

# Lecture 0

## Introduction to TensorFlow

明玉瑞 Yurui Ming  
yrming@gmail.com

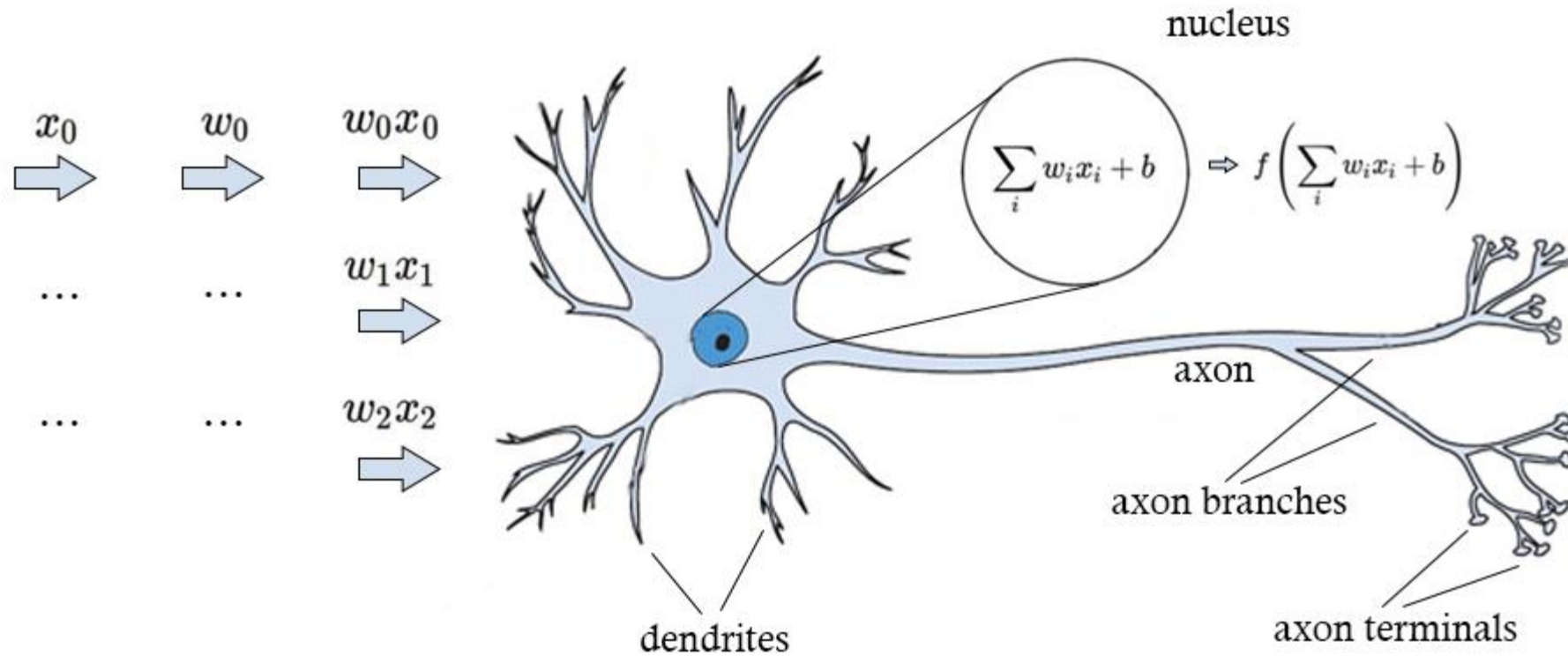
# 声明

## Disclaimer

- 本讲义在准备过程中由于时间所限，所用材料来源并未规范标示引用来源。所引材料仅用于教学所用，作者无意侵犯原著者之知识产权，所引材料之知识产权均归原著者所有；若原著者介意之，请联系作者更正及删除。

The time limit for the preparation of these slides incurs the situation that not all the sources of the used materials (texts or images) are properly referenced or clearly manifested. However, all materials in these slides are solely for teaching and the author is with no intention to infringe the copyright bestowed on the original authors or manufacturers. All credits go to corresponding IP holders. Please address the author for remedy including deletion have you had any concern.

# Motivation



$$y = \varphi\left(\sum_{i=1}^n w_i x_i + b\right) = \varphi(\mathbf{w}^T \mathbf{x} + b)$$

# Motivation

- 许多复杂的数学运算可以被分解为一系列标量运算。

Many complex math operations can be decomposed into a series of scalar operations.

- 一些运算以向量形式表示时，非常简洁且易于理解。

Vectorization of scalar operations can result in succinct representation which can be easily understood.

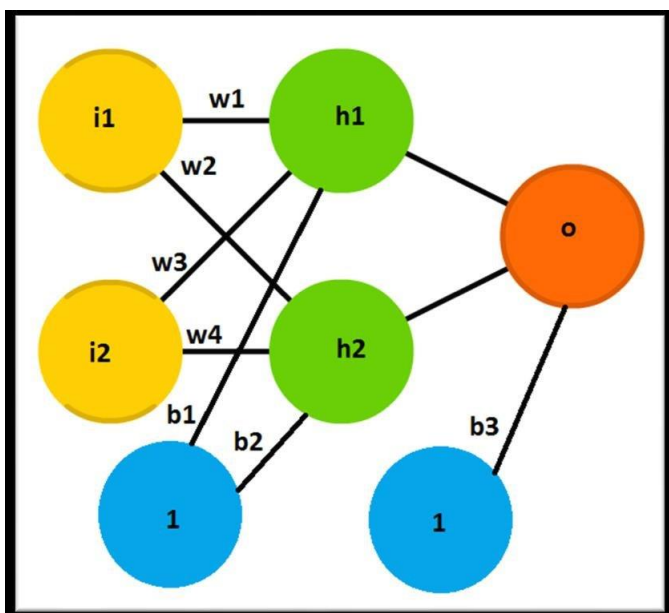
- 现代许多硬件可以原生地执行向量运算。

Many contemporary hardware are capable of doing vectorized operations.

# Motivation

- Python的数值运算库Numpy在一定程度上解决了上述问题。

The numerical computation library Numpy of Python to some extent some this problem.



```
import numpy as np

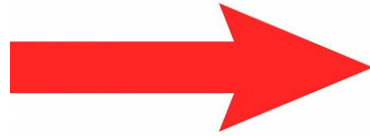
def predict(params, inputs):
    for W, b in params:
        outputs = np.dot(inputs, W) + b
        outputs = np.tanh(outputs)
    return outputs

def loss(params, batch):
    inputs, targets = batch
    preds = predict(params, inputs)
    return np.sum((preds - targets) ** 2)
```

# Motivation

- ▶ 上述利用原生的Numpy的方式能表面上解决问题，但并非最优，因为缺乏如下一些特性：
  - ▶ 基于GPU或TPU的硬件加速
  - ▶ 优化的自动微分机制
  - ▶ 编译优化的聚合、内存布局、重整
  - ▶ 向量化的批处理操作
  - ▶ 数据与操作的并行化
- ▶ Utilizing the original Numpy can solve the problem superficially, however, due to lacks of features below, it is not the optimal way:
  - ▶ Accelerator hardware (GPU/TPU)
  - ▶ Fast optimization via autodiff
  - ▶ Optimized compilation with fusion, memory layout, remat, ...
  - ▶ Vectorized batching of operations
  - ▶ Parallelization of data & computations

# Motivating TensorFlow



# TensorFlow

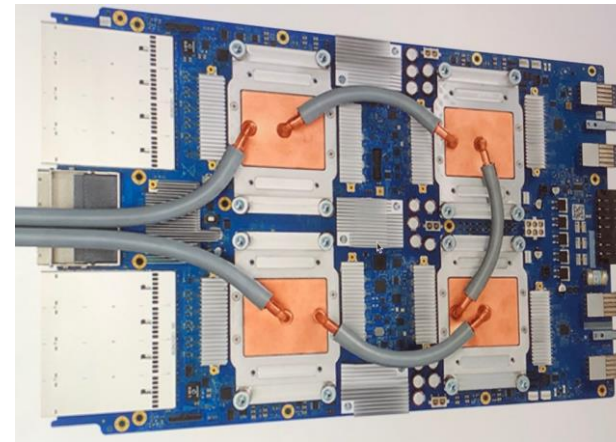
GPU/TPU

autodiff

batching

compilation

parallelization

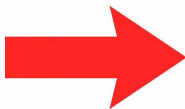


# Motivating TensorFlow

```
import numpy as np

def predict(params, inputs):
    for W, b in params:
        outputs = np.dot(inputs, W) + b
        outputs = np.tanh(outputs)
    return outputs

def loss(params, batch):
    inputs, targets = batch
    preds = predict(params, inputs)
    return np.sum((preds - targets) ** 2)
```

[illegible]



# TensorFlow

- TensorFlow最初的构想是为基于神经的网络或深度学习的人工智能探究提供一个便捷的编程平台，尽管最初TensorFlow更像类似一个基于库的神经网络计算工具，谷歌也将其定位是一个通用高性能数值计算库。但实际上，随着它支持的主要特性增多，包括Keras，其逐渐具有适配从初学者到专家，可以构建和部署机器学习模型的通用型框架。

TensorFlow was initially conceived as a convenient programming platform for neural network-based or deep learning artificial intelligence research. Although at first TensorFlow resembled more of a neural network computing tool based on a library, Google positioned it as a general high-performance numerical computation library. However, in practice, as it began to support more key features, including Keras, it gradually evolved into a universal framework adaptable for building and deploying machine learning models for both beginners and experts.

# Installation

- TensorFlow 原生支持 Linux 平台，依赖于 Python，并且与 Python 发行版本相关；在 Windows 平台的安装依赖于社区的工作，且对 GPU 的支持有些滞后。下面介绍基于 Anaconda 环境的安装。

- 对于基于 Intel CPU 的电脑，由于 Intel 针对该公司的 CPU 做过 Python 及相关库的优化，因此推荐使用 Intel 的 Python 发行版本：

- 网址为：

<https://www.intel.com/content/www/us/en/developer/tools/oneapi/distribution-for-python.html>

Intel® Distribution for Python\*

Achieve near-native code performance with this set of essential packages optimized for high-performance numerical and scientific computing.

What's Included Documentation Training Specifications Help

### High-Performance Python

The Intel® Distribution for Python\* provides:

- Near-native performance through acceleration of core numerical and machine learning packages with libraries like the [Intel® oneAPI Math Kernel Library](#) and [Intel® oneAPI Data Analytics Library](#)
- Support for the latest CPU instructions to accelerate workloads
- Performance using all available CPU cores on laptops, desktops, and powerful servers
- Productivity tools for compiling Python code into optimized instructions
- Essential Python bindings for easing integration of Intel native tools with your Python project

### Who Needs This Product

#### Beginners and Students

- Learn how to productively program in Python\* for the most performance.

#### Researchers and Scientific Computing Developers

- Accelerate and scale compute-intensive Python code with optimized

#### Download as Part of the Toolkit

Intel® Distribution for Python\* is included as part of the [Intel® AI Analytics Toolkit](#), which provides accelerated machine learning and data analytics pipelines with optimized deep-learning frameworks and high-performing Python libraries.

Get It Now

#### Download the Stand-Alone Version

A stand-alone download of Intel Distribution for Python is available. You can download binaries from Intel or choose your preferred repository.

Download

#### Download from Package Managers or Repositories

Installation Guides: [Conda\\*](#) | [YUM](#) | [APT](#)

# Installation

- ▶ TensorFlow is only natively supported on the Linux variant platforms. TensorFlow depends on the Python and its release version due to the compilation process. Installation on Windows platform is supported by the community however advanced features such as GPU are of limited support. The following instructions are based on the Anaconda toolset.
- ▶ Intel has optimized Python to exert potentials of their CPUs by exploiting some exclusive features. So if your computer is installed with an Intel CPU, use the Python release from Intel. Just install the standalone version from the below website is enough. You can search something like “Intel python installation standalone” to find the source:

<https://www.intel.com/content/www/us/en/developer/tools/oneapi/distribution-for-python.html>

- ▶ For AMD CPU, one can install the software directly from the official website of Anaconda:

<https://www.anaconda.com/>

# Installation

- 我们以Intel的Python发行版本为例进行讲解，首先下载并安装Python：

We exemplify the process by installing the Python shipped by Intel, first download and install it from Intel's website:

<https://www.intel.com/content/www/us/en/developer/articles/tool/oneapi-standalone-components.html#python>

## 2023.0.0 Release

Name (Click to initiate download)	Version	Size	Installer	Date
<a href="#">Intel Distribution for Python for Linux*</a>	2023.0.0	18 MB	Online	Dec. 09, 2022
<a href="#">Intel Distribution for Python for Linux</a>	2023.0.0	1.2 GB	Offline	Dec. 09, 2022
<a href="#">Intel Distribution for Python for Windows*</a>	2023.0.0	12 MB	Online	Dec. 14, 2022
<a href="#">Intel Distribution for Python for Windows</a>	2023.0.0	1.1 GB	Offline	Dec. 14, 2022
<a href="#">Intel Distribution for Python for macOS*</a>	2023.0.0	25 MB	Online	Dec. 08, 2022
<a href="#">Intel Distribution for Python for macOS</a>	2023.0.0	482 MB	Offline	Dec. 08, 2022

# Installation

- 如果用户的电脑性能有限，也可以直接选择从官网下载安装Python的Windows发行版本，通常推荐采用安装程序的推荐设置，例如将Python相关程序加入系统路径：

If the computers of users are of limited performance, an alternative is to download the Windows release from the official website. It is recommended to adopt the settings recommended by the installer, for example, add the executable into the system path.

- 另外一种选择是安装Miniconda，其仅具有基本的Python库与conda包管理程序。当用户用到具体库时，再进行安装。

Another option is to install Miniconda, which only includes the basic Python packages and conda package manager. When the user needs a specific package or library, then it can be installed up to the requirement.

Python官网

Python official website

<https://www.python.org/downloads/>

Miniconda官网

Miniconda official website

<https://docs.anaconda.com/miniconda/>

# Installation

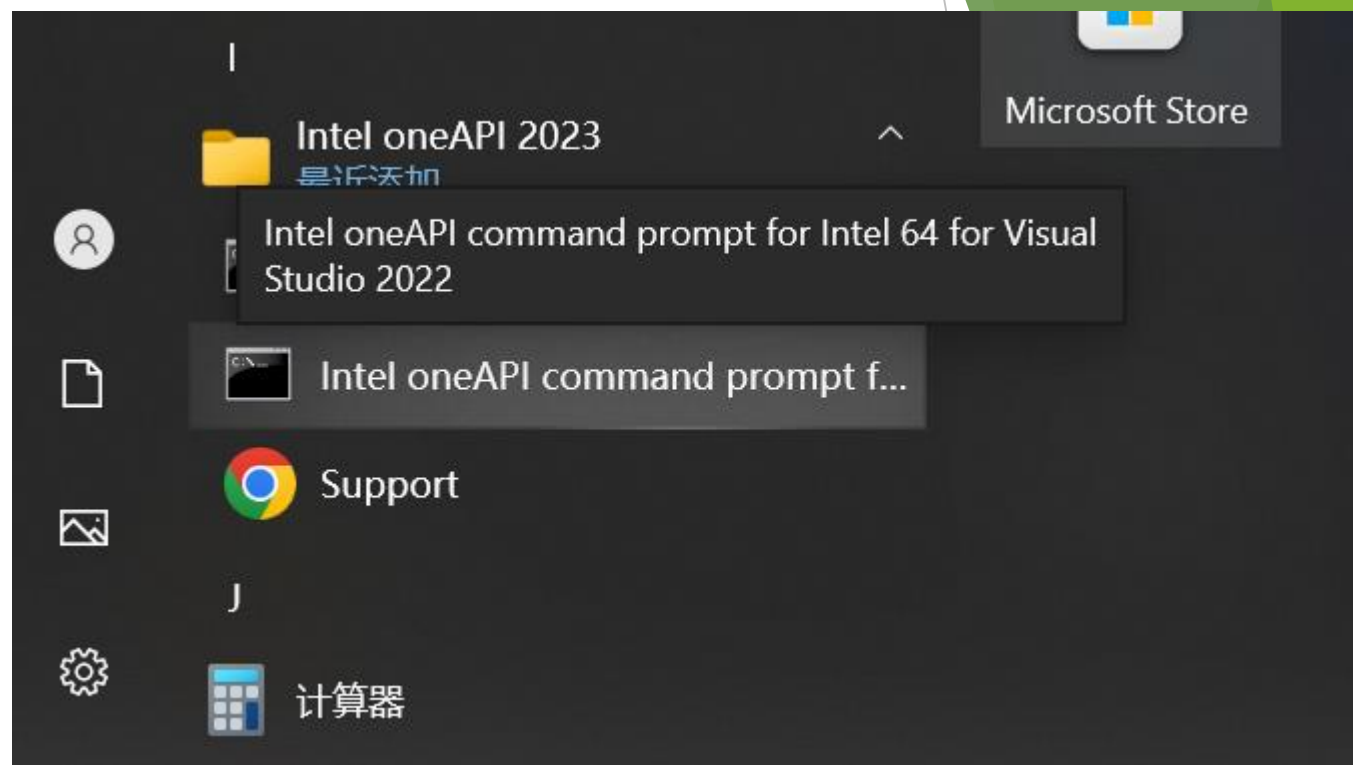
- ▶ 启动安装后的Intel Python环境，一般启动64位的。查看Python版本：

Launch the Intel Python environment, usually the 64-bit command prompt. Check the Python version:

```
python -V
```

- ▶ 可以看出作者所安装的Python的版本为3.9。

It is manifest that the python version of the author's installation is 3.9.



```
E:\Program Files (x86)\Intel\oneAPI>python -V  
Python 3.9.15 :: Intel Corporation
```



# Installation

- 无论我们安装的是Conda版本的Python，还是原生的Python，我们安装的一般是基础的使用环境。虽然我们可以在基础环境中继续安装，但更为合适的方式，是创建一个虚拟环境，在虚拟环境里安装我们所需要的包。当新安装的包有问题时，不需要更改原来的基础环境，只需要将有问题的虚拟环境删除，重新创建即可。

No matter we install the Conda version of Python or the native Python, what we installed is a basic environment. Although we can continue to install packages in the base environment, it is more appropriate to create a virtual environment and install the necessary packages within it. When the newly installed packages have issues, we do not need to alter the original base environment; we just need to delete the problematic virtual environment and recreate it.

- 我们首先以Conda版本为例，讲解如何创建虚拟环境并安装TensorFlow。首先更新Conda包管理器，然后通过如下命令创建名为TensorFlow的虚拟环境：

First, we will explain how to create a virtual environment and install TensorFlow using a Conda version Python as an example. First, update the conda before creating the virtual environment, then create a new environment named tensorflow by executing the following commands:

```
conda update conda
```

```
conda create -n tensorflow --clone base
```

# Installation

- ▶ 在创建虚拟环境之后，通过如下命令激活虚拟环境：

After create the virtual environment, activate conda environment via the following command:

```
conda activate tensorflow
```

- ▶ 如果用户不打算利用显卡训练网络，且在安装Intel的Python发行版的情况下，则可以考虑通过指定渠道安装Intel版本的TensorFlow发行版：

If the user has no intention to harness the GPU for training models, especially the installed Python is released by Intel, then it is recommended to install the TensorFlow released by Intel via specifying the channel:

```
conda install -c intel tensorflow
```

- ▶ 注意，如果我们使用的是Intel的CPU，但不是安装的Intel的Python发行版，例如通过Anaconda官网上安装，则推荐将Intel加入为源，然后再创建名为tensorflow的虚拟环境。



# Installation

If you are using Intel's CPU, but not installed the Python shipped by Intel, for example, you did it via downloading from Anaconda's official website, then it is recommended to add Intel as the channel first, following by creating a virtual environment named tensorflow:

```
conda config --add channels intel
```

```
conda create -n tensorflow intelpython3_core
```

- 注意，使用参数intelpython3\_core表明仅安装Intel维护的上游当前Python版本的核心功能，如想安装全部功能，则可以用参数intelpython3\_full。同时，可以用显示指定版本，例如安装3.7的全部功能：

Notably, the parameter intelpython3\_core indicates to install the core function of the up-stream python libraries maintained by Intel. To install the full packages, use the parameter intelpython3\_full. You can simultaneously specify a particular Python version, such as 3.7:

```
conda create -n tensorflow intelpython3_full python=3.7
```

# Installation


- 如果用户期望利用电脑的显卡支持模型的训练，则可以考虑安装GPU版本的TensorFlow。不过需要注意的是，原生Windows上的GPU支持仅适用于2.10或更早的版本，从TensorFlow 2.11开始，不再支持Windows上的CUDA构建。尽管可以基于WSL2利用TensorFlow-DirectML-Plugin使用TensorFlow GPU，但推荐读者安装TensorFlow 2.10版本并配置相应的CUDA环境。

If the user intends to utilize the GPU card to train neural network models, then TensorFlow GPU is a good candidate. Notable, GPU support on native-Windows is only available for 2.10 or earlier versions, starting in TF 2.11, CUDA build is not supported for Windows. By installing TensorFlow in WSL2 coupled by TensorFlow-DirectML-Plugin, the user can use TensorFlow GPU on Windows, however, the sophistication incurs the better choice to install TensorFlow 2.10 to simplify the overall process including CUDA environment configuration.

# Installation

- ▶ 由于TensorFlow 2.10支持的CUDA版本是11.2，cuDNN版本是8.1，因此我们去英伟达的网站下载对应的版本安装。注意，由于较旧的CUDA版本可能会导致显卡驱动回退，因此，如果用户对此存疑，则可使用仅CPU版本。

TensorFlow 2.10 only supports CUDA 11.2 and cuDNN 8.1., therefore, we need to download the corresponding libraries and install them. Notice that earlier version of CUDA might leads to the rollback of the latest GPU driver to old versions, has the user had any concern in this regard, just use the CPU version of TensorFlow.

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

[CUDA Toolkit 11.4.0 \(June 2021\), Versioned Online Documentation](#)  
[CUDA Toolkit 11.3.1 \(May 2021\), Versioned Online Documentation](#)  
[CUDA Toolkit 11.3.0 \(April 2021\), Versioned Online Documentation](#)  
[CUDA Toolkit 11.2.2 \(March 2021\), Versioned Online Documentation](#)  
[CUDA Toolkit 11.2.1 \(February 2021\), Versioned Online Documentation](#)  
[CUDA Toolkit 11.2.0 \(December 2020\), Versioned Online Documentation](#)

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

**Download cuDNN v8.2.0 (April 23rd, 2021), for CUDA 10.2**

**Download cuDNN v8.1.1 (Feburary 26th, 2021), for CUDA 11.0,11.1 and 11.2**

**Download cuDNN v8.1.1 (Feburary 26th, 2021), for CUDA 10.2**

# Installation

- ▶ 对于CUDA，英伟达提供了安装程序帮助用户进行安装，但对于cuDNN，则需要解压之后，再整体复制到CUDA安装目录。当用户安装所有的英伟达与CUDA相关的依赖库之后，则可以继续安装TensorFlow的GPU版本。注意，由于并没有官方构建的对应于conda包管理程序的版本，因此，只能通过pip的方式进行安装：

For CUDA, NVIDIA provides an installer to help users with the installation, but for cuDNN, it needs to be extracted and then copied entirely to the CUDA installation directory. After the user installs all the NVIDIA's CUDA-related dependencies, they can proceed to install the GPU version of TensorFlow. Note that, since there is no officially built version corresponding to the conda package manager, installation can only be done through pip:

```
pip install tensorflow-gpu==2.10.1
```

- ▶ 接下来，我们需要配置相关环境变量，以便可以使用CUDA库与cuDNN库。

In the following, we need to configure the environment variable in order to use the CUDA and cuDNN libraries.

# Installation

- ▶ 对于CUDA，英伟达提供了安装程序帮助用户进行安装，但对于cuDNN，则需要解压之后，再整体复制到CUDA安装目录。当用户安装所有的英伟达与CUDA相关的依赖库之后，则可以继续安装TensorFlow的GPU版本。注意，由于并没有官方构建的对应于conda包管理程序的版本，因此，只能通过pip的方式进行安装：

For CUDA, NVIDIA provides an installer to help users with the installation, but for cuDNN, it needs to be extracted and then copied entirely to the CUDA installation directory. After the user installs all the NVIDIA's CUDA-related dependencies, they can proceed to install the GPU version of TensorFlow. Note that, since there is no officially built version corresponding to the conda package manager, installation can only be done through pip:

```
pip install tensorflow-gpu==2.10.1
```






- ▶ 接下来，我们需要配置相关环境变量，以便可以使用CUDA库与cuDNN库。

In the following, we need to configure the environment variable in order to use the CUDA and cuDNN libraries.

# Installation

- 注意：（1）尽管CUDA可以匿名下载，但cuDNN的下载需要注册；（2）由于安装CUDA时，需要安装对应的驱动程序。如果显卡的驱动程序比CUDA11.4.4对应的驱动程序要新，则有可能发生安装失败的情况。因此，需要在安装之前将显卡驱动首先卸载掉：

Note, (1) Although CUDA could be downloaded in an anonymously way, nVidia requires registration for downloading cuDNN. (2) since installing CUDA will simultaneously install the underlying driver, so a installed driver with newer version might incur installation failure. Therefore, it is recommended to uninstall the GPU driver before launch the installation.

组织 ▾	
名称	发布者
 NVIDIA PhysX 系统软件 9.21.0713	NVIDIA Corporation
 NVIDIA Tools Extension SDK (NVTX) - 64 bit	NVIDIA Corporation
 NVIDIA 图形驱动程序 472.50	NVIDIA Corporation
 OBS Studio	OBS Project
 OpenShot Video Editor 3.0.0	OpenShot Studios, LLC



# Installation

- ▶ 对于cuDNN，若用户仅将解压后的内容拷贝到对应的CUDA文件夹中，则只需配置如下两个环境变量，否则，则需将cudnn解压后的bin文件夹添加到系统路径中：

If the user already copied the unzipped content of cuDNN to the corresponding folder of CUDA, then only two options need to be configured, otherwise the bin folder of cudnn needs to be configured as well:

> cudnn-11.4-windows-x64-v8.2.4.15 > cuda

名称

- bin
- include
- lib

(C:) > Program Files > NVIDIA GPU Computing Toolkit > CUDA > v11.4

名称

修改日期

bin	2023/2/3 14:31
compute-sanitizer	2023/2/3 14:28
extras	2023/2/3 14:28
include	2023/2/3 14:31
lib	2023/2/3 14:28
libnvvp	2023/2/3 14:28
nvml	2023/2/3 14:28

OS

Windows\_NT

Path

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.2\...

编辑环境变量

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.2\bin  
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.2\extras\CUPTI\lib64  
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.2\libnvvp

# Installation

- ▶ 若CUDA配置正确，则执行nvidia-smi命令时，如显示如右图所示的信息：

The user can execute the nvidia-smi and by examining the output to check whether CUDA is correctly configured or not:

- ▶ 执行下面命令，若无报错，则说明TensorFlow安装成功：

Execute the following command, no error output indicates the successful installation of tensorflow:

```
python -c "import tensorflow  
as tf"
```

```
C:\Users\Lenovo>nvidia-smi
Sat Feb  4 23:05:31 2023
```

NVIDIA-SMI 472.50				Driver Version: 472.50		CUDA Version: 11.4		
GPU	Name	TCC/WDDM	Bus-Id	Disp.A	Memory-Usage	Volatile GPU-Util	Uncorr. Compute M.	ECC MIG M.
Fan	Temp	Perf Pwr:Usage/Cap						
0	NVIDIA GeForce ...	WDDM	00000000:01:00.0	Off				N/A
N/A	58C	P8 N/A / N/A	119MiB / 2048MiB			0%	Default	N/A

```
Processes:
GPU  GI  CI  PID  Type  Process name  GPU Memory Usage
   ID  ID
=====
No running processes found
```