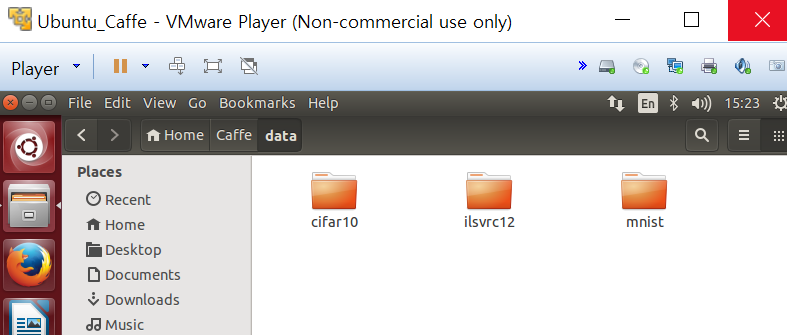
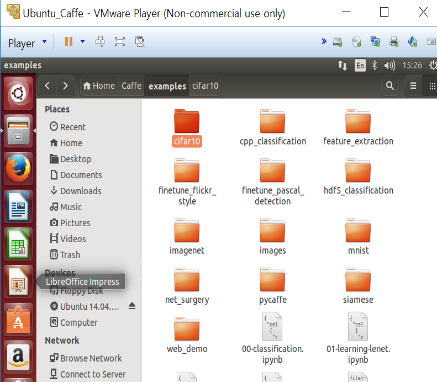
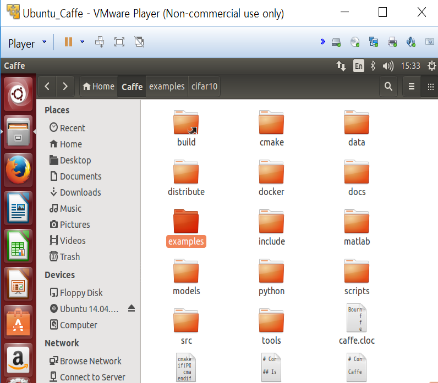
**Caffe를 사용하는 방법 보충 설명 (2부)**

1. **Caffe 폴더 구성에 대한 이해**

1부에서 테스트를 위한 자료를 준비하셨다면 지금부터는 Caffe프레임웍에서 이들을 사용하는 방법을 숙지할 차례입니다. Caffe를 사용하기에 앞서서 우선 Caffe에 사용되는 data등에 대해서 우선 이해를 하고 넘어가야 부분이 있습니다. 다양한 사이트에서 Caffe사용 관련 자료를 찾다보면 데이터를 다루는데 있어서 혼동이 가는 부분이 있습니다. 바로 데이터 셋에 관한 설명입니다. Caffe 설치하시고 폴더를 열어보시면 Caffe>data 안에 아래와 같은 3개의 폴더를 확인할 수 있습니다.



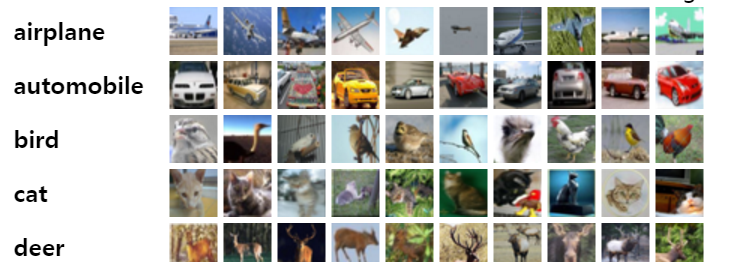
* Caffe>examples 폴더에서도 cafar10, imagenet, mnist를 찾을 수 있습니다.

1. **mnist, Cifar10, imagenet에 대하 이해**

* **mnist** : mnist는 간단한 컴퓨터 비전 데이터 세트로, 아래와 같이 손으로 쓰여진 이미지들로 구성되어 있습니다.

위의 이미지에는 각각 5, 0, 4, 1 이라는 라벨이 붙어 있습니다. MNIST 데이터는 [Yann LeCun](http://yann.lecun.com/exdb/mnist/)의 웹사이트에서 제공합니다. MNIST 데이터셋은 손 글씨에 의한 데이터셋을 제공하고 있으며 트레이닝 셋트 60,000 샘플 그리고 테스트 셋트 10,000 샘플으 제공합니다. 트레이닝 셋트는 30,000 SD-3패턴과 30,000 SD-1들로 구성이 되어 있습니다. 28 by 28 짜리 handwrite digit을 모아 놓은 데이터셋입니다.

* CIFAR-10 : 과 CIFAR-100는 8천만개의 작은 이미지로 구성이 된 데이터 서브셋입니다. 이 이미지들은 Alex Krizhevsky, Vinod Nair 그리고 Geoffrey Hinton 교수들에 의해 수집이 되었습니다. CIFAR-10 데이터셋은 60000 32x32 칼라이미지가 10개의 클라스에 분류되어 있고 각 클라스에는 6000 이미지 데이터가 들어가 있습니다. 그리고 10000개의 테스트 이미지로 구성되어 있습니다.



* Imagenet : 사실 데이터셋의 이름이 아니라 매년 새로운 task가 주어지는 competition입니다. 보통 실험에 사용하는건 [2012년 데이터셋 (ILSVRC 2012)](http://image-net.org/challenges/LSVRC/2012/index)인데, training 데이터가 1000개 class에 데이터 개수는 거의 128만개 가까이 되는 엄청나게 큰 데이터 셋입니다. Test data는 공개되지 않았고, 대신 validation set으로 공개된 데이터는 총 5만개짜리 데이터입니다. 전체 데이터 사이즈는 거의 150GB가까이 됩니다. [ILSVRC 2015](http://image-net.org/challenges/LSVRC/2015/)에는 더 이상 classification task가 존재하지 않고, object detection이나 object localization등의 task만 주어져 있는 상태입니다.

1. **Caffe에서 data path를 변경해주기**

* 모든 경로와 명령어는 root Caffe에서 실행이 되도록 하는 것이 시행착오를 줄이는 길입니다.
* “ImageNet”은 앞서 말씀드린 것처럼 ILSVRC12 challenge를 의미하지만 또한 학습을 진행할 수도 있습니다. 이미 트레이닝 데이터와 평가데이터를 다운받은 것으로 알고 진행을 하겠습니다.

/path/to/imagenet/train/n01440764/n01440764\_10026.JPEG

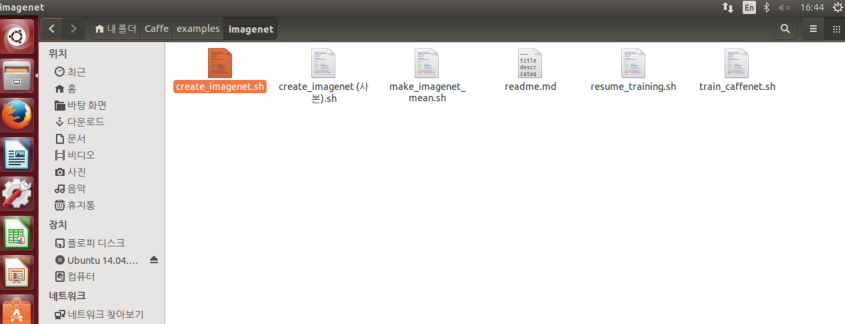
/path/to/imagenet/val/ILSVRC2012\_val\_00000001.JPEG

* 학습을 위해서 약간의 보조 데이터가 필요하게 됩니다.

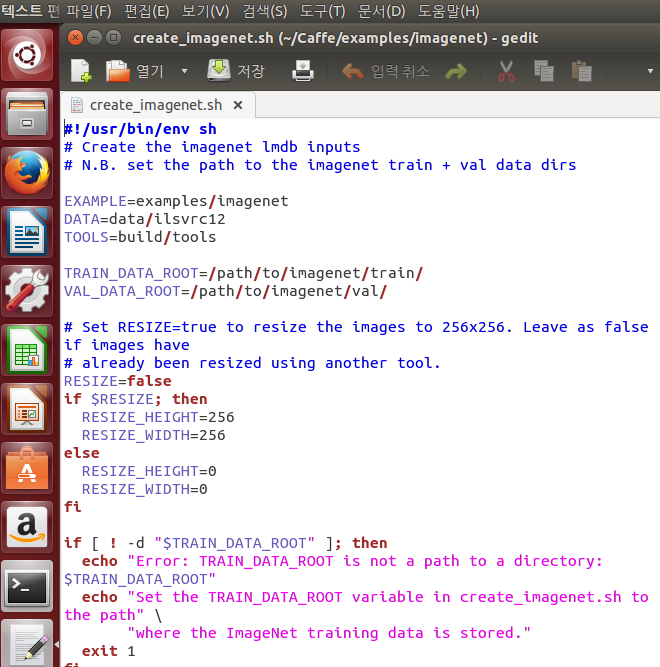
./data/ilsvrc12/get\_ilsvrc\_aux.sh

* 트레이닝 데이터와 테스트 데이터는 train.txt 과 val.txt에 기술하고 라벨링을 합니다. (1부에 세부적인 방법을 설명을 하였습니다.)

Caffe/examples/imagenet에 있는 create\_imagenet.sh파일을 열어서 아래 5가지를 자기자신이 저장해 놓은 root에 맞도록 수정을 합니다. (복사본을 만들어서 수정을 하는 것을 권해드립니다. 잘못되어 실행이 안될 경우는 복사한 파일을 다시 사용할 수 있습니다.)



* 리눅스 화면에서 내 폴더에 들어가 Caffe>examples>Imagenet을 연 화면

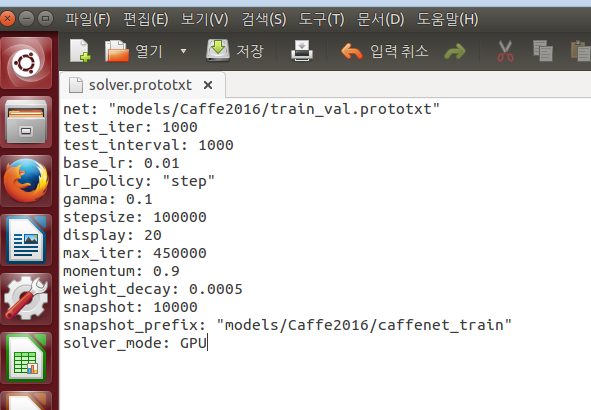
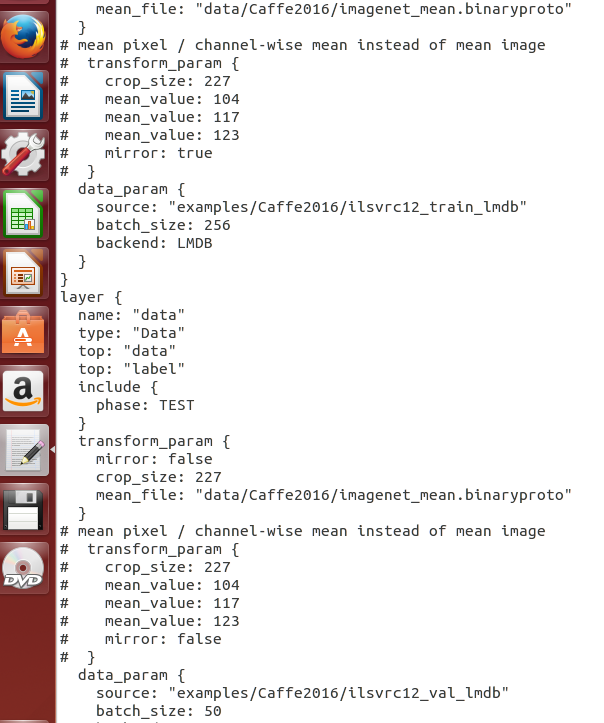


* EXAMPLE=examples/imagenet # LMDB파일이 생성이 되면 저장이 되는 곳
* DATA=data/ilsvrc12 # 레이블링 값이 적힌 .txt 파일이 저장되어 있는 곳
* TOOLS=build/tools # convert\_imageset파일이 저장되어 있는 곳
* TRAIN\_DATA\_ROOT=/path/to/imagenet/train/ # Train JPEG 원본 파일이 저장되어 있는 곳
* VAL\_DATA\_ROOT=/path/to/imagenet/val/ # val JPEG 원본 파일이 저장되어 있는 곳 (1부에서 데이터를 지정 시 train\_dataset / val\_dataset으로 합니다.

\*\* 준비한 이미지들의 사이즈가 256x256이 아니라면 **resize변수 값을 True로 변경**합니다.

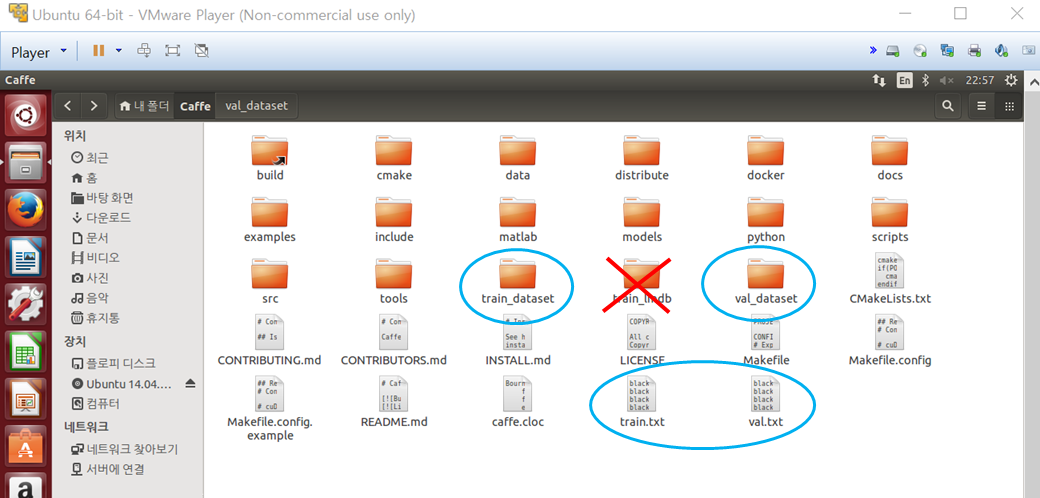
\*\* lmdb파일은 실행을 하기 전에 examples/imagenet/train\_lmdb 그리고 examples/imagenet/val\_lmdb 가 있으면 안됩니다.

1. **기타 설정하기**
2. /examples/imagenet/make\_imagenet\_mean.sh 파일을 원본 복사해서 편집합니다. 여기서 자신이 생성한 LMDB폴더의 경로를 EXAMPLE변수에 정확하게 기입합니다. 그리고 make\_imagenet\_mean.sh파일을 실행합니다.
3. /models/bvlc\_reference\_caffenet/의 위치에 있는 train\_val.prototxt파일을 복사해서 변경한다. Train\_val.protxt파일을 확인해 보면 2군데의 imagenet\_mean.binaryproto파일 경로와 2군데(val,train)의 LMDB파일의 경로를 적는 곳이 있습니다. 정확히 자신의 경로에 맞게 기입합니다.



* Solver.prototxt에서 총 이터레이션 반복 횟수, 테스트 인터벌 등을 조절할 수 있습니다. (실행 : ./build/tools/caffe train – solver=models/bvlc\_reference\_caffenet/solver.prototxt)

1. **Convert\_imageset를 사용하여 이미지 파일을 Caffe에서 사용이 가능한 lmdb 또는 leveldb파일로 전환하기**
2. 작업해 놓은 train\_dataset 및 val\_dataset폴더를 Caffe root에 복사를 합니다. 폴더를 복사한 후 우분투 화면에서 붙여넣기를 하시면 됩니다.



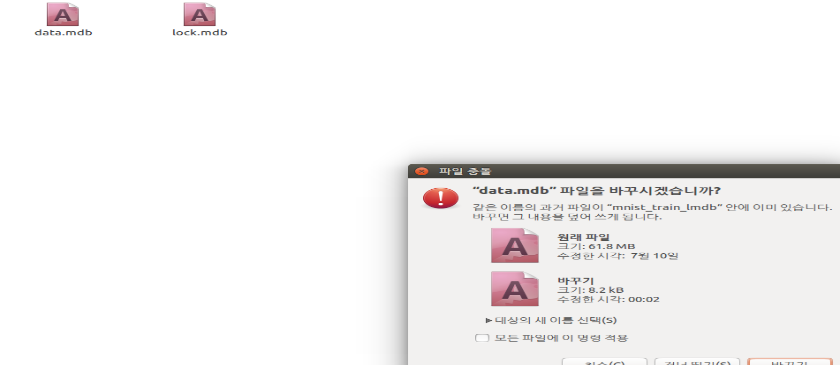
1. 다음은 우분투 terminal 화면에서 다음과 같은 명령어를 실행을 합니다.

|  |
| --- |
| ./build/tools/convert\_imageset <이미지디렉토리> <리스트파일> <보존된 곳> <이미지셔플 여부> --backend <변환방법> <높이> <폭> |

* 이미지 디렉토리 : 이미지 담겨져 있는 폴더명 => train\_dataset
* 리스트파일 : 트레이닝 데이터의 라벨링 txt 파일 => train.txt
* 보존된 곳 : 변환된 파일이 저장될 폴더명 => train\_lmdb
* 이미지 셔플 여부 : 0 셔플안하다. 1 셔플한다 => 1
* 변화방법 : 어덯게변환할 것인가? lmdb or leveldb => lmdb
* 높이 : 이미지를 resize하는 크기 => 28
* 폭 : 이미지를 resize하는 크기 => 28

|  |
| --- |
| ./build/tools/convert\_imageset train\_dataset train.txt train\_lmdb 1 --backend lmdb 28 28 |
| ./build/tools/convert\_imageset val\_dataset val.txt val\_lmdb 1 --backend lmdb 28 28 |

1. **lmdb파일을 examples\imagenet\lmdb폴더 파일과 교체하기**
2. Caffe root에서 만들어진 lmdb파일을 imagenet폴더에 있는 lmdb에 각각 덮어쓰기를 합니다. 기존의 lmdb 는 사본으로 저장을 합니다.



1. **이제 Caffe에서 imagenet를 학습할 차례입니다. 다시 5.3.1로 돌아가서 학습을 실행합니다.** 5-11을 따라서 직접 만든 lmdb파일을 이용하여 학습을 진행하여 학습모델(필터)를 만들어 냅니다.

|  |
| --- |
| ./examples/imagenet/train\_lenet.sh |

1. **모델을 이용하여 새로운 이미지가 입력이 되었을 때 어떤 결과를 보여주는지 확인합니다.**
2. ImageNet 의 경우 models/bvlc\_reference\_caffenet/deploy.prototxt 에 분류를 위한 모델정의를 담았고

 models/bvlc\_reference\_caffenet/bvlc\_reference\_caffenet.caffemodel  이것이 1번에서 최종적으로 만든 모델인데 이것도 물론 사용됩니다.

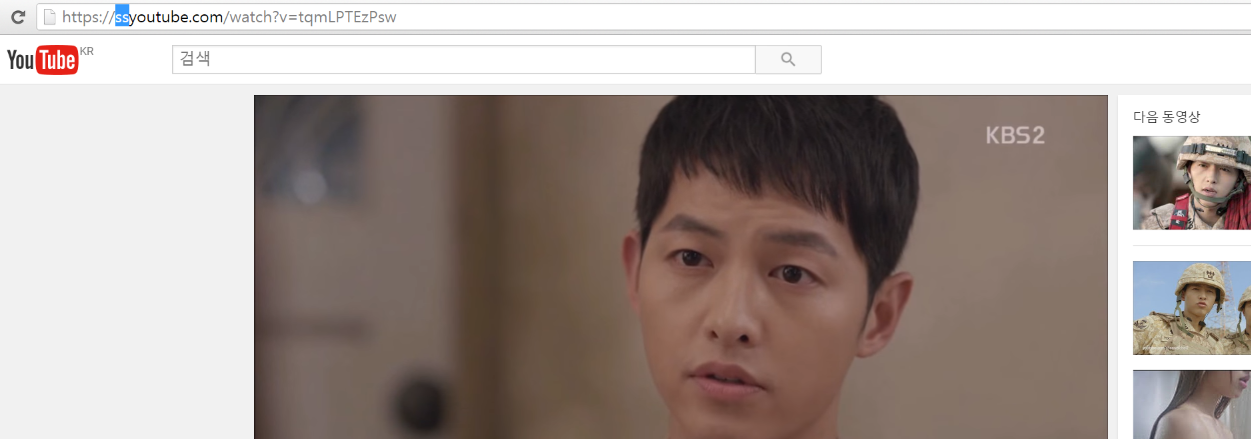
1. Caffe 를 처음 설치하면 Caffe 의 root 로 가서 간단히 run cat1.jpg 하면 고양이를 인식하게 되는데 run 의 내부에는 python/classify.py 를 호출하도록 되있습니다. classify.py 내부를 보면 아래와 같으며 , 내용을 가져다 우리의 데이터 분류에 사용하면 됩니다.

<http://caffe.berkeleyvision.org/gathered/examples/imagenet.html>

* **Youtube에서 원하는 동영상을 다운받는 방법.**
* <http://www.youtube.com/wach>......여기에 http://ssyoutube.com/wach....

으로 바꾸어 주시면 손쉽게 동영상을 다운 받을 수 있습니다.

* 이후 동영상 JPEG변환 프로그램을 통해서 이미지 파일을 얻어내실 수 있습니다.



* 다운로드 화면

