Neural Network

김 효 관

1. 딥러닝 개발환경 설치

설치 절차

1	Visual C++	텐서플로우는 실제 C기반	vc_redist.x64.exe
2	CUDA * nvidia cuda 지원 그래픽카드 없는경우 스킵	분산처리 플랫폼 (nvidia)	cuda_9.0.176_win10_network.exe
3	cuDNN	소프트웨어 개발 Kit / cuda lib	cudnn-9.0-windows10-x64-v7.4.2.24.zip
4	라이브러리 설치	딥러닝 라이브러리	

CUDA : Compute Unified Device Architecture

윈도우 10 기준 설치 파일은 www.sparkkorea.com AI강의안 자료실 내
01. 프로그램(64비트운영체제) -> 6. 텐서플로우 cuda 에서 한번에 다운 가능함

1. 딥러닝 개발환경 설치

설치 절차

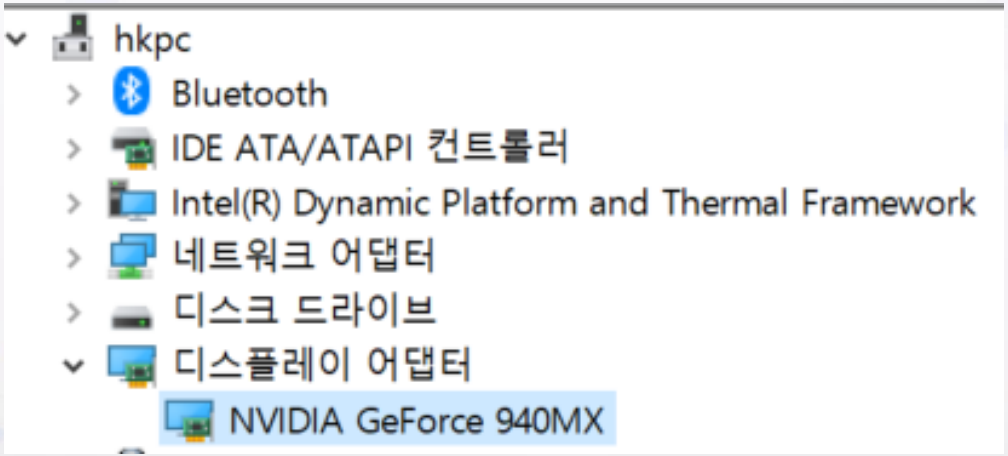
1	Visual C++	텐서플로우는 실제 C기반	vc_redist.x64.exe
2	CUDA * nvidia cuda 지원 그래픽카드 없는경우 스킵	분산처리 플랫폼 (nvidia)	cuda_9.0.176_win10_network.exe
3	cuDNN	소프트웨어 개발 Kit / cuda lib	cudnn-9.0-windows10-x64-v7.4.2.24.zip
4	라이브러리 설치	딥러닝 라이브러리	

CUDA : Compute Unified Device Architecture

윈도우 10 기준 설치 파일은 www.sparkkorea.com AI강의안 자료실 내
01. 프로그램(64비트운영체제) -> 6. 텐서플로우 cuda 에서 한번에 다운 가능함

Nvidia 확인

설치 준비



1-1. Visual C++

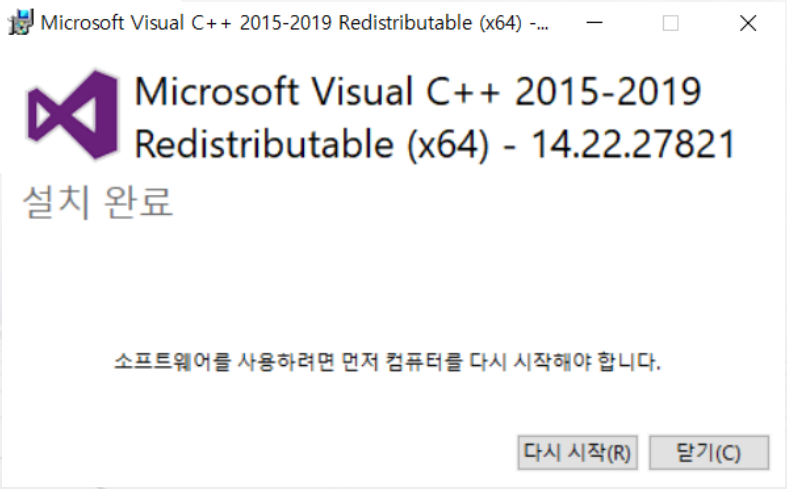
설치 방법

링크 : https://support.microsoft.com/ko-kr/help/2977003/the-latest-supported-visual-c-downloads

Visual Studio 2015, 2017 및 2019

Visual Studio 2015, 2017 및 2019용 Microsoft Visual C++ 재배포 가능 패키지를 다운로드합니다. 다음 업데이트는 Visual Studio 2015, 2017 및 2019용으로 지원되는 최신 Visual C++ 재배포 가능 패키지입니다. Universal C Runtime의 기본 버전이 포함되어 있습니다. 자세한 내용은 MSDN을 참조하세요.

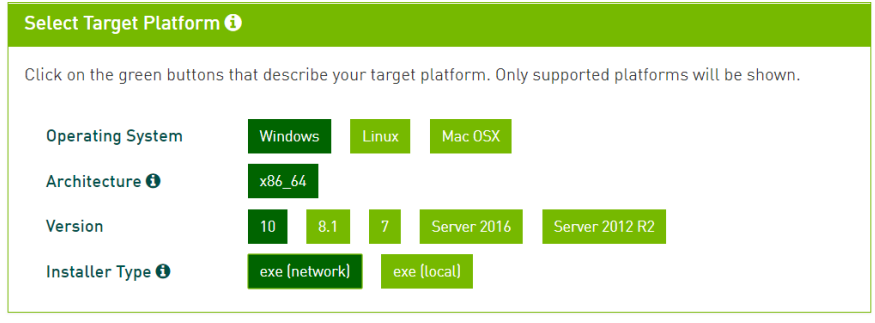
- x86: vc_redist.x86.exe
- x64: vc_redist.x64.exe
- ARM64: vc_redist.arm64.exe



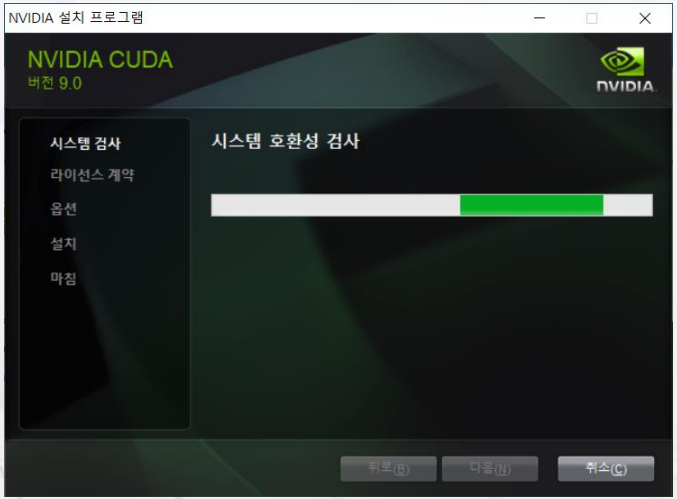
1-2. CUDA

설치 방법

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

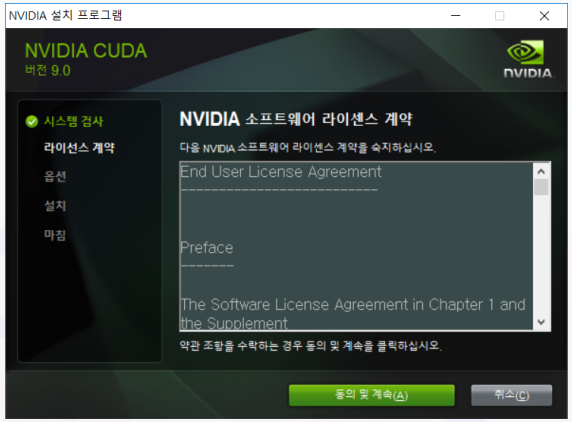


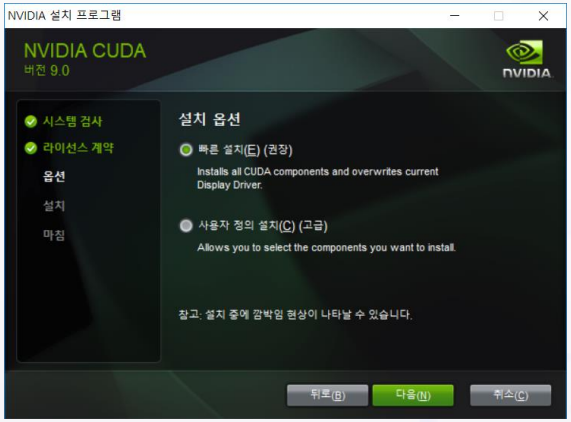
빠른설치 선택



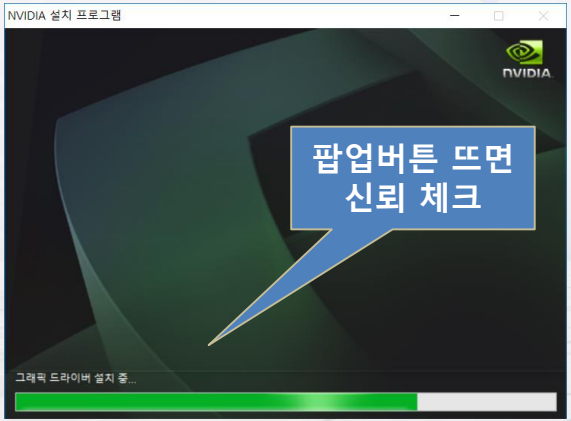
설치참조: https://pythonkim.tistory.com/137
http://twinstarinfo.blogspot.com/2018/12/tensorflow-gpu-install-nvidia-cuda.html

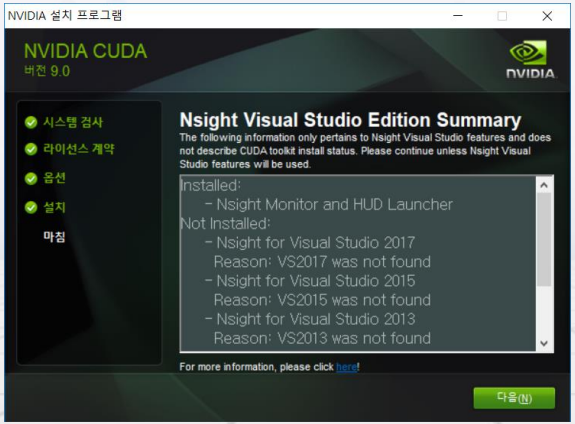
1-2. CUDA

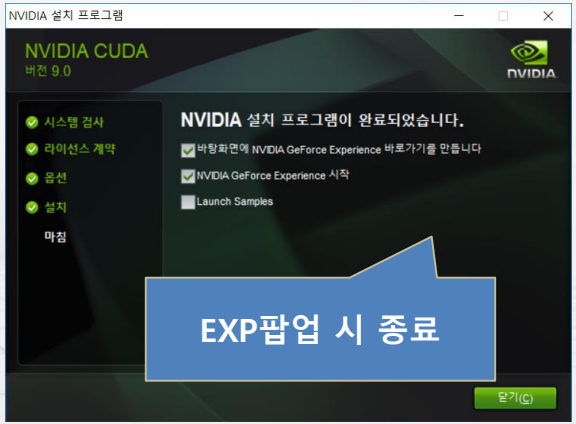

NVIDIA 설치 프로그램
NVIDIA CUDA
버전 9.0
NVIDIA
시스템 검사
라이선스 계약
옵션
설치
마침
NVIDIA 소프트웨어 라이센스 계약
다음 NVIDIA 소프트웨어 라이센스 계약을 숙지하십시오.
End User License Agreement
Preface
The Software License Agreement in Chapter 1 and the Supplement
약관 조항을 수락하는 경우 동의 및 계속을 클릭하십시오.
동의 및 계속(A)
취소(C)


NVIDIA 설치 프로그램
NVIDIA CUDA
버전 9.0
NVIDIA
시스템 검사
라이선스 계약
옵션
설치
마침
설치 옵션
빠른 설치(E) (권장)
Installs all CUDA components and overwrites current Display Driver.
사용자 정의 설치(C) (고급)
Allows you to select the components you want to install.
참고: 설치 중에 깜박임 현상이 나타날 수 있습니다.
뒤로(B)
다음(N)
취소(C)


NVIDIA 설치 프로그램
NVIDIA CUDA
버전 9.0
NVIDIA
시스템 검사
라이선스 계약
옵션
설치
마침
CUDA Visual Studio Integration
No supported version of Visual Studio was found. Some components of the CUDA Toolkit will not work properly. Please install Visual Studio first to get the full functionality.
I understand, and wish to continue the installation regardless.
뒤로(B)
NEXT
취소(C)


NVIDIA 설치 프로그램
NVIDIA
팝업버튼 뜨면
신뢰 체크
그래픽 드라이버 설치 중...

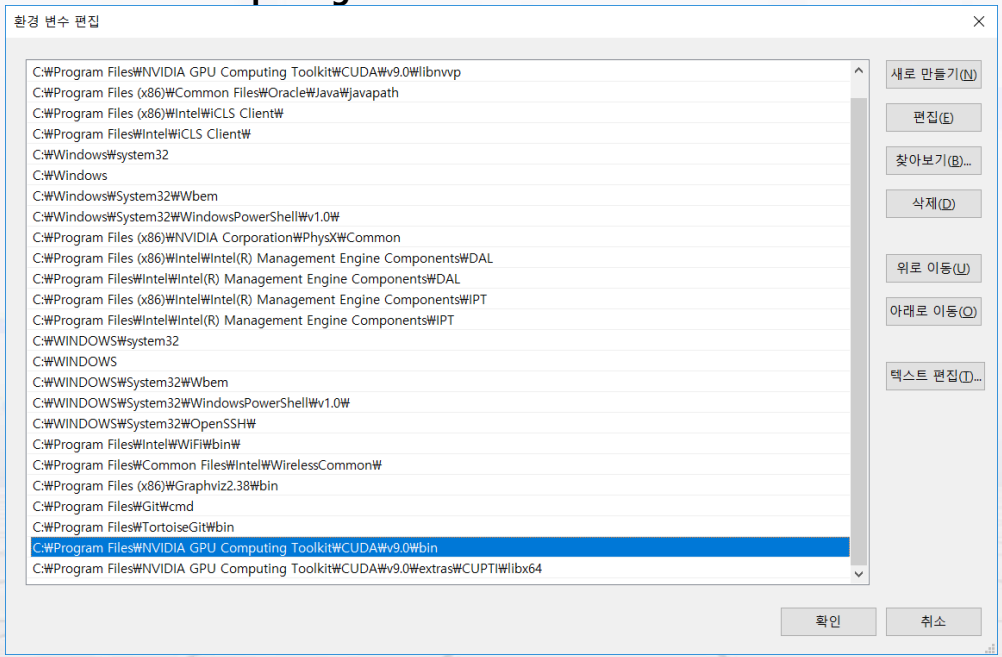

NVIDIA 설치 프로그램
NVIDIA CUDA
버전 9.0
NVIDIA
시스템 검사
라이선스 계약
옵션
설치
마침
Nsight Visual Studio Edition Summary
The following information only pertains to Nsight Visual Studio features and does not describe CUDA toolkit install status. Please continue unless Nsight Visual Studio features will be used.
Installed:
- Nsight Monitor and HUD Launcher
Not Installed:
- Nsight for Visual Studio 2017
Reason: VS2017 was not found
- Nsight for Visual Studio 2015
Reason: VS2015 was not found
- Nsight for Visual Studio 2013
Reason: VS2013 was not found
For more information, please click here!
다음(N)


NVIDIA 설치 프로그램
NVIDIA CUDA
버전 9.0
NVIDIA
시스템 검사
라이선스 계약
옵션
설치
마침
NVIDIA 설치 프로그램이 완료되었습니다.
바탕화면에 NVIDIA GeForce Experience 바로가기를 만듭니다
NVIDIA GeForce Experience 시작
Launch Samples
EXP팝업 시 종료
닫기(C)

1-2. CUDA

환경변수 등록 (개인별 cuda 설치 폴더 다를 수 있음)

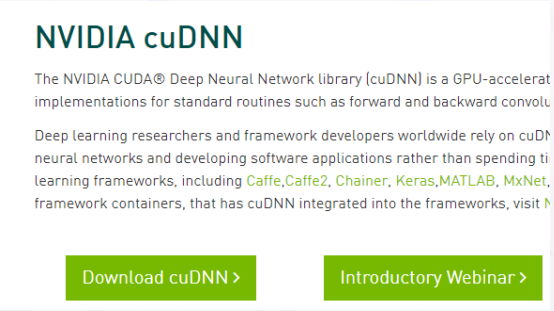
C:₩Program Files₩NVIDIA GPU Computing Toolkit₩CUDA₩v9.0₩bin
C:₩Program Files₩NVIDIA GPU Computing Toolkit₩CUDA₩v9.0₩extras₩CUPTI₩libx64

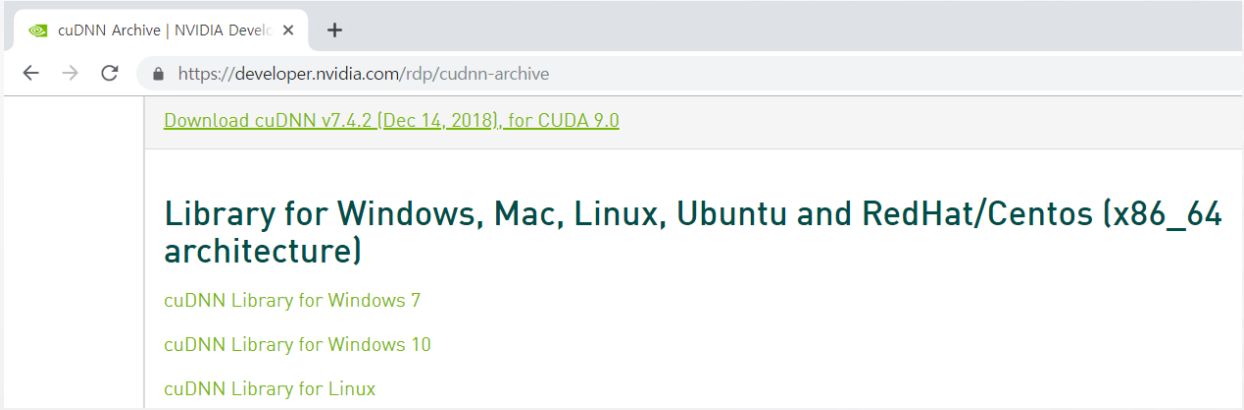


1-3. cuDNN

설치 방법

링크 : https://developer.nvidia.com/cudnn



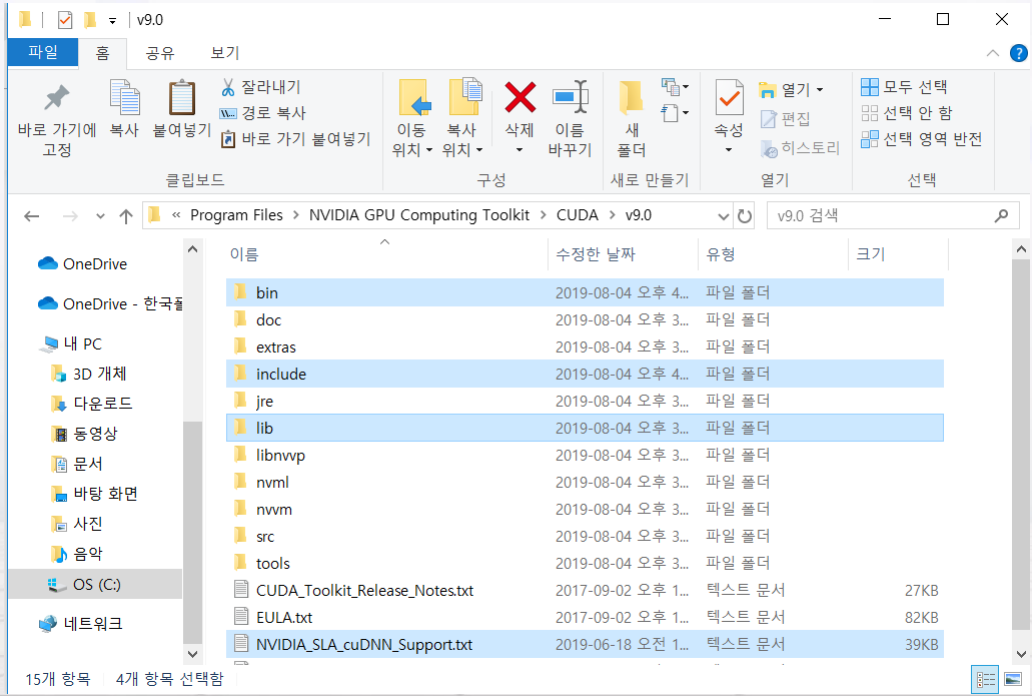


회원가입 필요

1-3. cuDNN

설치파일 복사

cuDNN 압축해제 후 아래 폴더 내용을 CUDA 설치 폴더 내 복사



1-4. 라이브러리 설치

라이브러리 호환성 맞춰 설치

텐서플로우와 호환성이 맞는 파이썬 3.6 downgrade 및 tensorflow 호환 버전 설치

```
# 아나콘다 downgrade
conda install python=3.6
# tensorflow 설치
pip install tensorflow-gpu==1.12
pip install keras==2.2.4
```

conda update conda 실행 후 진행 "requests"..error
io_loop error 시 pip install tornado==4.5.3

가상환경 생성 후 진행해도 무방
conda create –n haiteam pip puthon=3.6
conda activate haiteam

```
(base) C:₩Users₩hk>
(base) C:₩Users₩hk>pip install tensorflow-gpu==1.12
Collecting tensorflow-gpu==1.12
  Using cached https://files.pythonhosted.org/packages/88/73/13e4071739df8d5ee7a27780d66bc98a516125
21ad7e5a1e468d9507087c/tensorflow_gpu-1.12.0-cp36-cp36m-win_amd64.whl
Collecting protobuf>=3.6.1 (from tensorflow-gpu==1.12)
  Using cached https://files.pythonhosted.org/packages/74/74/44ec96740ed10ae6d0508efc083c6b7e605c50
9bc32136e9aea840d09daf/protobuf-3.9.0-cp36-cp36m-win_amd64.whl
```

pip install –upgrade sklearn
pip install –upgrade scipy

1-4. 라이브러리 설치

라이브러리 확인

import tensorflow as tf

from tensorflow.python.client import device_lib

device_lib.list_local_devices()

```
import tensorflow as tf

from tensorflow.python.client import device_lib

device_lib.list_local_devices()
```

```
[name: "/device:CPU:0"
 device_type: "CPU"
 memory_limit: 268435456
 locality {
 }
 incarnation: 18268563969281400942, name: "/device:GPU:0"
 device_type: "GPU"
 memory_limit: 1472089292
 locality {
   bus_id: 1
   links {
   }
 }
 incarnation: 6618853039916440327
 physical_device_desc: "device: 0, name: GeForce 940MX, pci bus id: 0000:01:00.0, compute capability: 5.0"]
```

1-4. 라이브러리 설치

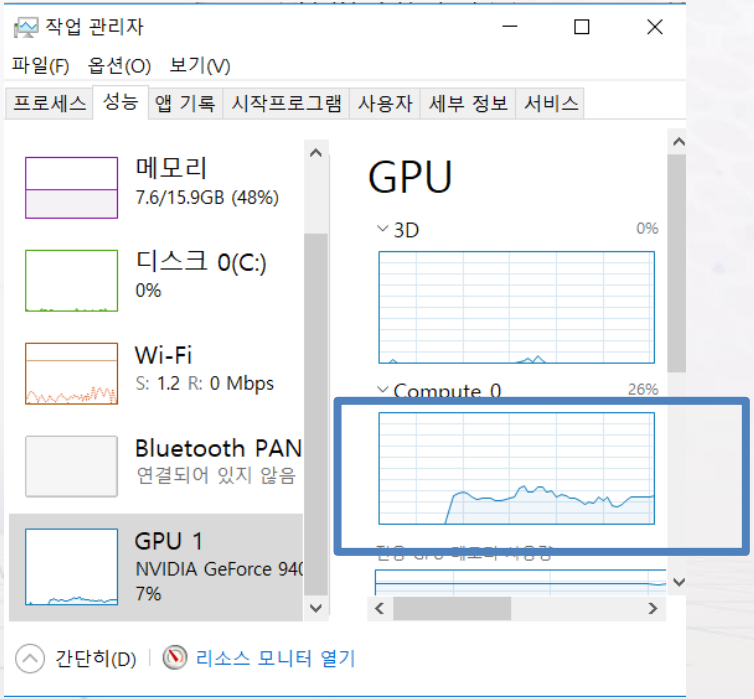
테스트모델 생성 및 확인

```
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_7 (Dense)              (None, 64)                320
_________________________________________________________________
dense_8 (Dense)              (None, 64)                4160
_________________________________________________________________
dense_9 (Dense)              (None, 1)                 65
=================================================================
Total params: 4,545
Trainable params: 4,545
Non-trainable params: 0
_________________________________________________________________
```

```
model.compile(loss='mean_squared_error',
              optimizer="adam",
              metrics=['mean_absolute_error', 'mean_squared_error'])
```

```
from keras.callbacks import EarlyStopping
#set early stopping monitor so the model stops training when it won't improve anymore
early_stopping_monitor = EarlyStopping(patience=2000)
EPOCHS = 2000
#train model
history =model.fit(normed_train_data, train_labels, validation_split=0.2,
                   epochs= EPOCHS, callbacks=[early_stopping_monitor])
```



참조. plaidml

1. 설치

1. cmd 창 open
2. plaidml 설치
   - pip install plaidml-keras plaidbench
3. plaidml 설정
   - plaidml-setup
   * Enable experimental device support : Y
   * GPU 번호 선택
   * Save Setting: Y
4. 커멘드 창에서 확인
   - plaidbench keras mobilenet

```
C:₩Users₩kopo>plaidbench keras mobilenet
Running 1024 examples with mobilenet, batch size 1, on backend plaid
INFO:plaidml:Opening device "opencl_intel_hd_graphics_620.0"
Compiling network...INFO:plaidml:Analyzing Ops: 280 of 413 operations complete
 Warming up... Running...
Example finished, elapsed: 13.652s (compile), 63.389s (execution)

--------------------------------------------------------------------------------
--------
Network Name          Inference Latency          Time / FPS
--------------------------------------------------------------------------------
--------
mobilenet             61.90 ms                   354.85 ms / 2.82 fps
Correctness: PASS, max_error: 9.247387424693443e-06, max_abs_error: 3.3155083656
11035e-07, fail_ratio: 0.0
```

참조. plaidml

2. 확인

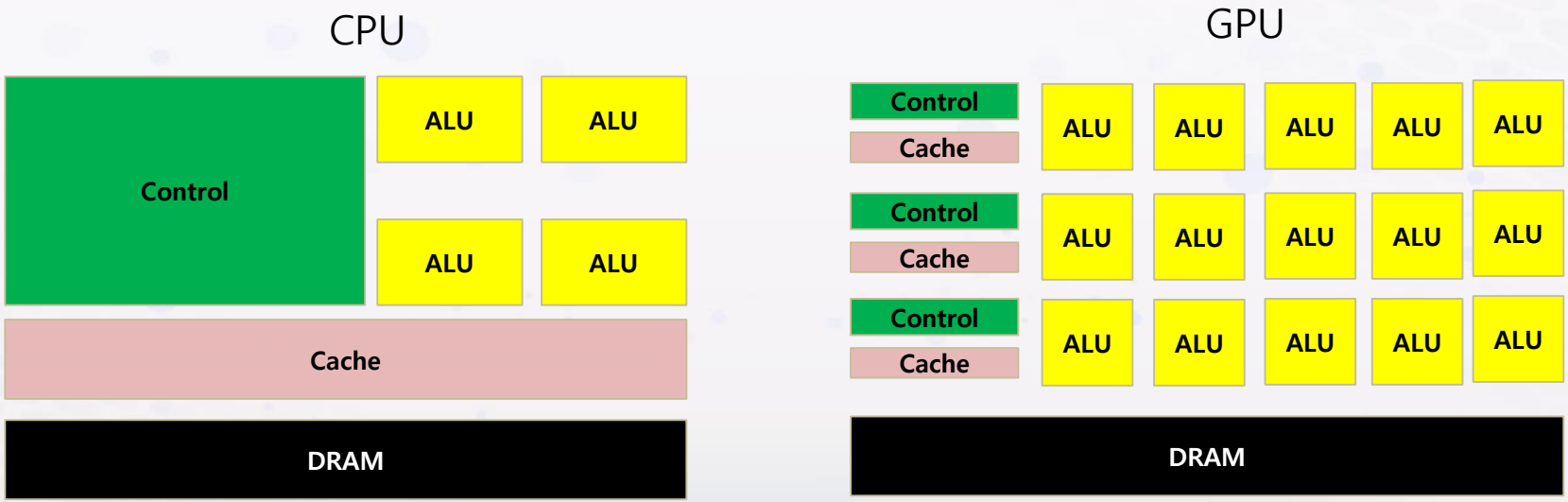
```
import numpy as np
import os
import time

os.environ["KERAS_BACKEND"] = "plaidml.keras.backend"

import keras
import keras.applications as kapp
from keras.datasets import cifar10

(x_train, y_train_cats), (x_test, y_test_cats) = cifar10.load_data()
batch_size = 8
x_train = x_train[:batch_size]
x_train = np.repeat(np.repeat(x_train, 7, axis=1), 7, axis=2)
model = kapp.VGG19()
model.compile(optimizer='sgd', loss='categorical_crossentropy',
              metrics=['accuracy'])

print("Running initial batch (compiling tile program)")
y = model.predict(x=x_train, batch_size=batch_size)

# Now start the clock and run 10 batches
print("Timing inference...")
start = time.time()
for i in range(10):
    y = model.predict(x=x_train, batch_size=batch_size)
print("Ran in {} seconds".format(time.time() - start))
```

```
1  import numpy as np
2  import os
3  import time
4
5  os.environ["KERAS_BACKEND"] = "plaidml.keras.backend"
6
7  import keras
8  import keras.applications as kapp
9  from keras.datasets import cifar10
10
11 (x_train, y_train_cats), (x_test, y_test_cats) = cifar10.load_data()
12 batch_size = 8
13 x_train = x_train[:batch_size]
14 x_train = np.repeat(np.repeat(x_train, 7, axis=1), 7, axis=2)
15 model = kapp.VGG19()
16 model.compile(optimizer='sgd', loss='categorical_crossentropy',
17               metrics=['accuracy'])
18
19 print("Running initial batch (compiling tile program)")
20 y = model.predict(x=x_train, batch_size=batch_size)
21
22 # Now start the clock and run 10 batches
23 print("Timing inference...")
24 start = time.time()
25 for i in range(10):
26     y = model.predict(x=x_train, batch_size=batch_size)
27 print("Ran in {} seconds".format(time.time() - start))
```

```
Using plaidml.keras.backend backend.
```

참조. CPU vs GPU

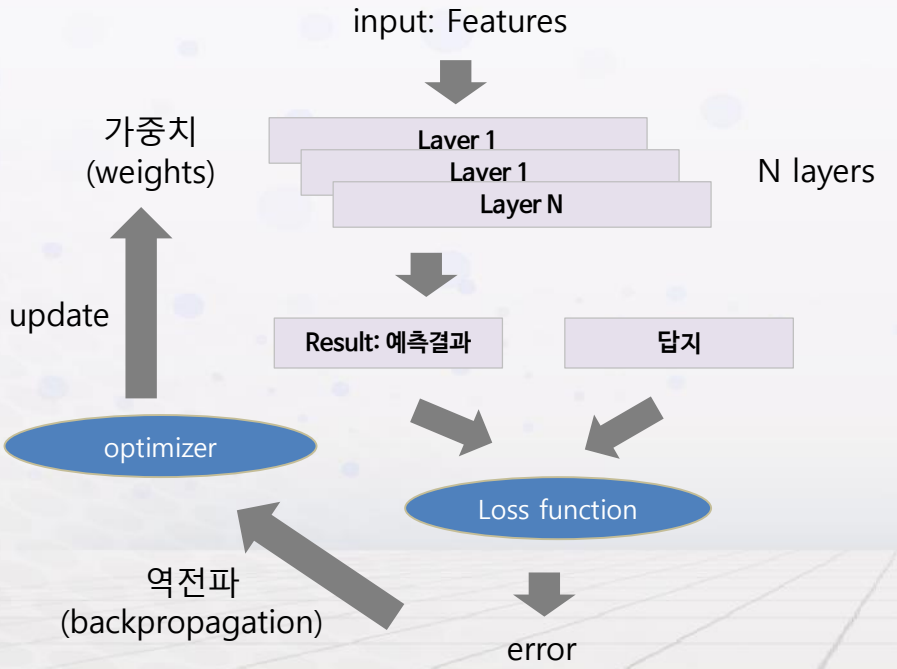
CPU vs GPU



GPU는 코어가 많음
CPI는 복잡한 계산을 빠르게 하지만 직렬 처리,GPU는 많은 연산을 병렬로 처리함

참조. 딥러닝 작동 원리

딥러닝 프로세스



주요용어	내용
답지	알고있는 Label
가중치	가장 효율적인 식 y=wx+b
손실함수 (Loss function)	예측결과와 실제값의 차이를 계산
역전파 (Backpropatation)	손실함수의 결과 개선을 위해 가중치 수정과정 (optimizer 담당)

감사합니다.