Windows WSL2におけるTensorflowベースの ディープラーニング用サーバの構築

常三島技術部門
計測制御システムグループ　　北島　孝弘 (KITAJIMA Takahiro)
大学院社会産業理工学研究部
理工学域 電気電子系　　安野　卓 (YASUNO Takashi)
鈴木　浩司 (SUZUKI Hiroshi)

Keywords: Deep-learning, Tensorflow, Windows, WSL2, Python

１．はじめに

ディープラーニングのフレームワークであるTensorflowのGPU版は，ver. 2.10でWindows用のリリースを終了した。それ以降のGPU版TensorflowをWindowsで利用するためには，Windows上でLinuxの仮想環境を構築できるWSL2 （Windows Subsystem for Linux 2）の利用が推奨されている。WSL2はWindows11で標準の機能として提供されている。本稿では，LinuxおよびTensorflowで構成されるディープラーニングサーバをWindows上で構築する手法について紹介する。

２．WSL2のインストール

WSL を使用すると，Windows と Linux を同時に利用できることに加え，OS 間でファイルのやりとりもできる。Windows11 であれば，コマンドプロンプトを管理者権限で起動して，下記のコマンドを入力するとWSLが有効化され，Linux（Ubuntu）がインストールされる。インストール後，画面の指示に従い再起動する。なお，GPU のドライバは最新版を事前にインストールしておく。

```
wsl --install
```

このとき，インストールされる Linux のバージョンは最新版 LTS となる。下記コマンドを実行すると，利用可能な Linux ディストリビューションの一覧が表示される。

```
wsl --list --online
```

ディストリビューションを指定してインストールする場合は，下記コマンドを実行する。

```
wsl --install -d “ディストリビューション名”
```

インストールが完了すると，Linux のユーザ名とパスワードを設定するように表示されるので，任意のものを入力する。なお，インストールされたディストリビューションは，コマンドプロンプト上で下記コマンドにより確認できる。

```
wsl --list --verbose
```

wsl の終了，起動は下記コマンドで実行できる。

```
wsl --shutdown　　（終了）
wsl　　（起動）
```

インストール後は，下記コマンドを実行してLinux のアップデートが可能である。

```
sudo apt update
sudo apt upgrade
```

３．CUDA Toolkitのインストール

あらかじめインストールしたいTensorflowのバージョンに対応するCUDA Toolkit，cuDNN，Pythonのバージョンを Tensorflowのwebページ[1]で確認しておく。そして，Nvidiaのwebページ[2]よりインストールしたいCUDA Toolkitのバージョンを選択する。このとき，Target Platformの選択は下記のように指定する（ここでは ver. 11.2.2を選択）。

Operating System	Linux
Architecture	x86_64
Distribution	WSL-Ubuntu
Version	2.0
Installer Type	deb(local)

そして，ページ下部の「Base Installer」欄に表示されるインストールコマンドを１行ずつLinuxのターミナルで実行する。

４．cuDNNのインストール

Nvidiaのwebページ[3]より，インストールしたいバージョンのcuDNN Runtime LibraryをCPUアーキテクチャとUbuntuのバージョンで絞込み，ダウンロード，Ubuntuのホームディレクトリへ保存する。このとき，Nvidiaへのログインが必要になるので，アカウントを持っていない場合は作成する。ダウンロードしたファイルのディレクトリへ移動し，下記コマンドを実行してインストールする（ここでは，ver. 8.1を選択）。

```
sudo dpkg -i “cuDNNファイル名.deb”
```

５．環境変数の設定

下記の内容を「~/.bashrc」に追記する。

```
export PATH=”/usr/local/cuda/bin:$PATH”
export LD_LIBRARY_PATH=”/usr/local/cuda/lib64:$LD_LIBRARY_PATH”
```

そして，下記コマンドを実行して変更を反映させる。

```
source ~/.bashrc
```

６．Pythonのインストール

Ubuntuをインストールすると，Pythonも同時にインストールされているが，ユーザのプログラミング環境として，Pythonを別途インストールする。インストールは，Python Japanホームページ[4]のPython環境構築ガイド（Ubuntu）の手順に沿って行う。

７．Pythonの仮想環境を構築

Pythonをインストールすると，複数のPython実行環境を構築するための仮想環境モジュールvenvも利用可能になっている。仮想環境を利用すると，異なるバージョンのPythonやライブラリの組み合わせでそれぞれ実行環境を構築することができる。機械学習の実行環境の作成においては，ライブラリ間のバージョンの相性で正常に動作しないことが多いので，実行環境を簡単に作成したり，削除したりできる仮想環境は便利である。仮想環境の構築は，以下のコマンドをホームディレクトリに移動してから実行する。“py309”には任意の名前を指定する。

```
mkdir venv
cd venv
python3.9 -m venv py309
```

仮想環境へ出入りするには，以下のコマンドを実行する。

```
仮想環境に入る時
source ~/venv/py309/bin/activate
仮想環境から出る時
deactivate
```

８．Tensorflowのインストール

作成した仮想環境に入って，以下のコマンドを実行する。以下の例は，Tensorflowのバージョンが2.11.0の場合である。

```
pip install tensorflow==2.11.0
```

９．Visual Studio Codeとの連携

Visual Studio Codeの拡張機能「Remote Development」をインストールすると，WSL2上のPython実行環境と接続できるようになる。インストール後，Visual Studio Codeを閉じ，WSL2のUbuntuターミナルで以下のコマンドを実行すると，Visual Studio Codeの起動と同時に仮想環境に接続できる。

```
cd venv/py309
code .
```

10．おわりに

本稿は執筆時点での情報をまとめたものである。今後の更新により構築手順やコマンドは変更される可能性があるため，最新情報はインターネット等での確認をお勧めする。

参考文献

[1] https://www.tensorflow.org/install/source?hl=ja#gpu

[2] https://developer.nvidia.com/cuda-toolkit-archive

[3] https://developer.nvidia.com/rdp/cudnn-archive

[4] https://www.python.jp/