IAM utente, un gruppo di utenti o un ruolo. Tali policy definiscono le azioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una politica basata sull'identità, consulta Creazione di politiche nella Guida per l'utente. IAM IAM

Le policy basate su identità possono essere ulteriormente classificate come policy inline o policy gestite. Le policy inline sono integrate direttamente in un singolo utente, gruppo o ruolo. Le politiche gestite sono politiche autonome che puoi allegare a più utenti, gruppi e ruoli nel tuo Account AWS. Le politiche gestite includono AWS politiche gestite e politiche gestite dai clienti. Per informazioni su come scegliere tra una politica gestita o una politica in linea, consulta Scelta tra politiche gestite e politiche in linea nella Guida per l'IAMutente.

Policy basate su risorse

Le politiche basate sulle risorse sono documenti di JSON policy allegati a una risorsa. Esempi di policy basate sulle risorse sono le policy di IAM role trust e le policy di Amazon S3 bucket. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le azioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o Servizi AWS.

Le policy basate sulle risorse sono policy inline che si trovano in tale servizio. Non puoi usare AWS politiche gestite da IAM una politica basata sulle risorse.

Liste di controllo degli accessi () ACLs

Le liste di controllo degli accessi (ACLs) controllano quali principali (membri dell'account, utenti o ruoli) dispongono delle autorizzazioni per accedere a una risorsa. ACLssono simili alle politiche basate sulle risorse, sebbene non utilizzino il formato del documento di policy. JSON

Amazon S3, AWS WAF e Amazon VPC sono esempi di servizi che supportanoACLs. Per ulteriori informazioniACLs, consulta la panoramica di Access control list (ACL) nella Amazon Simple Storage Service Developer Guide.

Altri tipi di policy

AWS supporta tipi di policy aggiuntivi e meno comuni. Questi tipi di policy possono impostare il numero massimo di autorizzazioni concesse dai tipi di policy più comuni.

- Limiti delle autorizzazioni: un limite di autorizzazioni è una funzionalità avanzata in cui si impostano le autorizzazioni massime che una politica basata sull'identità può concedere a un'entità (utente o ruolo). IAM IAM È possibile impostare un limite delle autorizzazioni per un'entità. Le autorizzazioni risultanti sono l'intersezione delle policy basate su identità dell'entità e i relativi limiti delle autorizzazioni. Le policy basate su risorse che specificano l'utente o il ruolo nel campo `Principal`sono condizionate dal limite delle autorizzazioni. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni sui limiti delle autorizzazioni, consulta Limiti delle autorizzazioni per le entità nella Guida per l'utente. IAM IAM
- Politiche di controllo del servizio (SCPs): SCPs sono JSON politiche che specificano le autorizzazioni massime per un'organizzazione o un'unità organizzativa (OU) in AWS Organizations. AWS Organizations è un servizio per il raggruppamento e la gestione centralizzata di più Account AWS di cui è proprietaria la tua azienda. Se abiliti tutte le funzionalità di un'organizzazione, puoi applicare le politiche di controllo del servizio (SCPs) a uno o tutti i tuoi account. I SCP limiti e le autorizzazioni per le entità presenti negli account dei membri, inclusi tutti Utente root dell'account AWS. Per ulteriori informazioni su Organizations andSCPs, vedere Service control policies nel AWS Organizations Guida per l'utente.
- Policy di sessione: le policy di sessione sono policy avanzate che vengono trasmesse come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un utente federato. Le autorizzazioni della sessione risultante sono l'intersezione delle policy basate su identità del ruolo o dell'utente e le policy di sessione. Le autorizzazioni possono anche provenire da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni, consulta le politiche di sessione nella Guida IAM per l'utente.

Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per scoprire come AWS determina se consentire una richiesta quando sono coinvolti più tipi di policy, consulta Logica di valutazione delle policy nella Guida per l'IAMutente.

IAMcon Amazon EMR

È possibile utilizzare… AWS Identity and Access Management con Amazon EMR per definire gli utenti, AWS risorse, gruppi, ruoli e politiche. Puoi anche controllare quali AWS servizi a cui possono accedere questi utenti e ruoli.

Per ulteriori informazioni sull'utilizzo IAM con AmazonEMR, consulta AWS Identity and Access Management per Amazon EMR.

Registrazione e monitoraggio AWS Deep Learning AMIs

Il tuo AWS Deep Learning AMIs l'istanza è dotata di diversi strumenti di GPU monitoraggio, tra cui un'utilità che riporta le statistiche di GPU utilizzo ad Amazon CloudWatch. Per ulteriori informazioni, consulta GPUMonitoring and Optimization and Monitoring Amazon EC2.

Monitoraggio dell'utilizzo

I seguenti AWS Deep Learning AMIs le distribuzioni del sistema operativo includono codice che consente AWS per raccogliere informazioni sul tipo di istanza, sull'ID dell'istanza, sul DLAMI tipo e sul sistema operativo. Nessuna informazione sui comandi utilizzati all'interno di DLAMI viene raccolta o conservata. Non DLAMI viene raccolta o conservata nessun'altra informazione su.

- Ubuntu 16.04
- Ubuntu 18.04
- Ubuntu 20.04
- Amazon Linux 2

Per disattivare il monitoraggio dell'utilizzo per la tuaDLAMI, aggiungi un tag alla tua EC2 istanza Amazon durante il lancio. Il tag deve utilizzare la chiave `OPT_OUT_TRACKING` con il valore associato impostato `sutrue`. Per ulteriori informazioni, consulta Tagga le tue EC2 risorse Amazon.

Convalida della conformità per AWS Deep Learning AMIs

I revisori esterni valutano la sicurezza e la conformità di AWS Deep Learning AMIs come parte di più AWS programmi di conformità. Per informazioni sui programmi di conformità supportati, consulta Compliance Validation for Amazon EC2.

Per un elenco di AWS servizi che rientrano nell'ambito di specifici programmi di conformità, vedere AWS Servizi rientranti nell'ambito del programma di conformità . Per informazioni generali, vedere AWS Programmi di conformità di conformità.

È possibile scaricare report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta Scaricamento dei report in AWS Rapporti sul download di .

La responsabilità di conformità dell'utente durante l'utilizzo DLAMI è determinata dalla sensibilità dei dati, dagli obiettivi di conformità dell'azienda e dalle leggi e dai regolamenti applicabili. AWS fornisce le seguenti risorse per contribuire alla conformità:

- Guide rapide su sicurezza e conformità Guide introduttive implementazione illustrano considerazioni sull'architettura e forniscono passaggi per implementare ambienti di base incentrati sulla sicurezza e la conformità su AWS.
- AWS Risorse per la conformità : questa raccolta di cartelle di lavoro e guide potrebbe riguardare il tuo settore e la tua località.
- Valutazione delle risorse con regole in AWS Config Guida per gli sviluppatori: la AWS Config il servizio valuta la conformità delle configurazioni delle risorse alle pratiche interne, alle linee guida del settore e alle normative.
- AWS Security Hub— Questo AWS il servizio fornisce una visione completa dello stato di sicurezza all'interno AWS che consente di verificare la conformità agli standard e alle best practice del settore della sicurezza.

Resilienza in AWS Deep Learning AMIs

Il AWS l'infrastruttura globale è costruita attorno AWS Regioni e zone di disponibilità. AWS Le Regioni forniscono più zone di disponibilità fisicamente separate e isolate che sono connesse tramite reti altamente ridondanti, a bassa latenza e velocità di trasmissione effettiva elevata. Con le zone di disponibilità, puoi progettare e gestire applicazioni e database che eseguono automaticamente il failover tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili rispetto alle infrastrutture a data center singolo o multiplo tradizionali.

Per ulteriori informazioni sull' AWS Regioni e zone di disponibilità, vedi AWS Infrastruttura globale.

Per informazioni sulle funzionalità che aiutano a supportare le tue esigenze di resilienza e backup dei dati, consulta Resilience in Amazon. EC2

Sicurezza dell'infrastruttura in AWS Deep Learning AMIs

La sicurezza dell'infrastruttura di AWS Deep Learning AMIs è supportato da AmazonEC2. Per ulteriori informazioni, consulta la sezione Sicurezza dell'infrastruttura in Amazon EC2.

DLAMIpolitica di supporto quadro

Qui puoi trovare i dettagli della politica di supporto per AWS Deep Learning AMIs (DLAMI) framework.

Per un elenco dei framework che DLAMI AWS attualmente supporta, vedere la pagina DLAMIFramework Support Policy. Nelle tabelle di quella pagina, tieni presente quanto segue:

- La versione corrente specifica la versione del framework nel formato x.y.z. In questo formato, x si riferisce alla versione principale, y si riferisce alla versione secondaria e z si riferisce alla versione patch. Ad esempio, per TensorFlow 2.10.1, la versione principale è 2, la versione secondaria è 10 e la versione patch è 1.
- La fine della patch specifica per quanto tempo AWS supporta quella versione del framework.

Per informazioni dettagliate su specificheDLAMIs, vedereNote di rilascio per DLAMIs.

DLAMIsupporto del framework FAQs

- A quali versioni del framework vengono applicate le patch di sicurezza?
- A cosa servono le immagini AWS vengono pubblicate quando vengono rilasciate nuove versioni del framework?
- Quali immagini diventano nuove SageMaker/AWS caratteristiche?
- Come viene definita la versione corrente nella tabella Supported Frameworks?
- Cosa succede se utilizzo una versione che non è inclusa nella tabella Supported Frameworks?
- Sono DLAMIs supportate le versioni precedenti di TensorFlow?
- Come posso trovare l'ultima immagine con patch per una versione del framework supportata?
- Con che frequenza vengono rilasciate nuove immagini?
- La mia istanza verrà aggiornata mentre il mio carico di lavoro è in esecuzione?
- Cosa succede quando è disponibile una nuova versione del framework patchata o aggiornata?
- Le dipendenze vengono aggiornate senza modificare la versione del framework?
- Quando termina il supporto attivo per la mia versione del framework?
- Le immagini con versioni del framework che non vengono più gestite attivamente verranno patchate?
- Come posso usare una versione precedente del framework?

- Come posso attenermi up-to-date alle modifiche di supporto nei framework e nelle relative versioni?
- Ho bisogno di una licenza commerciale per utilizzare l'Anaconda Repository?

A quali versioni del framework vengono applicate le patch di sicurezza?

Se la versione del framework è etichettata come Supportata nel AWS Deep Learning AMIs Tabella Framework Support Policy, ottiene le patch di sicurezza.

A cosa servono le immagini AWS vengono pubblicate quando vengono rilasciate nuove versioni del framework?

Ne pubblichiamo di nuove DLAMIs subito dopo il rilascio TensorFlow e PyTorch il rilascio delle nuove versioni di. Sono incluse le versioni principali, le versioni principali e secondarie e le major-minor-patch versioni dei framework. Aggiorniamo le immagini anche quando diventano disponibili nuove versioni di driver e librerie. Per ulteriori informazioni sulla manutenzione delle immagini, vedere Quando termina il supporto attivo per la mia versione del framework?

Quali immagini diventano nuove SageMaker/AWS caratteristiche?

Le nuove funzionalità in genere vengono rilasciate nell'ultima versione di DLAMIs for PyTorch and TensorFlow. Consulta le note di rilascio per un'immagine specifica per i dettagli sulle nuove SageMaker o AWS caratteristiche. Per un elenco di quelle disponibiliDLAMIs, consulta le note di rilascio per DLAMI. Per ulteriori informazioni sulla manutenzione delle immagini, vedere Quando termina il supporto attivo per la mia versione del framework?

Come viene definita la versione corrente nella tabella Supported Frameworks?

La versione corrente in AWS Deep Learning AMIs La tabella Framework Support Policy si riferisce alla versione più recente del framework che AWS rende disponibile su. GitHub Ogni versione più recente include aggiornamenti ai driver, alle librerie e ai pacchetti pertinenti diDLAMI. Per informazioni sulla manutenzione delle immagini, vedere Quando termina il supporto attivo per la mia versione del framework?

Cosa succede se utilizzo una versione che non è inclusa nella tabella Supported Frameworks?

Se si esegue una versione che non è presente in AWS Deep Learning AMIs Tabella Framework Support Policy, potresti non avere i driver, le librerie e i pacchetti pertinenti più aggiornati. Per un'altra up-to-date versione, ti consigliamo di eseguire l'aggiornamento a uno dei framework supportati disponibili utilizzando l'ultima versione DLAMI di tua scelta. Per un elenco delle versioni disponibiliDLAMIs, consulta le note di rilascio per. DLAMI

Sono DLAMIs supportate le versioni precedenti di TensorFlow?

No. Supportiamo l'ultima versione patch dell'ultima versione principale di ogni framework rilasciata 365 giorni dopo il GitHub rilascio iniziale, come indicato nel AWS Deep Learning AMIs Tabella Framework Support Policy. Per ulteriori informazioni, consulta Cosa succede se utilizzo una versione che non è inclusa nella tabella Supported Frameworks?

Come posso trovare l'ultima immagine con patch per una versione del framework supportata?

Per utilizzare un DLAMI con l'ultima versione del framework, recupera l'DLAMIID e usalo per avviarlo DLAMI utilizzando la EC2 Console. Per esempio AWS CLIcomandi per recuperare il AWS Deep Learning AMIs ID, fai riferimento alla pagina delle note di DLAMI rilascio, note di DLAMIrilascio a framework singolo. La versione del framework scelta deve essere etichettata come Supportata nel AWS Deep Learning AMIs Tabella Framework Support Policy.

Con che frequenza vengono rilasciate nuove immagini?

Fornire versioni di patch aggiornate è la nostra massima priorità. Creiamo regolarmente immagini con patch non appena possibile. Monitoriamo la presenza di nuove versioni del framework con patch (ad es. TensorFlow da 2.9 a TensorFlow 2.9.1) e nuove versioni secondarie (es. TensorFlow da 2.9 a TensorFlow 2.10) e renderli disponibili il prima possibile. Quando TensorFlow viene rilasciata una versione esistente di con una nuova versione diCUDA, ne rilasciamo una nuova DLAMI per quella versione TensorFlow con supporto per la nuova CUDA versione.

La mia istanza verrà aggiornata mentre il mio carico di lavoro è in esecuzione?

No. Gli aggiornamenti delle patch per non DLAMI sono aggiornamenti «sul posto».

È necessario attivare una nuova EC2 istanza, migrare i carichi di lavoro e gli script e quindi disattivare l'istanza precedente.

Cosa succede quando è disponibile una nuova versione del framework patchata o aggiornata?

Controlla regolarmente la pagina delle note di rilascio della tua immagine. Ti consigliamo di eseguire l'aggiornamento a nuovi framework patchati o aggiornati non appena disponibili. Per un elenco di quelli disponibiliDLAMIs, consulta le note di rilascio per. DLAMI

Le dipendenze vengono aggiornate senza modificare la versione del framework?

Aggiorniamo le dipendenze senza modificare la versione del framework. Tuttavia, se un aggiornamento delle dipendenze causa un'incompatibilità, creiamo un'immagine con una versione diversa. Assicurati di controllare le note di rilascio DLAMI per informazioni aggiornate sulle dipendenze.

Quando termina il supporto attivo per la mia versione del framework?

DLAMIIe immagini sono immutabili. Una volta create non cambiano. Esistono quattro ragioni principali per cui il supporto attivo per una versione del framework termina:

- Aggiornamenti della versione del framework (patch)
- AWS patch di sicurezza
- Data di fine della patch (scadenza)
- Dipendenza end-of-support

> **Note**
>
> A causa della frequenza degli aggiornamenti delle patch di versione e delle patch di sicurezza, consigliamo di controllare DLAMI spesso la pagina delle note di rilascio e di aggiornare quando vengono apportate modifiche.

Aggiornamenti della versione del framework (patch)

Se disponi di un DLAMI carico di lavoro basato sulla versione TensorFlow 2.7.0 e TensorFlow versioni successive alla versione 2.7.1, allora GitHub AWS ne rilascia una nuova con 2.7.1. DLAMI TensorFlow Le immagini precedenti con 2.7.0 non vengono più mantenute attivamente una volta rilasciata la nuova immagine con TensorFlow 2.7.1. La versione DLAMI TensorFlow 2.7.0 non riceve ulteriori patch. La pagina delle note di DLAMI rilascio per la versione TensorFlow 2.7 viene quindi aggiornata con le informazioni più recenti. Non esiste una pagina delle note di rilascio individuale per ogni patch minore.

Le nuove DLAMIs create a seguito degli aggiornamenti delle patch vengono contrassegnate con un nuovo AMIID.

AWS patch di sicurezza

Se hai un carico di lavoro basato su un'immagine con TensorFlow 2.7.0 e AWS crea una patch di sicurezza, quindi DLAMI viene rilasciata una nuova versione di 2.7.0. TensorFlow La versione precedente delle immagini con TensorFlow 2.7.0 non viene più mantenuta attivamente. Per ulteriori informazioni, consulta La mia istanza verrà aggiornata mentre il mio carico di lavoro è in esecuzione? Per la procedura di ricerca delle versioni più recentiDLAMI, vedere Come posso trovare l'ultima immagine con patch per una versione del framework supportata?

I nuovi DLAMIs creati a seguito degli aggiornamenti delle patch vengono contrassegnati con un nuovo AMIID.

Data di fine della patch (scadenza)

DLAMIshanno raggiunto la data di fine della patch 365 giorni dopo la data di GitHub rilascio.

Per multi-framework DLAMIs, quando una delle versioni del framework viene aggiornata, è necessaria una nuova versione DLAMI con la versione aggiornata. La versione DLAMI con la vecchia versione del framework non viene più mantenuta attivamente.

> **Important**
>
> Facciamo un'eccezione quando c'è un importante aggiornamento del framework. Ad esempio, se la versione TensorFlow 1.15 viene aggiornata alla versione TensorFlow 2.0, continuiamo a supportare la versione più recente della TensorFlow 1.15 per un periodo di due anni dalla data di GitHub rilascio o sei mesi dopo la cessazione del supporto da parte del team di manutenzione del framework di origine, a seconda di quale data sia precedente.

Dipendenza end-of-support

Se stai eseguendo un carico di lavoro su un'immagine TensorFlow 2.7.0 DLAMI con Python 3.6 e quella versione di Python è contrassegnata per, tutte le immagini DLAMI basate su Python 3.6 non end-of-support verranno più gestite attivamente. Allo stesso modo, se una versione del sistema operativo come Ubuntu 16.04 è contrassegnata per end-of-support, tutte le DLAMI immagini che dipendono da Ubuntu 16.04 non verranno più gestite attivamente.

Le immagini con versioni del framework che non vengono più gestite attivamente verranno patchate?

No. Le immagini che non vengono più gestite attivamente non avranno nuove versioni.

Come posso usare una versione precedente del framework?

Per utilizzare una DLAMI versione del framework precedente, recupera l'DLAMIID e usalo per avviare l'DLAMIutilizzo della EC2Console. In AWS CLIcomandi per recuperare l'AMIID, fate riferimento alla pagina delle note di rilascio nelle note di rilascio del framework singolo DLAMI.

Come posso attenermi up-to-date alle modifiche di supporto nei framework e nelle relative versioni?

Rimani fedele up-to-date ai DLAMI framework e alle versioni utilizzando il AWS Deep Learning AMIs Tabella Framework Support Policy, note di DLAMI rilascio.

Ho bisogno di una licenza commerciale per utilizzare l'Anaconda Repository?

Anaconda è passata a un modello di licenza commerciale per determinati utenti. Mantenuti attivamente, sono DLAMIs stati migrati alla versione open source di Conda (conda-forge) disponibile al pubblico dal canale Anaconda.

Modifiche importanti del NVIDIA driver a DLAMIs

Il 15 novembre 2023, AWS ha apportato importanti modifiche a AWS Deep Learning AMIs (DLAMI) in relazione al NIVIDA driver DLAMIs utilizzato. Per informazioni su cosa è cambiato e se ciò ha influito sull'utilizzo diDLAMIs, consultaDLAMINVIDIAcambio di driver FAQs.

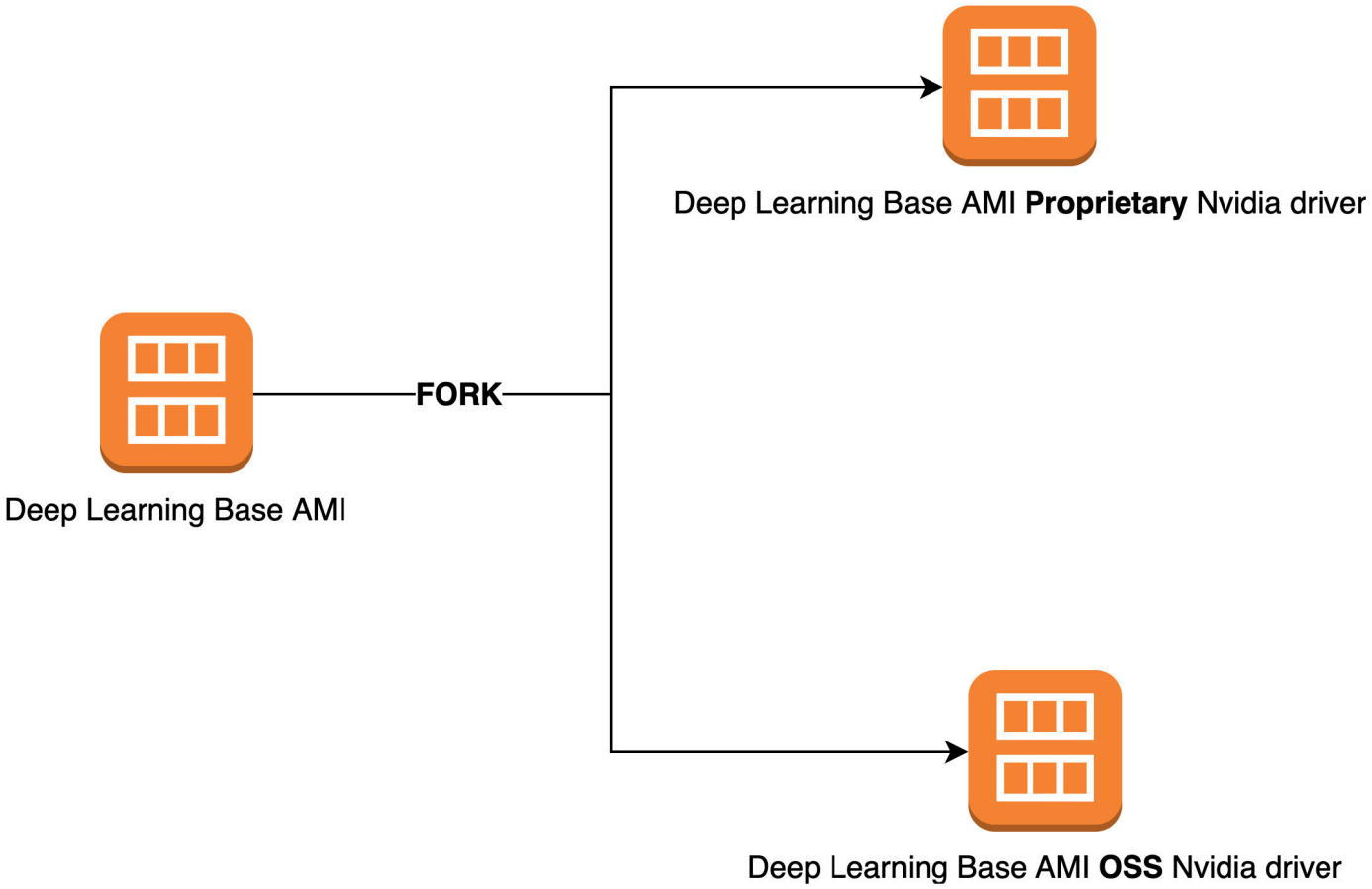
DLAMINVIDIAcambio di driver FAQs

Cosa è cambiato?

Ci siamo DLAMIs divisi in due gruppi separati:

- DLAMIsche utilizzano driver NVIDIA proprietari (per supportare P3, P3dn, G3)
- DLAMIsche usano il NVIDIA OSS driver (per supportare G4dn, G5, P4, P5)

Di conseguenza, ne abbiamo create di nuove DLAMIs per ciascuna delle due categorie con nuovi nomi e nuove. AMI IDs Non DLAMIs sono intercambiabili. Cioè, le istanze DLAMIs di un gruppo non supportano le istanze supportate dall'altro gruppo. Ad esempio, DLAMI quello che supporta P5 non supporta G3 e DLAMI quello che supporta G3 non supporta P5.



Perché è stata necessaria questa modifica?

In precedenza, DLAMIs for NVIDIA GPUs includeva un driver del kernel proprietario di. NVIDIA Tuttavia, la comunità del kernel Linux originale ha accettato una modifica che isola i driver proprietari del kernel, come il driver, dalla comunicazione con altri NVIDIA GPU driver del kernel. Questa modifica disabilita GPUDirect RDMA le istanze delle serie P4 e P5, che è il meccanismo che ne consente l'uso efficiente per l'addestramento distribuito. GPUs EFA Di conseguenza, DLAMIs ora utilizzate il driver OpenRM (driver NVIDIA open source), collegato ai driver open source per supportare G4dn, EFA G5, P4 e P5. Tuttavia, questo driver OpenRM non supporta le istanze precedenti (come P3 e G3). Pertanto, per continuare a fornire servizi aggiornati, performanti e sicuri DLAMIs che supportino entrambi i tipi di istanze, ci siamo DLAMIs divisi in due gruppi: uno con il driver OpenRM (che supporta G4dn, G5, P4 e P5) e uno con il driver proprietario precedente (che supporta P3, P3dn e G3).

Su DLAMIs cosa ha influito questa modifica?

Questa modifica ha influito su tuttiDLAMIs.

Cosa significa questo per te?

Tutti DLAMIs continueranno a fornire funzionalità, prestazioni e sicurezza fintanto che li eseguirai su un tipo di istanza Amazon Elastic Compute Cloud (AmazonEC2) supportato. Per determinare i tipi di EC2 istanza DLAMI supportati da a, consulta le relative note di rilascioDLAMI, quindi cerca le EC2istanze supportate. Per un elenco delle DLAMI opzioni attualmente supportate e i collegamenti alle relative note di rilascio, consultaNote di rilascio per DLAMIs.

Inoltre, è necessario utilizzare la versione corretta AWS Command Line Interface (AWS CLI) comandi per richiamare la correnteDLAMIs.

Per una base DLAMIs che supporta P3, P3dn e G3, usa questo comando:

```
aws ec2 describe-images --region us-east-1 --owners amazon \
--filters 'Name=name,Values=Deep Learning Base Proprietary Nvidia Driver AMI (Amazon
 Linux 2) Version ??.?' 'Name=state,Values=available' \
--query 'reverse(sort_by(Images, &CreationDate))[:1].ImageId' --output text
```

Per una base DLAMIs che supporta G4dn, G5, P4 e P5, usa questo comando:

```
aws ec2 describe-images --region us-east-1 --owners amazon \
--filters 'Name=name,Values=Deep Learning Base OSS Nvidia Driver AMI (Amazon Linux 2)
 Version ??.?' 'Name=state,Values=available' \
--query 'reverse(sort_by(Images, &CreationDate))[:1].ImageId' --output text
```

C'è qualche perdita di funzionalità con la versione più recente? DLAMIs

No, non vi è alcuna perdita di funzionalità. Le versioni correnti DLAMIs offrono tutte le funzionalità, le prestazioni e la sicurezza delle versioni precedentiDLAMIs, a condizione che vengano eseguite su un tipo di EC2 istanza supportato.

Questa modifica ha influito sui Deep Learning Containers?

No, questa modifica non ha influito AWS Deep Learning Containers, perché non includono il NVIDIA driver. Tuttavia, assicurati di eseguire Deep Learning Containers compatibili con le istanze sottostanti. AMIs

Informazioni correlate su DLAMI

Puoi trovare altre risorse con informazioni correlate DLAMI al di fuori del AWS Deep Learning AMIs Guida per gli sviluppatori. Abilitato AWS re:Post, dai un'occhiata alle domande poste DLAMI da altri clienti o poni le tue domande. Sul AWS Blog sul Machine Learning e altro AWS blog, leggi i post ufficiali in meritoDLAMI.

AWS re:Post

Etichetta: AWS Deep Learning AMIs

AWS Blog

- AWS Blog sul Machine Learning | Categoria: AWS Deep Learning AMIs
- AWS Blog sul Machine Learning | Formazione più rapida con TensorFlow 1.6 ottimizzato su istanze Amazon EC2 C5 e P3
- AWS Blog sul Machine Learning | Nuovo AWS Deep Learning AMIs per professionisti del Machine Learning
- AWS Partner Network (APN) Blog | Nuovi corsi di formazione disponibili: Introduzione al Machine Learning e al Deep Learning su AWS
- AWS Blog di notizie | Entra nel deep learning con AWS

Note di rilascio per DLAMIs

Qui puoi trovare note di rilascio dettagliate per tutte le versioni attualmente supportate AWS Deep Learning AMIs (DLAMI) opzioni.

Per le note di rilascio per i DLAMI framework che non supportiamo più, consulta la sezione Unsupported Framework Release Notes Archive della pagina Framework DLAMISupport Policy.

> **Note**
>
> Il AWS Deep Learning AMIs hanno una cadenza di rilascio notturna per le patch di sicurezza. Non includiamo queste patch di sicurezza incrementali nelle note di rilascio ufficiali.

Base DLAMIs

GPU

- X86
  - AWS Base di apprendimento approfondito AMI (Amazon Linux 2)
  - AWS Base di apprendimento profondo AMI (Ubuntu 22.04)
  - AWS Base di apprendimento profondo AMI (Ubuntu 20.04)
- ARM64
  - AWS Base di apprendimento profondo ARM64 AMI (Ubuntu 22.04)
  - AWS Base di apprendimento approfondito ARM64 AMI (Amazon Linux 2)

AWS Neurone

- X86
  - AWS Base di apprendimento profondo AMI Neuron (Amazon Linux 2)
  - AWS Base di apprendimento profondo AMI Neuron (Ubuntu 20.04)

Qualcomm

- X86

- AWS Base di apprendimento approfondito Qualcomm (AMIAmazon Linux 2)

Framework singolo DLAMIs

PyTorch-specifico AMIs

GPU

- X86
  - AWS Deep Learning AMI GPU PyTorch 2.3 (Ubuntu 20.04)
  - AWS Apprendimento profondo AMI GPU PyTorch 2.3 (Amazon Linux 2)
  - AWS Apprendimento profondo AMI GPU PyTorch 2.2 (Ubuntu 20.04)
  - AWS Apprendimento profondo AMI GPU PyTorch 2.2 (Amazon Linux 2)
  - AWS Apprendimento profondo AMI GPU PyTorch 1.13 (Amazon Linux 2)
  - AWS Apprendimento profondo AMI GPU PyTorch 1.13 (Ubuntu 20.04)
- ARM64
  - AWS Apprendimento profondo ARM64 AMI GPU PyTorch 2.3 (Ubuntu 22.04)
  - AWS Apprendimento profondo ARM64 AMI GPU PyTorch 2.2 (Ubuntu 20.04)

AWS Neurone

- X86
  - AWS Apprendimento profondo AMI Neuron PyTorch 1.13 (Amazon Linux 2)
  - AWS Deep Learning AMI Neuron PyTorch 1.13 (Ubuntu 20.04)

TensorFlow-specifico AMIs

GPU

- X86
  - AWS Apprendimento approfondito AMI GPU TensorFlow 2.16 (Amazon Linux 2)
  - AWS Apprendimento profondo AMI GPU TensorFlow 2.16 (Ubuntu 20.04)
  - AWS Apprendimento approfondito AMI GPU TensorFlow 2.15 (Amazon Linux 2)
  - AWS Apprendimento profondo AMI GPU TensorFlow 2.15 (Ubuntu 20.04)

AWS Neurone

- X86
  - AWS Apprendimento profondo AMI Neuron TensorFlow 2.10 (Amazon Linux 2)
  - AWS Deep Learning AMI Neuron TensorFlow 2.10 (Ubuntu 20.04)

Multi-framework DLAMIs

> **Tip**
>
> Se utilizzi solo un framework di machine learning, ti consigliamo un framework singolo DLAMI.

GPU

- X86
  - AWS Apprendimento approfondito AMI (Amazon Linux 2)

AWS Neurone

- X86
  - AWS Deep Learning AMI Neuron (Ubuntu 22.04)

Funzionalità obsolete di DLAMI

La tabella seguente elenca le funzionalità obsolete di AWS Deep Learning AMIs (DLAMI), la data in cui le abbiamo rese obsolete e dettagli sul motivo per cui le abbiamo rese obsolete.

Funzionalità	Data	Informazioni
Ubuntu 16.04	10/07/2021	Ubuntu Linux 16.04 LTS ha raggiunto la fine della sua LTS finestra quinquenn ale il 30 aprile 2021 e non è più supportato dal suo fornitore. A partire da ottobre 2021 non ci sono più aggiornamenti alla Deep Learning Base AMI (Ubuntu 16.04) nelle nuove versioni. Le versioni precedenti continueranno a essere disponibili.
Amazon Linux	10/07/2021	Amazon Linux è end-of-li feaggiornato a dicembre 2020. A partire da ottobre 2021 non ci sono più aggiornamenti al Deep Learning AMI (Amazon Linux) nelle nuove versioni. Le versioni precedent i di Deep Learning AMI (Amazon Linux) continuer anno a essere disponibili.
Chainer	01/07/2020	Chainer ha annunciat o la fine delle versioni principali a dicembre 2019. Di conseguenza, non

Funzionalità	Data	Informazioni
		includeremo più gli ambienti Chainer Conda DLAMI a partire da luglio 2020. Le versioni precedenti di DLAMI che contengono questi ambienti continuer anno a essere disponibili. Tuttavia, verranno forniti aggiornamenti a questi ambienti solo se sono disponibili correzioni di sicurezza pubblicate dalla comunità open source per questi framework.
Python 3.6	15/06/2020	A causa delle richieste dei clienti, stiamo passando a Python 3.7 per le nuove versioni TF/MX/PT.
Python 2	01/01/2020	La comunità open source di Python ha ufficialmente interrotto il supporto per Python 2. Le TensorFlow MXNet comunità e hanno anche annunciato che le versioni TensorFlow 1.15, TensorFlo w 2.1, PyTorch 1.4 e MXNet 1.6.0 saranno le ultime a supportare Python 2. PyTorch

Cronologia dei documenti per DLAMI

La tabella seguente fornisce una cronologia delle DLAMI versioni recenti e delle relative modifiche al AWS Deep Learning AMIs Guida per gli sviluppatori.

Modifiche recenti

Modifica	Descrizione	Data
ARM64 DLAMI	Il AWS Deep Learning AMIs ora supporta immagini basate su processori GPUs Arm64.	29 novembre 2021
TensorFlow 2	Il Deep Learning AMI con Conda ora include TensorFlow 2 con CUDA 10.	3 dicembre 2019
AWS Inferenza	Il Deep Learning ora supporta AMI AWS Hardware Inferentia e AWS Neurone. SDK	3 dicembre 2019
Utilizzo di TensorFlow Serving con un modello Inception	Un esempio di utilizzo dell'inferenza con un modello Inception è stato aggiunto per TensorFlow Serving, con e senza Elastic Inference.	28 novembre 2018
Elastic Inference	I prerequisiti di inferenza elastica e le informazioni correlate sono stati aggiunti alla guida all'installazione.	28 novembre 2018
Installazione PyTorch da una build notturna	È stato aggiunto un tutorial che spiega come PyTorch disinstallare e quindi installar e una build serale di Deep Learning con PyTorch Conda. AMI	25 settembre 2018

Tutorial Conda	L'esempio MOTD è stato aggiornato per riflettere una versione più recente.	23 luglio 2018

Modifiche precedenti

La tabella seguente fornisce una cronologia delle DLAMI versioni precedenti e delle relative modifiche precedenti a luglio 2018.

Modifica	Descrizione	Data
TensorFlow con Horovod	Aggiunto un tutorial per allenarsi ImageNet con TensorFlow e Horovod.	6 giugno 2018
Guida per l'upgrade	Aggiunta della guida per l'upgrade.	15 maggio 2018
Nuovo regioni e nuovo tutorial di 10 minuti	Aggiunta di nuove regioni: Stati Uniti occidentali (California settentrionale), Sud America, Canada (Centrale), UE (Londra) e UE (Parigi). Inoltre, la prima versione di un tutorial di 10 minuti intitolato: «Getting Started with Deep Learning». AMI	26 Aprile 2018
Tutorial di Chainer	È stato aggiunto un tutorial per l'utilizzo di Chainer in modalità multipla GPUGPU, singola e CPU in modalità. CUDAl'integrazione è stata aggiornata da CUDA 8 a CUDA 9 per diversi framework.	28 febbraio 2018

Modifica	Descrizione	Data
Linux AMIs v3.0, oltre all'introduzione di MXNet Model Server, Serving e TensorFlow TensorBoard	Sono stati aggiunti tutorial per Conda AMIs con nuove funzionalità di creazione di modelli e visualizzazioni utilizzando MXNet Model Server v0.1.5, Serving v1.4.0 e v0.4.0. TensorFlow TensorBoard AMICUDAe funzionalità del framework descritte in Conda e panoramiche. CUDA Note di rilascio più recenti spostate in https://aws.amazon.com/releasenotes/	25 gennaio 2018
Linux v2.0 AMIs	Base, Source e Conda AMIs aggiornati con NCCL 2.1. Source e Conda AMIs aggiornati con MXNet v1.0, PyTorch 0.3.0 e Keras 2.0.9.	11 dicembre 2017
AMISono state aggiunte due opzioni di Windows	AMIsRilasciati Windows 2012 R2 e 2016: aggiunti alla guida alla AMI selezione e aggiunti alle note di rilascio.	30 novembre 2017
Prima versione della documentazione	Descrizione dettagliata della modifica con link all'argomento/alla sezione oggetto della modifica.	15 novembre 2017

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.