

# Reactor

PyTorch - 基礎學習

---

2021-12-03



# Map



# 我係



## Kinfey Lo – (盧建暉)

Microsoft Cloud Advocate

前微軟MVP、Xamarin MVP和微軟RD，擁有超過10年的雲原生、人工智能和流動應用程式開發經驗，為教育、金融和醫療提供應用。在微軟，為技術人員和不同行業宣講技術和相關應用場景。



Love Coding(Python , C# , TypeScript , Swift , Rust , Go )

專注於人工智能，雲原生，流動平台移動開發

**Github :** <https://github.com/kinfey>

**Email :** [kinfeylo@microsoft.com](mailto:kinfeylo@microsoft.com)   **Blog :** <https://dev.to/kinfey>

**Twitter :** @Ljh8304

# PyTorch 介紹



# PyTorch

<https://pytorch.org/>

開源機器學習框架，可加快從研究原型設計到生產部署的過程。

## 開發場景

使用 TorchScript 在 Eager 和圖形模式之間無縫轉換，並使用 TorchServe 加快反覆運算。

## 分散式

研究和生產中的可擴展分散式訓練和性能優化由 `torch.distributed` 後端實現。

## 應用場景和庫都齊備

豐富的工具和庫生態系統擴展了 PyTorch 並支援電腦視覺、NLP 等領域的開發。

## 雲端支持

PyTorch 在主要雲平臺上得到很好的支援，提供無縫開發和非常好的擴展性能。



# PyTorch 安裝

<https://pytorch.org/>

PyTorch Build	Stable (1.10)	Preview (Nightly)	LTS (1.8.2)
Your OS	Linux	Mac	Windows
Package	Conda	Pip	LibTorch Source
Language	Python	C++ / Java	
Compute Platform	CUDA 10.2	CUDA 11.3	ROCM 4.2 (beta) CPU
Run this Command:	<pre>pip3 install torch==1.10.0+cpu torchvision==0.11.1+cpu torchaudio==0.10.0+cpu -f https://download.pytorch.org/wheel/cpu/torch_stable.html</pre>		

# 基於機器學習環境的配置

## 輕鬆搭建機器學習/深度學習開發環境

<https://blog.csdn.net/kinfey/article/details/117635067>

## 從開發者角度玩Windows 11

<https://blog.csdn.net/kinfey/article/details/120614677>



# PyTorch 基礎介紹



# Tensor 張量介紹

張量是一種特殊的資料結構，與陣列和矩陣非常相似。在 PyTorch 中，我們使用張量對模型的輸入和輸出以及模型的參數進行編碼。

張量類似於 NumPy 的 ndarray，不同之處在於張量可以在 GPU 或其他硬體加速器上運行。事實上，張量和 NumPy 陣列通常可以共用相同的底層記憶體，從而無需複製資料。張量也針對自動微分進行了優化。如果您熟悉 ndarrays，那麼您將熟悉 Tensor API。



# Tensor 張量介紹

```
data = [[1, 2],[3, 4]]  
x_data = torch.tensor(data)  
print(x_data)
```

## 張量運算

有超過100種張量相關的運算操作, 例如轉置、索引、切片、數學運算、線性代數、隨機採樣等。更多的運算可以在這裡查看。

所有這些運算都可以在GPU上運行(相對於CPU來說可以達到更高的運算速度)。



# Sample



# Datasets 和 Dataloaders



# Datasets 和 Dataloaders

處理資料樣本的代碼可能會變得混亂且難以維護；我們理想地想要我們的資料集代碼與我們的模型訓練代碼分離，以獲得更好的可讀性和模組化。

PyTorch 提供了兩種資料原語：`torch.utils.data.DataLoader` 和 `torch.utils.data.Dataset`

允許您使用預載入的資料集以及您自己的資料。

`Dataset` 存儲樣本及其相應的標籤，`DataLoader` 包裝了一個可反覆運算物件  
`Dataset` 可以輕鬆訪問樣本。

PyTorch 域庫提供了許多預載入的資料集（例如 FashionMNIST），這些資料集子類 `torch.utils.data.Dataset` 並實現特定於特定資料的功能。

它們可用於對您的模型進行原型設計和基準測試。你可以找到他們此處：

圖像資料集 <https://pytorch.org/vision/stable/datasets.html>

文本資料集 <https://pytorch.org/text/stable/datasets.html>

音訊資料集 <https://pytorch.org/audio/stable/datasets.html>



# 示例



# Transforms



# Transforms 數值變換

資料並不總是以其所需的最終處理形式出現訓練機器學習演算法。 我們使用 **transforms\***來執行一些處理數據並使其適合訓練。

所有 TorchVision 資料集都有兩個參數（`transform` 以修改特徵和`target_transform` 修改標籤）接受包含轉換邏輯的可調用物件。<https://pytorch.org/vision/stable/transforms.html> 模組提供幾種常用的開箱即用轉換。



# 示例



# 神經網絡創建



# 構建神經網絡

神經網絡由對資料執行操作的層/模組組成。`torch.nn`命名空間提供了您需要的所有構建塊建立自己的神經網絡。

PyTorch 中的每個模組都是 `nn.Module` 的子類。

神經網絡是一個模組，由其他模組（層）組成。

這種嵌套結構允許輕鬆構建和管理複雜的架構。



# 示例



# autograd



# 使用“torch.autograd”自動微分

在訓練神經網絡時，最常用的演算法是反向傳播。

在這個演算法中，參數（模型權重）是根據損失函數的梯度進行調整到給定的參數。

為了計算這些梯度，PyTorch 有一個內置的微分引擎稱為“`torch.autograd`”。它支持任何梯度的自動計算計算圖。

考慮最簡單的一層神經網路，輸入為“`x`”，參數 `w` 和 `b`，以及一些損失函數。



# 示例



# 優化模型參數



# 優化模型參數

現在我們有了模型和資料，是時候通過優化資料上的參數來訓練、驗證和測試我們的模型了。訓練模型是一個反覆運算過程；在每次反覆運算中 *epoch*，模型對輸出進行猜測，計算其猜測中的誤差 *loss*，收集誤差相對於其參數的導數（如我們在模組中看到的），並使用梯度下降**優化**這些參數。



# Sample



# 模型應用



# 示例



# Sample



# Sample



# Microsoft Learn上的人工智慧

<https://docs.microsoft.com/zh-hk/learn>



**PyTorch 基礎知識**

<https://aka.ms/PytorchStudy>



**TensorFlow 基礎知識**

<https://aka.ms/TensorflowLearn>

# Q&A





# Reactor

## Thank You!