

# vSphere Bitfusion での TensorFlow の実行

2020 年 11 月 5 日

VMware vSphere Bitfusion 2.5

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)

**ヴィエムウェア株式会社**  
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 VMware, Inc. All rights reserved. [著作権および商標情報。](#)

# 目次

『vSphere Bitfusion での TensorFlow の実行』について	4
<b>1 TensorFlow と vSphere Bitfusion の併用についての概要</b>	<b>5</b>
<b>2 vSphere Bitfusion での TensorFlow のインストールと実行</b>	<b>6</b>
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	8
NVIDIA cuDNN のインストール	9
CentOS または Red Hat Linux での Python3 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	11
TensorFlow ベンチマークの実行	12

# 『vSphere Bitfusion での TensorFlow の実行』 について

『vSphere Bitfusion での TensorFlow の実行』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。

『vSphere Bitfusion での TensorFlow の実行』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、vSphere Bitfusion をインストール、アップグレード、または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。vSphere Bitfusion のインストールについては、『VMware vSphere Bitfusion インストール ガイド』を参照してください。
- 2 NVIDIA CUDA 10.0 をインストールします。
- 3 NVIDIA cuDNN 7 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 をインストールします。
- 5 TensorFlow 1.13.1 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 Python3 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。

### 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 使用している Ubuntu バージョン用の NVIDIA CUDA ディストリビューションを、`wget` コマンドを使用してダウンロードします。

Ubuntu の各バージョンに適切な NVIDIA CUDA パッケージは、NVIDIA ダウンロード サイトからダウンロードできます。

オプション	説明
<b>Ubuntu 16.04</b>	<a href="https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/cuda-repo-ubuntu1604_10.0.130-1_amd64.deb">https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/cuda-repo-ubuntu1604_10.0.130-1_amd64.deb</a>
<b>Ubuntu 18.04</b>	<a href="https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64/cuda-repo-ubuntu1804_10.0.130-1_amd64.deb">https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64/cuda-repo-ubuntu1804_10.0.130-1_amd64.deb</a>

```
wget <NVIDIA_CUDA_download_URL>
```

- 3 使用している Ubuntu バージョン用の CUDA パッケージを、`dpkg -i` コマンドを使用してインストールします。

```
sudo dpkg -i cuda-repo-ubuntu1804_10.0.130-1_amd64.deb
```

- 4 `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。使用している Ubuntu バージョン用のキーを指定します。

オプション	説明
<b>Ubuntu 16.04</b>	<a href="https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/7fa2af80.pub">https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1604/x86_64/7fa2af80.pub</a>
<b>Ubuntu 18.04</b>	<a href="https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64/7fa2af80.pub">https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64/7fa2af80.pub</a>

```
sudo apt-key adv --fetch-keys <authentication_key_URL>
```

- 5 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-10.0
```

- 6 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
```

- 7 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
```

- 8 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。

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

CUDA は CentOS または Red Hat Linux 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 使用している Linux バージョン用の NVIDIA CUDA パッケージを、`wget` コマンドを使用してダウンロードします。

```
wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64/cuda-repo-ubuntu1804_10.0.130-1_amd64.deb
```

- 3 `rpm -i` コマンドを使用して CUDA パッケージをインストールします。

```
sudo rpm -i cuda-repo-rhel7-10.0.130-1.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-10-0
```

- 5 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とバージョンに基づいて、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

- ◆ 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、cuDNN パッケージをインストールします。

- Ubuntu バージョン 16.04 または 18.04

```
sudo dpkg -i libcudnn7_7.6.5.32-1+cuda10.0_amd64.deb
# Verify it is installed
ldconfig -p | grep cudnn
```

- CentOS および Red Hat Linux

```
sudo rpm -ivh libcudnn7-7.6.5.32-1.cuda10.0.x86_64.rpm sudo
# Update libraries list
ldconfig
# Verify that cuDNN is installed
ldconfig -p | grep cudnn
```

## 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 Python3 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 Python3 のインストール

CentOS および Red Hat Linux では、Python 3 が環境にインストールされていない場合、インストールする必要があります。

Python 3 は、Software Collection 内にインストールします。Software Collection は、システム全体にインストールされているパッケージに影響を与えずに、同一システム上の複数のバージョンのソフトウェアをビルド、インストール、および使用するための機能です。

## 前提条件

Ubuntu に Python 3 が事前にインストールされている場合は、Ubuntu システムでこの手順を実行する必要はありません。

## 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 scl コマンドを使用して、Python 3 を実行する Software Collection 環境を作成します。

```
sudo yum install -y centos-release-scl
sudo yum install -y rh-python36
```

- 3 Software Collection 環境で Python 3 を実行するには、scl enable コマンドを実行します。

インストールする Python のバージョンを指定し、bash コマンドを実行して、インタラクティブな Bash シェル セッションを作成します。scl enable コマンドで環境を構成し、PATH 環境変数に Python 3 を追加します。

```
scl enable rh-python36 bash
```

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

```
python -V
Python 3.6.3
```

- 5 (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 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 バージョン 16.04 または 18.04

```
sudo apt-get install -y python3-pip
sudo pip3 install absl-py
```

## ■ CentOS および Red Hat Linux

```
sudo yum install -y python36-devel
sudo yum install -y python36-pip
```

## 2 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==1.13.1
pip3 list
```

### 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 ~/bitfusion を作業ディレクトリにします。

```
cd ~/bitfusion
```

#### 2 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
```

#### 3 ベンチマーク ディレクトリに移動し、次のコマンド シーケンスを使用してリポジトリをブランチします。

```
cd benchmarks
git branch -a
* master
remotes/origin/HEAD -> origin/master
remotes/origin/cnn_tf_v1.10_compatible
...
remotes/origin/cnn_tf_v1.13_compatible
...
```

#### 4 次のコマンド シーケンスを使用して、Git checkout を実行し、TensorFlow ベンチマーク リポジトリをリストします。

```
git checkout cnn_tf_v1.13_compatible
Branch cnn_tf_v1.13_compatible set up to track remote branch cnn_tf_v1.13_compatible
```

```
from origin.
Switched to a new branch 'cnn_tf_v1.13_compatible'
git branch
* cnn_tf_v1.13_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 ベンチマークを使用することで、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 ディレクトリ /data の事前インストールされたデータを使用して、ベンチマークを実行します。

- a ~/bitfusion/ ディレクトリに移動します。

```
cd ~/bitfusion/
```

- b 以下のように、bitfusion run コマンドを使用して tf\_cnn\_benchmarks.py スクリプトを実行します。  
 これにより、1つの GPU でベンチマークが実行されます (-n 1)。ただし、GPU をパーティション分割しないため、GPU メモリの 100% サイズのパーティションで実行されます。

```
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
```

- 2 -p 0.67 パラメータを指定して bitfusion run コマンドを使用して tf\_cnn\_benchmarks.py スクリプトを実行します。

-p 0.67 パラメータは、GPU のメモリの 67% のみを使用するように指定します。これにより、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
```

### 3 合成されたデータを使用してベンチマークを実行します。

bitfusion run コマンドを使用して tf\_cnn\_benchmarks.py スクリプトを実行します。

```
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 サーバから共有 GPU を提供する 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。また、vSphere Bitfusion で他のフレームワークや ML アプリケーションをインストールして実行するための知識と経験が身に付きました。