


[지식 확장 정보](#) | [컴퓨터 정보](#) | [코딩 정보](#) | [음악 정보](#) | [책 정보](#) | [BLOGSPOT 꾸미기 정보](#)


Breaking

PYTHON

[pydot install 문제 해결법](#)

PYTHON

[folium가 jupyter notebook에 display가 되지 않을 때 해결법](#)

PYT



POPULAR

RECENT

C

[tensorflow gpu inst](#)
[NVIDIA CUDA 설치하](#)
[텐서플로 설치 \(First S](#)
[Tensorflow : install](#)

Scripts activate.bat
 또는 외부 명령, 실행할
 그램, 또는 배치 파일이
 (Scripts activate.ba
 recognized as an internal or ex
 command, operable program o
 file.) solution

[Shrimp Vs Prawn -](#)
[데? <별별정보>](#)
[plaster - 어디가 아파](#)
[보>](#)
[Home](#) * [python](#) * [tensorflow](#) * [tensorflow gpu install - NVIDIA CUDA 설치하기](#)

tensorflow gpu install - NVIDIA CUDA 설치하기

🧑🏻 남남편 12월 29, 2018 📁 ,python ,tensorflow

tensorflow gpu install - NVIDIA CUDA 설치하기 - 별별정보

DeepLearning을 하게되면서 CPU만으로 코드를 돌리는 것에 한계를느낄 때 쯤 GPU에 눈이 돌아가게 되었다. 특히나 이미
 지를 다루는 CNN에서 크게 영향을 받기 시작할 것이다.

DeepLearning의 초시라고 여기는 ILSVRC ImageNet 2012년 AlexNet의 경우에도 비록 그 당시의 GPU가 현재에 비할바
 는 아니지만 NVIDIA GTX 580을 사용하여 훈련 이미지를 5~6일에 거쳐 이미지를 학습시켰다고한다. 현재는 두말할 나위 없이
 GPU의 사용이 필수적으로 여겨지고 있다.

그렇다면 GPU의 사용이 AMD나 NVIDIA에 국한된 것이냐고 묻는다면 꼭 그런 것은 아니다.

컴퓨터를 잘 모르던 시기에 완제품을 산 사람들의 경우 내장 그래픽카드를 소유하고 있는 경우가 많은데 우리나라의 경우는 특
 히 Intel 사의 HD Graphics를 많이 장착한 경우가 많다.

그러나 이들의 사용이 완전 불가능한 것은 아니나 우리가 현재 다룰 tensorflow에 있어서는 다른 회사의 GPU는 전혀 사용하
 지 못하고 NVIDIA사의 GTX만이 사용가능하다.

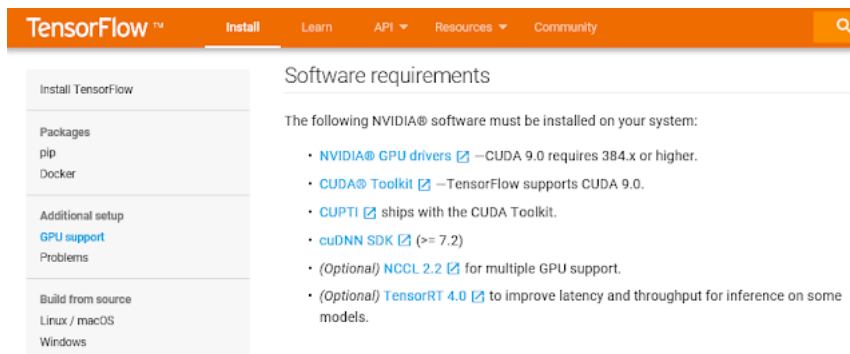
따라서 tensorflow gpu를 사용하기 위해서는 반드시 자신의 컴퓨터에 NVIDIA GTX가 장착되어 있는지 확인해야할 것이다.
 다른 회사의 내장 그래픽카드 혹은 AMD 관련 GPU 사용은 OpenCV 관련 계열이나 의학 사진과 관련해서 자체적으로 제공하
 는 툴킷이 있는데 이것은 관련 홈페이지를 찾아 참고하길 바란다.

2018년 12월 29일 글을 적는 현재 Tensorflow가 지원하는 tensorflow-gpu의 필요 조건을 살펴보면 다음과 같다.

전체 페이지뷰

8,092





여기서 여러분이 해야하는 일은 **CUDA Toolkit 9.0**과 **cuDNN SDK 7.2 Version**이상을 설치하는 것이다.

참고로 버전 정보와 관련해서는 지금 작성하는 2018년 12월 29일에서 많은 시간 차이가 난다면 꼭 Tensorflow 홈페이지에 방문해서 Software requirements를 확인해야한다.

<https://www.tensorflow.org/install/gpu>

이제 우리는 조건에서 제시한 소프트웨어를 설치해보자.

검색엔진에 cuda toolkit 9.0을 치거나 아래 주소 링크에 들어가서 Base Installer를 설치한다.

<https://developer.nvidia.com/cuda-90-download-archive>

설치 전에 필요 용량이 대략 4G 이상 필요하니 여유 공간을 미리 만들어둬야한다.

CUDA Toolkit 9.0 Downloads

Select Target Platform ⓘ

Click on the green buttons that describe your target platform. Only supported platforms will be shown.

Operating System	Windows	Linux	Mac OSX		
Architecture ⓘ	x86_64				
Version	10	8.1	7	Server 2016	Server 2012 R2
Installer Type ⓘ	exe (network)	exe (local)			

Download Installers for Windows 10 x86_64

The base installer is available for download below.
There are 4 patches available. These patches require the base installer to be installed first.

Base Installer	Download (14.5 MB)
----------------	--------------------

Installation Instructions:

- Double click cuda_9.0.176_win10_network.exe
- Follow on-screen prompts

관련 작업을 모두 끝냈다면 cuDNN SDK를 설치해야하는데 버전이 7.2 이상을 설치하라고 했으니 이에 맞게 작업하면 된다. 다만 여기서는 로그인 필수적으로 필요하니 NVIDIA의 Join을 눌러 가입해야한다. 개인적인 생각으로는 Tensorflow GPU 동반자가 NVIDIA이기에 가입해두는 것도 나쁘지 않은 선택 같다. 반 강제적이지만... ㅋ

<https://developer.nvidia.com/cudnn> -> Download cuDNN -> Login

-> cuDNN v (>=)7.2 for CUDA 9.0

저의 경우는 가장 최신 버전인 7.4.2를 다운받았습니다.

cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

☒ I Agree To the Terms of the [cuDNN Software License Agreement](#)

Note: Please refer to the [Installation Guide](#) for release prerequisites, including supported GPU architectures and compute capabilities, before downloading.

For more information, refer to the cuDNN Developer Guide, Installation Guide and Release Notes on the [Deep Learning SDK Documentation](#) web page.

[Download cuDNN v7.4.2 \[Dec 14, 2018\], for CUDA 10.0](#)

[Download cuDNN v7.4.2 \[Dec 14, 2018\], for CUDA 9.2](#)

[Download cuDNN v7.4.2 \[Dec 14, 2018\], for CUDA 9.0](#)

Library for Windows, Mac, Linux, Ubuntu and RedHat/Centos (x86_64 architecture)

[cuDNN Library for Windows 7](#)

[cuDNN Library for Windows 10](#)

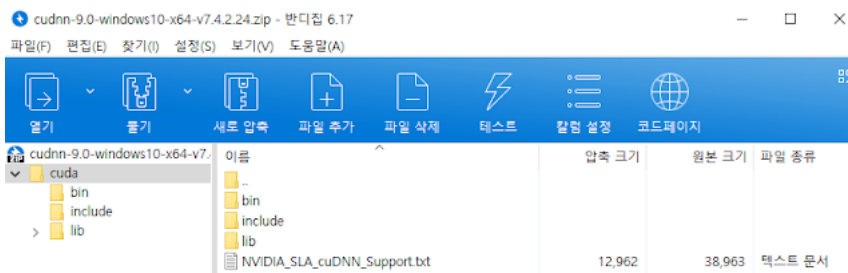
[cuDNN Library for Linux](#)

[cuDNN Runtime Library for Ubuntu16.04 \[Deb\]](#)

[cuDNN Developer Library for Ubuntu16.04 \[Deb\]](#)

여기까지 다운받고나면 cuDNN을 설치할 때부터가 중요해집니다. 그 이유는 경로 설정 때문인데 **2가지 방법이 존재**합니다.

첫 번째는 **CUDA Toolkit 9.0**을 설치한 곳에 **cuDNN**을 강제적으로 풀어 **bin, include, lib**를 덮어쓰기를 하는 방식을 권한 경우도 있습니다만 저는 그렇게 하지 않는 것을 권유하고 싶습니다. 그 이유는 제가 그렇게 했다 실패했던 경험 때문입니다. 물론 덮어쓰기를 통해 바로 실행이 되는 경우도 있습니다만 저는 오히려 기존의 CUDA Toolkit에 존재하는 bin, Include, lib가 필요한 경우도 컴퓨터에서 필요했던 것이 아닐까하는 생각을 하게되었기 때문에 그렇게 설치하는 방법을 가르쳐드리기보다 따로 폴더 한 곳에 다운받은 **cuDNN**을 풀어놓기를 권유하고 싶습니다. 이것이 두 번째 방법입니다.



참고로 덮어쓰는 방식을 알려드리자면 기존의 CUDA Toolkit 9.0를 설치한 경로에 다운받은 cudnn-9.0-windows10-x64-v7.4.2.24를 풀어 넣으시기만 하면됩니다.

CUDA Toolkit 9.0를 설치한 경로 Default

=>C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0

그러나 저는 따로 cudnn-9.0-windows10-x64-v7.4.2.24를 바탕화면에 cuda라는 폴더이름으로 풀어놓았습니다.

여기서부터 중요합니다.

제어판 - 시스템 - 고급 시스템 설정 - 환경 변수 - (자신의 컴퓨터 이름)에 대한 사용자 변수 Path 클릭 - 편집 클릭하면 **환경 변수 편집**이라는 창이 뜰 것 입니다.

그곳에 자신이 풀어놓은 **cuDNN의 bin, include, lib**를 새로 만들기를 통해 경로에 추가합니다.

예를 들어드리면 저는 바탕화면에 풀어놨기 때문에

bin,include, lib 파일의 경로가 다음과 같습니다.

C:\Users\{자신의 컴퓨터 이름}\Desktop\cuda\bin

C:\Users\{자신의 컴퓨터 이름}\Desktop\cuda\include

C:\Users\{자신의 컴퓨터 이름}\Desktop\cuda\lib

이것을 환경 변수 편집에서 새로 만들기 3번 반복을 통해 집어넣으면 끝입니다.

만약 제가 실패했던 첫 번째 방식을 하고 계시다면 default 경로를 다음과 같이 설정해야합니다.

C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\bin
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\include
C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.0\lib

이제 모두 확인 누르고 **Anaconda Prompt를 관리자 권한으로 실행**시켜봅시다.

여기서 가상 환경을 만들어 gpu가 사용가능한 것을 분리하겠습니다.
root에서까지 gpu를 사용하게 만들면 컴퓨터의 리소스가 지나친감이 있기 때문에 필요할 때 GPU를 사용하도록 만들기 위한
입니다.

가상환경 만들기 & 가상환경 활성화 & tensorflow gpu 설치

conda create -n tf_gpu python=3.5

activate tf_gpu

pip install --ignore-installed --upgrade tensorflow-gpu

이제 Anaconda Prompt에 python을 치고 import tensorflow를 쳐봅시다.

```
(tf_gpu) C:\Users\Wla>python
Python 3.5.6 [Anaconda, Inc.] (default, Aug 26 2018, 16:05:27) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright()", "credits" or "license()" for more.
>>> import tensorflow as tf
>>> a=tf.constant(3)
>>> sess=tf.Session()
2018-12-23 15:51:00.983148: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
2018-12-23 15:51:01.240284: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1482] Found device 0 with properties:
name: GeForce GTX 1050 Ti major: 6 minor: 1 memoryClockRate(GHz): 1.506
pciBusID: 0000:01:00:0
totalMemory: 4.00618 freeMemory: 3.28618
2018-12-23 15:51:01.243915: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1511] Adding visible gpu devices: 0
2018-12-23 15:51:02.006801: I tensorflow/core/common_runtime/gpu/gpu_device.cc:882] Device interconnect StreamExecutor with strength 1 edge matrix:
2018-12-23 15:51:02.008943: I tensorflow/core/common_runtime/gpu/gpu_device.cc:888] 0
2018-12-23 15:51:02.010365: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1001] 0: N
2018-12-23 15:51:02.012558: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1115] Created TensorFlow device (/job:local/localhost/replica:0/task:0/device:GPU:0 with 3007 MB memory) -> physical GPU (device: 0, name: GeForce GTX 1050 Ti, pci bus id: 0000:01:00:0, compute capability: 6.1)
>>>
```

session을 실행시키면 다음과 같이 제가 장착한 GTX 장치와 더불어 실행됨을 볼 수 있습니다.

PyCharm에서 실행시켜서 실제로 디버깅에 CPU와 GPU 사용량을 확인해볼까요?

작업 관리자		프로세스				
		상태	100% CPU	74% 메모리	0% 디스크	0% 네트워크
업 (8)						
>			0%	5.3MB	0MB/s	0Mbps
>			1.9%	493.2MB	0MB/s	0Mbps
>			90.8%	2,268.9MB	0MB/s	0Mbps

위 사진은 오로지 CPU만으로 구동하는 경우입니다. PyCharm에서만 CPU 90.8%, GPU 0% 사용량이 보이나요?

>	PyCharm(5)	27.1%	2,738.4MB	3.6MB/s	0Mbps	63.9%	GPU 0 - Copy
---	------------	-------	-----------	---------	-------	-------	--------------

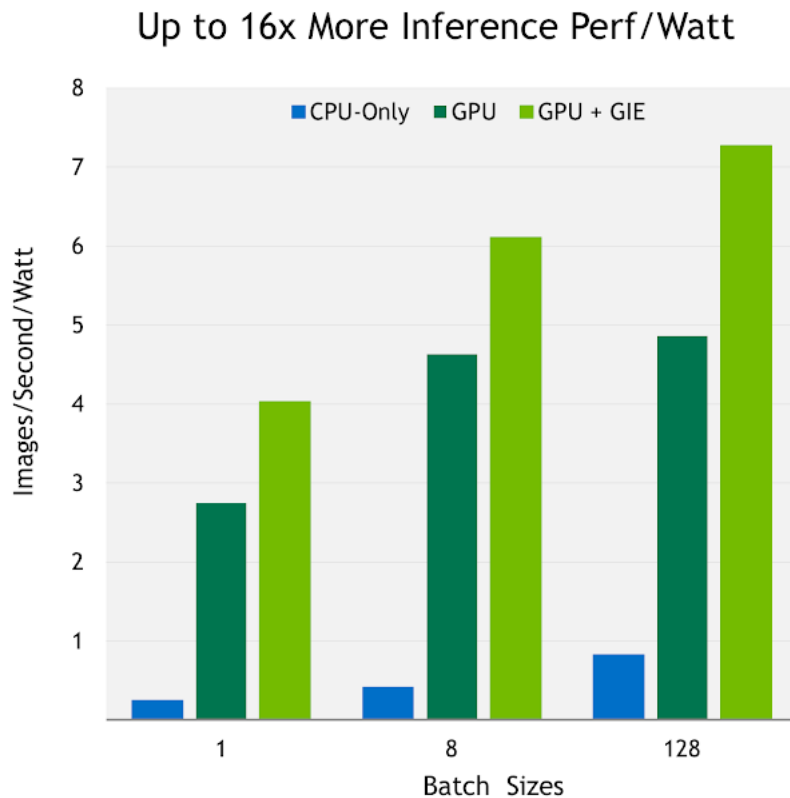
이제 우리가 설치한 tf_gpu를 사용해 구동한 경우를 확인해볼까요?

PyCharm의 CPU 사용량이 27.1%로 줄은 대신 GPU 사용량이 63.9%로 증가함을 볼 수 있습니다.

속도는 실제로 어떨까요?

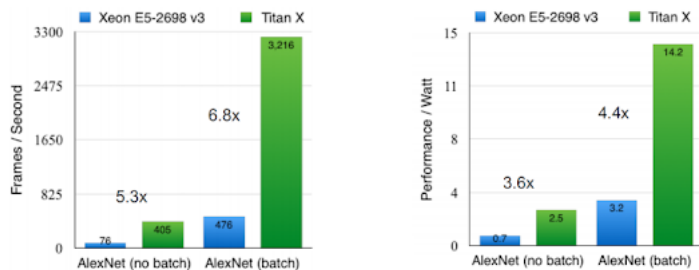
제가 구동시킨 코드는 Keras로 구현된 VGG 19 모델입니다. 나중에 다루겠지만 이미지 관련 모델로 ILSVRC 2014년 Oxford 팀의 팀이름에서 따온 VGG 가 Layer 19개 층을 의미한다고해서 명명된 것이 VGG19인데 이를 사용해 2만 개의 이미지를 train 시켰을 때
CPU만을 구동시킨다면 13000~16000초 즉 3.6시간 ~ 4.4시간이 걸리지만
GPU를 구동시킨다면 대략 20분이면 가능합니다.

엄청난 차이가 보이지 않나요?



Comparing CPU and GPU - server class

Xeon E5-2698 and Tesla M40



NVIDIA Whitepaper "GPU based deep learning inference: A performance and power analysis."

두 개의 이미지 모두 NVIDIA에서 가져온 것인데 batch size가 어땠던 모델이 어떤 것이든 CPU보다 GPU 사용이 훨씬 더 빠르게 해결해준다는 것을 집약적으로 보여줍니다.

사실 이렇게까지 차이가 나는데에는 간단히 축약해서 말하면 GPU가 가진 병렬 연산 처리가 CPU에 비해 월등히 높기 때문에 차이 나는 것 입니다. 이것과 관련해서 아래 NVIDIA의 글과 영상을 꼭 보시길 추천드립니다.

GPU 가속 컴퓨팅이란? - NVIDIA

여기까지 잘 마치셨다면 여러분은 tensorflow gpu를 통해 훨씬 빠른 구동으로 모델을 훈련시키는 딥러닝을 여행하시게 될 것입니다.


앞으로도 행운을 빕니다

읽어주셔서 감사합니다.

Tags [python#](#) [tensorflow#](#)

[Share This](#)

<http://twinstarinfo.blogspot.com/2018/12/tensorflow-gpu-install-nvidia-cuda.html>

 Author Image

About 낭낭편치

Write for expanding Knowledge.

[f](#) [t](#) [G+](#) [@](#) [p](#) [in](#)

[pydot install 문제 해결법](#)

[ResourceExhaustedError: OOM
when allocating tensor with
shape 관련 error 해결법](#)

[tensorflow gpu install - NVIDIA
CUDA 설치하기](#)

[최근 게시물](#)

[이전 게시물](#)

[BLOGGER](#)

[DISQUS](#)

[FACEBOOK](#)

댓글 없음:

댓글 쓰기

항후에 댓글을 관리하려면 Google 계정으로 댓글을 쓰세요. 익명으로 댓글을 쓰면 댓글을 수정하거나 삭제할 수 없습니다. [자세히 알아보기](#)

댓글을 입력하세요...



작성자

[로그아웃](#)

[게시](#)

[미리보기](#)

☐ 알림

RELATED POSTS

[링크 생성](#)

[HOME](#)

About Me

컴퓨터 지식을 포함한 새로운 지식을 확장하고자 만든 블로그입니다. 여러분에게도 이 블로그가 도움이 되길 바랍니다.

[Read More](#)

Archive

Tags

Subscribe

Enter your email address to subscribe this blog and receive notifications of posts by email.

Email address...

[SUBMIT](#)

관심 사용자

관심 사용자(0명)

관심 블로그 등록

PYTHON 코딩 정보 BAEKJOON

PANDAS 지식 확장 정보 컴퓨터 정보

TENSORFLOW 음악 정보 고양이 정보

애니 정보 ANACONDA DB LENOVO

NOBELPRIZE PHOTOSHOP PRAWN

SHRIMP 노벨상 대하 데이터베이스

레노버 분리수거 잡담 책 정보 추석

크림히어로즈 포토샷

Recent News