

# VMware vSphere Bitfusion サンプル ガイド

2021 年 5 月 11 日

VMware vSphere Bitfusion 3.5 を含めるように更新

VMware vSphere Bitfusion 3.0

最新の技術ドキュメントは、VMware の Web サイト (<https://docs.vmware.com/jp/>)

**VMware, Inc.**  
3401 Hillview Ave.  
Palo Alto, CA 94304  
[www.vmware.com](http://www.vmware.com)

**VMware株式会社**  
105-0013 東京都港区浜松町 1-30-5  
浜松町スクエア 13F  
[www.vmware.com/jp](http://www.vmware.com/jp)

Copyright © 2020-2021 VMware, Inc. All rights reserved. [著作権および商標情報](#)。

# 目次

vSphere Bitfusion サンプル ガイド 4

**1** TensorFlow と vSphere Bitfusion の併用についての概要 5

**2** vSphere Bitfusion での TensorFlow のインストールと実行 6

NVIDIA CUDA のインストール 6

Ubuntu での NVIDIA CUDA のインストール 6

CentOS または Red Hat Linux での NVIDIA CUDA のインストール 7

NVIDIA cuDNN のインストール 8

CentOS または Red Hat Linux での Python のインストール 9

TensorFlow のインストール 10

TensorFlow ベンチマークのインストール 10

TensorFlow ベンチマークの実行 11

# vSphere Bitfusion サンプル ガイド

『vSphere Bitfusion サンプル ガイド』では、vSphere Bitfusion を使用して VMware vSphere で TensorFlow を実行する方法について説明しています。

VMware では、多様性の受け入れを尊重しています。お客様、パートナー企業、社内コミュニティとともにこの原則を推進することを目的として、多様性に配慮した言葉遣いでコンテンツを作成します。

『vSphere Bitfusion サンプル ガイド』では、TensorFlow およびオープンソース ベンチマークをインストールし、vSphere Bitfusion を使用してベンチマークを実行する方法について説明しています。このガイドは、vSphere Bitfusion で TensorFlow、およびその他の人工知能 (AI) や機械学習 (ML) のアプリケーションとフレームワークを使用する方法について理解するための基礎となります。

## 対象読者

この情報は、機械学習プラットフォームで vSphere Bitfusion を使用するユーザーを対象としています。ここに記載の情報は、Linux のシステム管理者としての経験があり、VMware vSphere を使用した仮想マシン テクノロジーおよびデータセンターの運用に詳しい方を想定しています。

# TensorFlow と vSphere Bitfusion の併用についての概要

# 1

TensorFlow を vSphere Bitfusion と使用するには、複数のコンポーネントをインストールして構成する必要があります。

TensorFlow を vSphere Bitfusion と使用するには、次のタスクを実行します。

- 1 vSphere Bitfusion をインストールします。『VMware vSphere Bitfusion インストール ガイド』を参照してください。
- 2 NVIDIA CUDA 11 をインストールします。
- 3 NVIDIA cuDNN 8 をインストールします。
- 4 CentOS または Red Hat Linux を使用している場合は、Python 3 をインストールする必要があります。
- 5 TensorFlow 2.4 をインストールします。
- 6 TensorFlow ベンチマークをインストールします。
- 7 TensorFlow ベンチマークを実行して、システムのパフォーマンスを測定します。

# vSphere Bitfusion での TensorFlow のインストールと実行

## 2

TensorFlow を vSphere Bitfusion と使用するには、いくつかのソフトウェア パッケージとプログラミング フレームワークをインストールして構成します。

この章には、次のトピックが含まれています。

- [NVIDIA CUDA のインストール](#)
- [NVIDIA cuDNN のインストール](#)
- [CentOS または Red Hat Linux での Python のインストール](#)
- [TensorFlow のインストール](#)
- [TensorFlow ベンチマークのインストール](#)
- [TensorFlow ベンチマークの実行](#)

## NVIDIA CUDA のインストール

CUDA は、グラフィック処理ユニット (GPU) での一般的なコンピューティング用に NVIDIA が開発した並列コンピューティング プラットフォームおよびプログラミング モデルです。CUDA は、GPU の処理能力を使用することで、コンピューティング アプリケーションを大幅に高速化します。CUDA は、TensorFlow ベンチマークで使用されます。

## Ubuntu での NVIDIA CUDA のインストール

Ubuntu Linux に CUDA をインストールすることができます。

Ubuntu オペレーティング システムに vSphere Bitfusion クライアントがインストールされていることを確認します。

### 手順

- 1 NVIDIA CUDA ディストリビューションのダウンロード先となる仮想マシン上のディレクトリに移動します。

```
cd <download_directory>
```

- 2 `cuda-ubuntu2004.pin` ファイルをダウンロードして、移動します。

```
wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin
sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-pin-600
```

- 3 `wget` コマンドを使用して、Ubuntu 20.04 用の NVIDIA CUDA ディストリビューションをダウンロードします。

```
wget <https://developer.download.nvidia.com/compute/cuda/11.0.3/local_installers/cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb>
```

- 4 `dpkg -i` コマンドを使用して、Ubuntu 20.04 用の CUDA 11 パッケージをインストールします。

```
sudo dpkg -i cuda-repo-ubuntu2004-11-0-local_11.0.3-450.51.06-1_amd64.deb
```

- 5 `apt-key` コマンドを使用して、ソフトウェア パッケージの認証用キーをインストールします。

`apt-key` コマンドは、`apt` がパッケージの認証で使用するキーのリストを管理します。これらのキーを使用して認証したパッケージは、信頼されているとみなされます。

```
sudo apt-key add /var/cuda-repo-ubuntu2004-11-0-local/7fa2af80.pub
```

- 6 CUDA ソフトウェア パッケージをアップデートおよびインストールします。

```
sudo apt-get update
sudo apt-get install cuda
```

- 7 (オプション) GPU パーティション サイズの確認、または vSphere Bitfusion デプロイ環境で使用可能なリソースの確認を実行するには、NVIDIA System Management Interface (`nvidia-smi`) 監視アプリケーションを実行します。

```
bitfusion run -n 1 nvidia-smi
```

- 8 CUDA Matrix Multiplication (`matrixMul`) のサンプル ファイルが格納されているディレクトリに移動します。

```
cd /usr/local/cuda/samples/0_Simple/matrixMul
```

- 9 `matrixMul` のサンプル ファイルに対して `make` コマンドと `bitfusion run` コマンドを実行します。

```
sudo make
bitfusion run -n 1 ./matrixMul
```

## 次のステップ

NVIDIA cuDNN をインストールして構成します。[NVIDIA cuDNN のインストール](#)を参照してください。

## CentOS または Red Hat Linux での NVIDIA CUDA のインストール

CUDA 11 は CentOS 8 または Red Hat Linux 8 にインストールできます。

## 手順

- 1 NVIDIA CUDA ディストリビューションのダウンロード先となる仮想マシン上のディレクトリに移動します。

```
cd <download_directory>
```

- 2 CentOS 8 または Red Hat Linux 8 用の NVIDIA CUDA 11 パッケージをダウンロードするには、`wget` コマンドを実行します。

```
wget https://developer.download.nvidia.com/compute/cuda/11.0.3/local_installers/cuda-repo-  
rhel8-11-0-local-11.0.3_450.51.06-1.x86_64.rpm
```

- 3 CUDA パッケージをインストールするには、`rpm -i` コマンドを実行します。

```
sudo rpm -i cuda-repo-rhel8-11-0-local-11.0.3_450.51.06-1.x86_64.rpm
```

- 4 コマンド `yum clean all` と `yum -y install` を実行して、環境を更新し、CUDA ソフトウェア パッケージをインストールします。

```
sudo yum clean all  
sudo yum -y install cuda
```

- 5 (オプション) GPU パーティション サイズの確認、または vSphere Bitfusion デプロイ環境で使用可能なリソースの確認を実行するには、NVIDIA System Management Interface (`nvidia-smi`) 監視アプリケーションを実行します。

```
bitfusion run -n 1 nvidia-smi
```

- 6 CUDA Matrix Multiplication (`matrixMul`) のサンプル ファイルが格納されているディレクトリに移動します。

```
cd /usr/local/cuda/samples/0_Simple/matrixMul
```

- 7 `matrixMul` のサンプル ファイルに対してコマンド `make` と `bitfusion run` を実行します。

```
sudo make  
bitfusion run -n 1 ./matrixMul
```

## 次のステップ

NVIDIA cuDNN をインストールして構成します。[NVIDIA cuDNN のインストール](#)を参照してください。

## NVIDIA cuDNN のインストール

cuDNN は、ディープ ニューラル ネットワークでの使用を目的とした、GPU アクセラレーション型のプリミティブのライブラリです。



## 前提条件

使用している NVIDIA CUDA バージョンと、Linux ディストリビューションに基づいて、cuDNN パッケージのダウンロード元となる NVIDIA 開発者アカウントを作成します。<https://developer.nvidia.com/cudnn> を参照してください。

## 手順

- 1 使用している Linux のディストリビューションのコマンド シーケンスを実行し、cuDNN パッケージをインストールします。

- ◆ Ubuntu バージョン 20.04

```
sudo dpkg -i libcudnn8_8.0.5.39-1+cuda11.0-amd64.deb
```

- ◆ CentOS 8 および Red Hat Linux 8

```
sudo rpm -ivh libcudnn8-8.0.5.39-1.cuda11.0.x86_64.rpm
```

- 2 cuDNN がインストールされていることを確認するには、`ldconfig -p | grep cudnn` を実行します。

## 次のステップ

- CentOS または Red Hat Linux を使用している場合は、Python 3 をインストールする必要があります。[CentOS または Red Hat Linux での Python のインストール](#)を参照してください。
- Ubuntu Linux を使用している場合は、TensorFlow をインストールできます。[TensorFlow のインストール](#)を参照してください。

# CentOS または Red Hat Linux での Python のインストール

CentOS および Red Hat Linux を使用している場合は、Python 3 をインストールする必要があります。

Ubuntu を使用している場合は、この手順を実行する必要はありません。Ubuntu には Python 3 が事前にインストールされています。

## 手順

- 1 `yum update` コマンドを実行し、現在インストールされているすべてのパッケージをアップデートします。

```
sudo yum update
```

- 2 Python 3 をインストールするには、`dnf` コマンドを実行します。

```
sudo dnf install python3
```

- 3 `python3 -V` コマンドで Python 3 を使用していることを確認します。

```
python3 -V  
Python 3.6.8
```

- 4 (オプション) 環境のスナップショットを作成します。

### 次のステップ

TensorFlow をインストールします。[TensorFlow のインストール](#)を参照してください。

## TensorFlow のインストール

TensorFlow は、Bitfusion で使用する機械学習 (ML) フレームワークです。

TensorFlow は、Python 3 のパッケージ インストーラである `pip3` を使用してインストールします。

### 手順

- 1 Ubuntu 20.04 に TensorFlow をインストールする場合は、追加の Python リソースをインストールします。

```
sudo apt-get -y install python3-testresources
```

- 2 使用している Linux のディストリビューションとバージョン用のコマンド シーケンスを実行し、`pip3` をインストールします。

- Ubuntu 20.04

```
sudo apt-get install -y python3-pip
```

- CentOS 8 および Red Hat Linux 8

```
sudo yum install -y python36-devel  
sudo pip3 install -U pip setuptools
```

- 3 `pip3 install` コマンドを使用して、TensorFlow をインストールします。

```
sudo pip3 install tensorflow-gpu==2.4
```

### 次のステップ

TensorFlow ベンチマークを実行をして vSphere Bitfusion デプロイ環境のパフォーマンスをテストできます。[TensorFlow ベンチマークのインストール](#)を参照してください。

## TensorFlow ベンチマークのインストール

TensorFlow ベンチマークは、オープンソースの機械学習 (ML) アプリケーションで、TensorFlow フレームワークのパフォーマンス テストを目的として設計されています。

TensorFlow ベンチマークは、ローカル環境に対してブランチおよびダウンロードします。Git では、ブランチは開発の別ラインです。

### 手順

- 1 Git をインストールします。

```
sudo yum -y update  
sudo yum install git
```

- 2 ~/bitfusion を作成して、作業ディレクトリにします。

```
mkdir bitfusion
cd ~/bitfusion
```

- 3 Tensorflow ベンチマークの Git リポジトリのクローンをローカル環境に作成します。

```
git clone https://github.com/tensorflow/benchmarks.git
```

- 4 ベンチマーク ディレクトリに移動し、リポジトリのブランチを一覧表示します。

```
cd benchmarks
git branch -a
```

```
master
remotes/origin/HEAD -> origin/master
...
remotes/origin/cnn_tf_v1.13_compatible
...
remotes/origin/cnn_tf_v2.1_compatible
...
```

- 5 Git チェックアウトを実行し、TensorFlow ベンチマーク リポジトリを一覧表示します。

```
git checkout cnn_tf_v2.1_compatible
```

```
Branch cnn_tf_v2.1_compatible set up to track remote branch cnn_tf_v2.1_compatible
from origin.
Switched to a new branch 'cnn_tf_v2.1_compatible'
```

```
git branch
```

```
cnn_tf_tf_v2.1_compatible
master
```

#### 次のステップ

TensorFlow ベンチマークを実行をして vSphere Bitfusion デプロイ環境のパフォーマンスをテストできます。  
[TensorFlow ベンチマークの実行](#)を参照してください。

## TensorFlow ベンチマークの実行

vSphere Bitfusion と TensorFlow デプロイ環境のパフォーマンスをテストするには TensorFlow ベンチマークを実行します。

さまざまな構成を使用して TensorFlow ベンチマークを実行することで、使用している vSphere Bitfusion 環境での ML ワークロードの応答方法を把握できます。

## 手順

- 1 ~/bitfusion/ ディレクトリに移動するには、`cd ~/bitfusion/` を実行します。

- 2 `tf_cnn_benchmarks.py` ベンチマーク スクリプトを使用するには、`bitfusion run` コマンドを実行します。

この例のコマンドを実行すると、単一 GPU のメモリ全体と、/data ディレクトリに事前にインストールされた ML データが使用されます。

```
bitfusion run -n 1 -- python3 \
./benchmarks/scripts/tf_cnn_benchmarks/tf_cnn_benchmarks.py \
--data_format=NCHW \
--batch_size=64 \
--model=resnet50 \
--variable_update=replicated \
--local_parameter_device=gpu \
--nodistortions \
--num_gpus=1 \
--num_batches=100 \
--data_dir=/data \
--data_name=imagenet \
--use_fp16=False
```

- 3 `tf_cnn_benchmarks.py` ベンチマーク スクリプトを使用するには、`-p 0.67` パラメータを指定して `bitfusion run` コマンドを実行します。

この例のコマンドを実行すると、単一 GPU のメモリの 67% と、/data ディレクトリに事前にインストールされた ML データが使用されます。`-p 0.67` パラメータを使用すると、GPU のメモリ パーティションの残りの 33% で、別のジョブを実行できます。

```
bitfusion run -n 1 -p 0.67 -- python3 \
./benchmarks/scripts/tf_cnn_benchmarks/tf_cnn_benchmarks.py \
--data_format=NCHW \
--batch_size=64 \
--model=resnet50 \
--variable_update=replicated \
--local_parameter_device=gpu \
--nodistortions \
--num_gpus=1 \
--num_batches=100 \
--data_dir=/data \
--data_name=imagenet \
--use_fp16=False
```

- 4 `tf_cnn_benchmarks.py` ベンチマーク スクリプトを使用するには、合成されたデータを使用して `bitfusion run` コマンドを実行します。

この例のコマンドを実行すると、単一 GPU のメモリ全体が使用され、事前にインストールされた ML データは使用されません。TensorFlow は、イメージ セットのようなものを使用して、合成されたデータを作成できます。

```
bitfusion run -n 1 -- python3 \
./benchmarks/scripts/tf_cnn_benchmarks/tf_cnn_benchmarks.py \
--data_format=NCHW \
```

```
--batch_size=64 \  
--model=resnet50 \  
--variable_update=replicated \  
--local_parameter_device=gpu \  
--nodistortions \  
--num_gpus=1 \  
--num_batches=100 \  
--use_fp16=False
```

## 結果

リモート サーバから、GPU を共有している vSphere Bitfusion に TensorFlow ベンチマークを実行できるようになりました。ベンチマークがサポートするさまざまなモデルとパラメータは、規模の大きい調査での機械学習に役立ちます。詳細については、『VMware vSphere Bitfusion ユーザー ガイド』を参照してください。