驚呆了！ 這樣可以將Numpy加速700倍！

小白學視覺 2021-12-22 10:05

點擊上方“**小白學視覺**”，選擇加"**星標**"或“**置頂**”
重磅乾貨，第一時間送達

選自| towardsdatascience 作者| George Seif
來自| 機器之心

本文介紹瞭如何利用CuPy 庫來加速Numpy 運算速度。

就其自身來說，Numpy 的速度已經較Python 有了很大的提升。當你發現Python 代碼運行較慢，尤其出現大量的for-loops 循環時，通常可以將數據處理移入Numpy 並實現其向量化最高速度處理。

但有一點，上述Numpy 加速只是在CPU 上實現的。由於消費級CPU 通常只有8 個核心或更少，所以並行處理數量以及可以實現的加速是有限的。

這就催生了新的加速工具——CuPy 庫。

何為CuPy?

CuPy 是一個借助CUDA GPU 庫在英偉達GPU 上實現Numpy 數組的庫。基於Numpy 數組的實現，GPU 自身俱有的多個CUDA 核心可以促成更好的並行加速。

CuPy 接口是Numpy 的一個鏡像，並且在大多情況下，它可以直接替換Numpy 使用。只要用兼容的CuPy 代碼替換Numpy 代碼，用戶就可以實現GPU 加速。

CuPy 支持Numpy 的大多數數組運算，包括索引、廣播、數組數學以及各種矩陣變換。

如果遇到一些不支持的特殊情況，用戶也可以編寫自定義Python 代碼，這些代碼會利用到CUDA 和GPU 加速。整個過程只需要C++格式的一小段代碼，然後CuPy 就可以自動進行GPU 轉換，這與使用Cython 非常相似。

在開始使用CuPy 之前，用戶可以通過pip 安裝CuPy 庫：

```
pip install cupy
```

使用CuPy 在GPU 上運行

為符合相應基準測試，PC 配置如下：

- i7–8700k CPU
- 1080 Ti GPU
- 32 GB of DDR4 3000MHz RAM
- CUDA 9.0

CuPy 安裝之後，用戶可以像導入Numpy 一樣導入CuPy：

```
import numpy as np
import cupy as cp
import time
```

在接下來的編碼中，Numpy 和CuPy 之間的切換就像用CuPy 的cp 替換Numpy 的np 一樣簡單。如下代碼為Numpy 和CuPy 創建了一個具有10 億1』s 的3D 數組。為了測量創建數組的速度，用戶可以使用Python 的原生time 庫：

```
### Numpy and CPU
s = time.time()
*x_cpu = np.ones((1000,1000,1000))*
e = time.time()
```

```
print(e - s)### CuPy and GPU
s = time.time()
*x_gpu = cp.ones((1000,1000,1000))*
e = time.time()
print(e - s)
```

這很簡單！

令人難以置信的是，即使以上只是創建了一個數組，CuPy 的速度依然快得多。Numpy 創建一個具有10 億1』s 的數組用了1.68 秒，而CuPy 僅用了0.16 秒，實現了10.5 倍的加速。

但CuPy 能做到的還不止於此。

比如在數組中做一些數學運算。這次將整個數組乘以5，並再次檢查Numpy 和CuPy 的速度。

```
### Numpy and CPU
s = time.time()
*x_cpu *= 5*
e = time.time()
print(e - s)### CuPy and GPU
s = time.time()
*x_gpu *= 5*
e = time.time()
print(e - s)
```

果不其然，CuPy 再次勝過Numpy。Numpy 用了0.507 秒，而CuPy 僅用了0.000710 秒，速度整整提升了714.1 倍。

現在嘗試使用更多數組並執行以下三種運算：

1. 數組乘以 5
2. 數組本身相乘
3. 數組添加到其自身

```
### Numpy and CPU
s = time.time()
*x_cpu *= 5
x_cpu *= x_cpu
x_cpu += x_cpu*
e = time.time()
print(e - s)### CuPy and GPU
s = time.time()
*x_gpu *= 5
x_gpu *= x_gpu
```

```
x_gpu += x_gpu*
e = time.time()
print(e - s)
```

結果顯示，Numpy 在CPU 上執行整個運算過程用了1.49 秒，而CuPy 在GPU 上僅用了0.0922 秒，速度提升了16.16 倍。

數組大小（數據點）達到1000 萬，運算速度大幅度提升

使用CuPy 能夠在GPU 上實現Numpy 和矩陣運算的多倍加速。值得注意的是，用戶所能實現的加速高度依賴於自身正在處理的數組大小。下表顯示了不同數組大小（數據點）的加速差異：

Search this file...

Operation	Array Size	Speedup
Create	1000000	0.011
Multiply by 5	1000000	0.95X
Create	10000000	0.1
Multiply by 5	10000000	9.5
Create	100000000	1.1
Multiply by 5	100000000	739.1
Create	1000000000	10.5
Multiply by 5	1000000000	714.1

cupy_results.csv hosted with ♥ by GitHub view raw

數據點一旦達到1000 萬，速度將會猛然提升；超過1 億，速度提升極為明顯。Numpy 在數據點低於1000 萬時實際運行更快。此外，GPU 內存越大，處理的數據也就更多。所以用戶應當注意，GPU 內存是否足以應對CuPy 所需要處理的數據。

原文鏈接：https://towardsdatascience.com/heres-how-to-use-cupy-to-make-numpy-700x-faster-4b920dda1f56

下載1：OpenCV-Contrib擴展模塊中文版教程

在「**小白學視覺**」公眾號後台回复：**擴展模塊中文教程，**即可下載全網第一份OpenCV擴展模塊教程中文版，涵蓋**擴展模塊安裝、SFM算法、立體視覺、目標跟踪、生物視覺、超分辨率處理**等二十多章內容。

下載2：Python+OpenCV視覺實戰項目52講

在「**小白學視覺**」公眾號後台回复：**Python視覺實戰項目，**即可下載包括**圖像分割、口罩檢測、車道線檢測、車輛計數、添加眼線、車牌識別、字符識別、情緒檢測、文本內容提取、面部識別**等31

個視覺實戰項目，助力快速學校計算機視覺。

下載3：Pytorch常用函數手冊

在「**小白學視覺**」公眾號後台回复：**pytorch常用函數手冊，**即可下載含有**200餘個**Pytorch常用函數的使用方式，幫助快速入門深度學習。

交流群

歡迎加入公眾號讀者群一起和同行交流，目前有**SLAM 、三維視覺、傳感器、自動駕駛、計算攝影**、檢測、分割、識別、**醫學影像、GAN 、算法競賽**等微信群（以後會逐漸細分），請掃描下面微信號加群，備註：”暱稱+學校/公司+研究方向“，例如：”張三 + 上海交大 + 視覺SLAM“。**請按照格式備註，否則不予通過。**添加成功後會根據研究方向邀請進入相關微信群。**請勿**在群內發送**廣告**，否則會請出群，謝謝理解~

喜歡此內容的人還喜歡

浙大博士整理的計算機視覺學習路線（含時間建議分配）

小白學視覺

一路逆襲！他讀博後3年發2篇Nature，創造校史……

青年參考

【解盤】繼續防守

知著