Installing and working with caffe – CPU Version for testing different nets.

Durvesh Pathak

Electrical & Computer Engineering

Submitted to Dr. Mohamed El - Sharkawy

Index

Abstract:

This report aims at accelerating the process of installing caffe framework to test different types of Neural Networks.

This Report outlines the basic process of installing caffe on Ubuntu 17.04 ubuntu. You can also find information on http://caffe.berkeleyvision.org/install_apt.html. We will install caffe and run mnist example to check the installation.

Please note that this is not the installation process for Training the network that has to be done using GPU's. I will shortly start compiling process on training squeezenet.

Caffe: caffe is a deep learning framework developed by Berkeley AR research team. Which we will be using to train and deploy our Squeezenet V1.0.

Pl NOTE : the operating system used is 17.04

Pre Requisites:

1. Git

Git is a crucial requirement to clone repo from Github following command will install latest git on your system.

```
$ sudo apt-get install git
```

2. Python 2.7

Most Ubuntu installation comes with python 2.7.x but if you want to work on virtual environments install anaconda and the procedure for installation is available on this link - > https://yangcha.github.io/Caffe-Conda/

3. Opencv 3.x

To install this package type In the following commands

```
$  pip install opencv-python

$ pip install pillow

$ sudo apt-get install python-tk
```

You will need the following dependencies to run caffe:

```
$ sudo apt-get install -y --no-install-recommends libboost-all-dev
$ sudo apt-get install libprotobuf-dev libleveldb-dev libsnappy-dev libopencv-dev libboost-all-dev
$ sudo apt-get install vim
$ sudo apt-get install libhdf5-serial-dev libgflags-dev libgoogle-glog-dev liblmdb-dev protobuf-compiler
$ sudo apt-get install libopenblas-dev
$ sudo apt-get install libatlas-dev-dev
$ sudo apt install python-pip
$ sudo pip install scikit-image protobuf
```

Next, we will have to install all the necessary Python packages, using *pip*. Navigate to *python* folder, and type the line below:

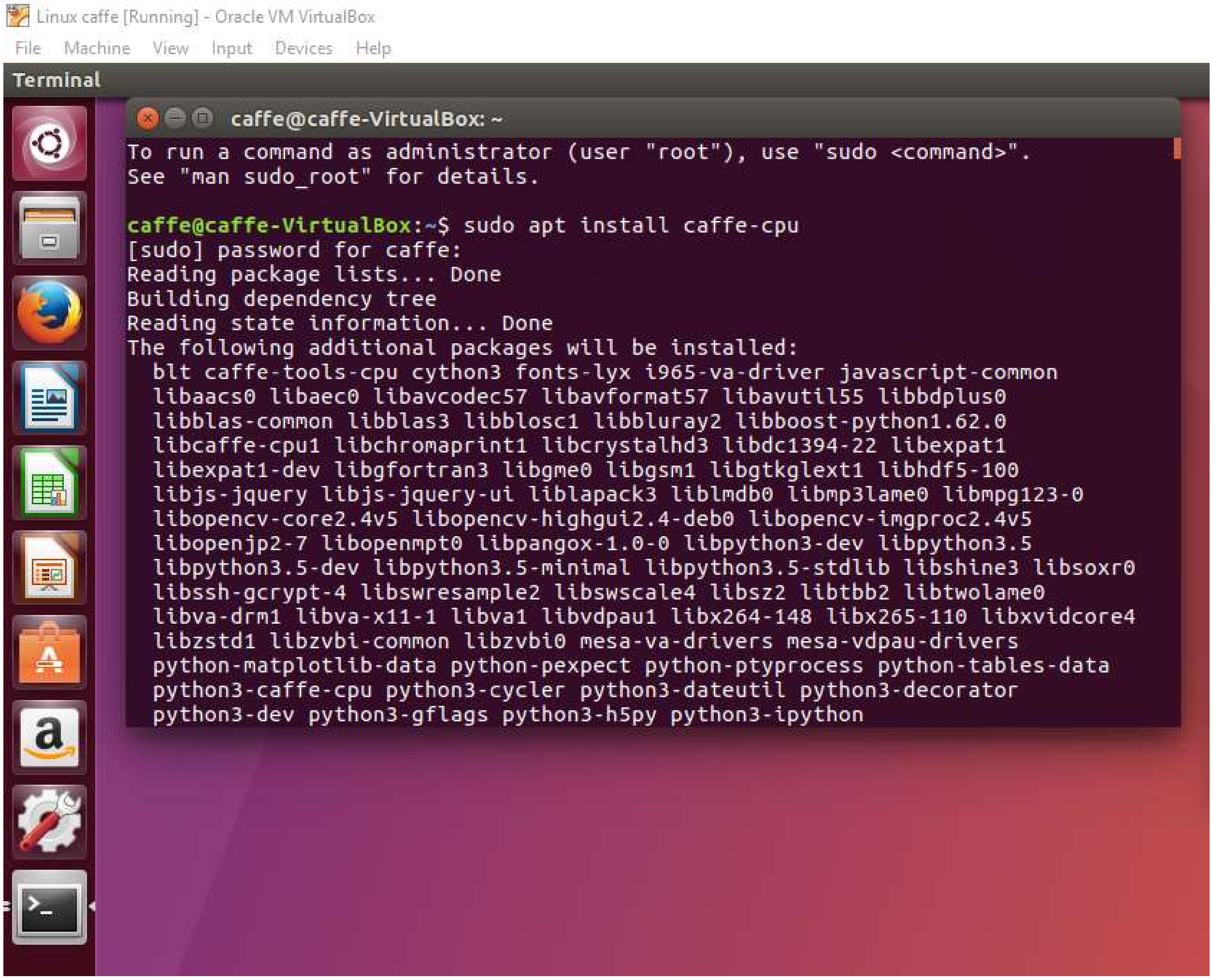
```
$ cd python(Paths may be different for you)
$ for req in $(cat requirements.txt); do sudo pip install $req; done
```

Installation:

The very first thing to do is install caffe packages.

Everything including caffe itself is packaged in 17.04 and higher versions. To install pre-compiled Caffe package, just do it by

$ sudo apt install caffe-cpu


Linux caffe [Running] - Oracle VM VirtualBox
File
Machine
View
Input
Devices
Help
Terminal
caffe@caffe-VirtualBox: ~

```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

caffe@caffe-VirtualBox:~$ sudo apt install caffe-cpu
[sudo] password for caffe:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  blt caffe-tools-cpu cython3 fonts-lyx i965-va-driver javascript-common
  libaacs0 libaec0 libavcodec57 libavformat57 libavutil55 libbdplus0
  libblas-common libblas3 libblosc1 libbluray2 libboost-python1.62.0
  libcaffe-cpu1 libchromaprint1 libcrystalhd3 libdc1394-22 libexpat1
  libexpat1-dev libgfortran3 libgme0 libgsm1 libgtkglext1 libhdf5-100
  libjs-jquery libjs-jquery-ui liblapack3 liblmdb0 libmp3lame0 libmpg123-0
  libopencv-core2.4v5 libopencv-highgui2.4-deb0 libopencv-imgproc2.4v5
  libopenjp2-7 libopenmpt0 libpangox-1.0-0 libpython3-dev libpython3.5
  libpython3.5-dev libpython3.5-minimal libpython3.5-stdlib libshine3 libsoxr0
  libssh-gcrypt-4 libswresample2 libswscale4 libsz2 libtbb2 libtwolame0
  libva-drm1 libva-x11-1 libva1 libvdpau1 libx264-148 libx265-110 libxvidcore4
  libzstd1 libzvbi-common libzvbi0 mesa-va-drivers mesa-vdpau-drivers
  python-matplotlib-data python-pexpect python-ptyprocess python-tables-data
  python3-caffe-cpu python3-cycler python3-dateutil python3-decorator
  python3-dev python3-gflags python3-h5py python3-ipython
```

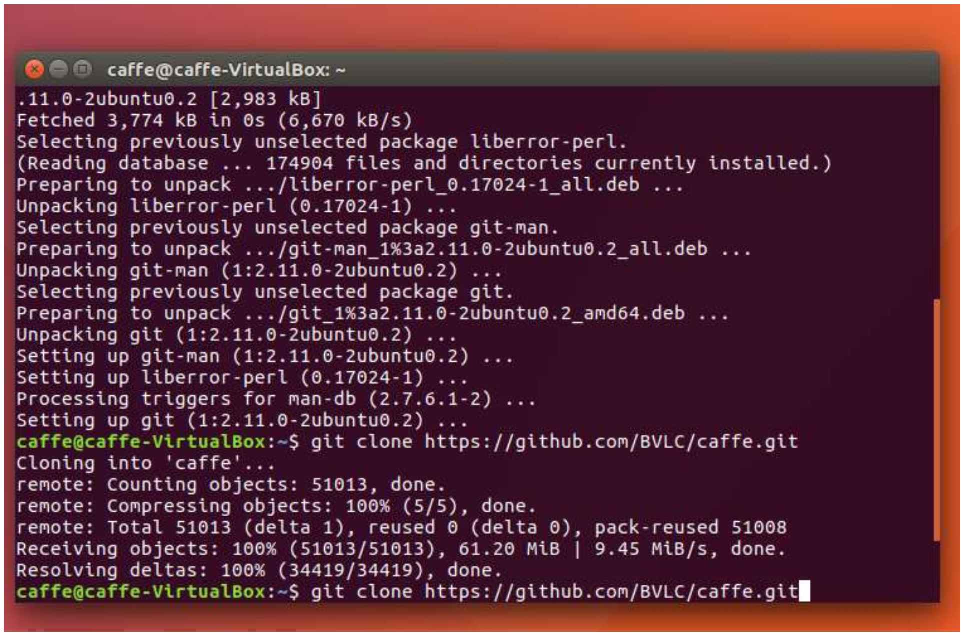
Fig: 1

This command will install the required caffe files on the local pc.

Next Clone the following repositories from GitHub.com I will be uploading Makefiles that will eliminate a lot of manual process later.

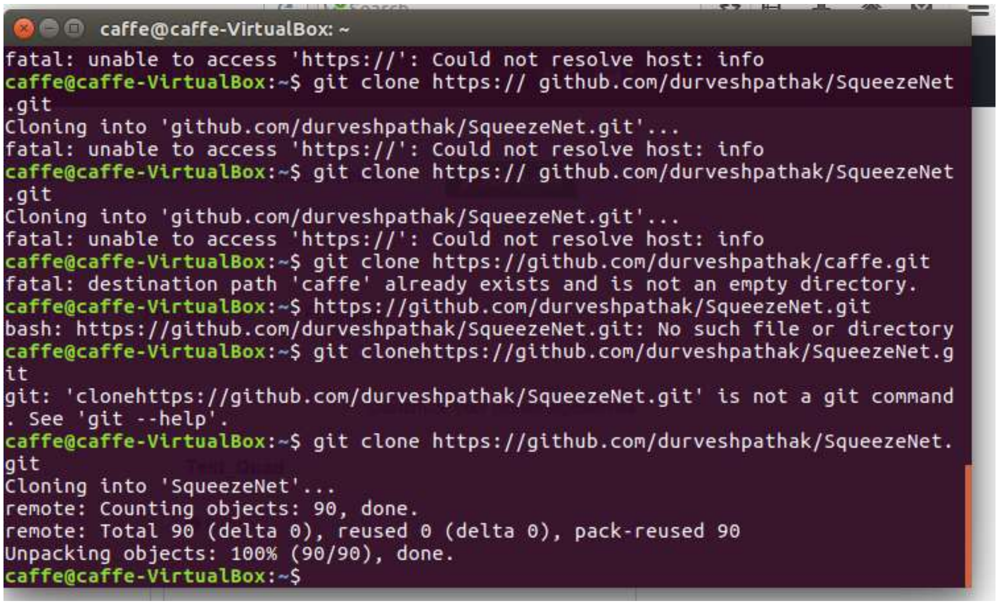
https://github.com/durveshpathak/SqueezeNet.git

https://github.com/durveshpathak/caffe.git (I will keep updating the python files asper our projects)



```
.11.0-2ubuntu0.2 [2,983 kB]
Fetched 3,774 kB in 0s (6,670 kB/s)
Selecting previously unselected package liberror-perl.
(Reading database ... 174904 files and directories currently installed.)
Preparing to unpack .../liberror-perl_0.17024-1_all.deb ...
Unpacking liberror-perl (0.17024-1) ...
Selecting previously unselected package git-man.
Preparing to unpack .../git-man_1%3a2.11.0-2ubuntu0.2_all.deb ...
Unpacking git-man (1:2.11.0-2ubuntu0.2) ...
Selecting previously unselected package git.
Preparing to unpack .../git_1%3a2.11.0-2ubuntu0.2_amd64.deb ...
Unpacking git (1:2.11.0-2ubuntu0.2) ...
Setting up git-man (1:2.11.0-2ubuntu0.2) ...
Setting up liberror-perl (0.17024-1) ...
Processing triggers for man-db (2.7.6.1-2) ...
Setting up git (1:2.11.0-2ubuntu0.2) ...
caffe@caffe-VirtualBox:~$ git clone https://github.com/BVLC/caffe.git
Cloning into 'caffe'...
remote: Counting objects: 51013, done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 51013 (delta 1), reused 0 (delta 0), pack-reused 51008
Receiving objects: 100% (51013/51013), 61.20 MiB | 9.45 MiB/s, done.
Resolving deltas: 100% (34419/34419), done.
caffe@caffe-VirtualBox:~$ git clone https://github.com/BVLC/caffe.git
```

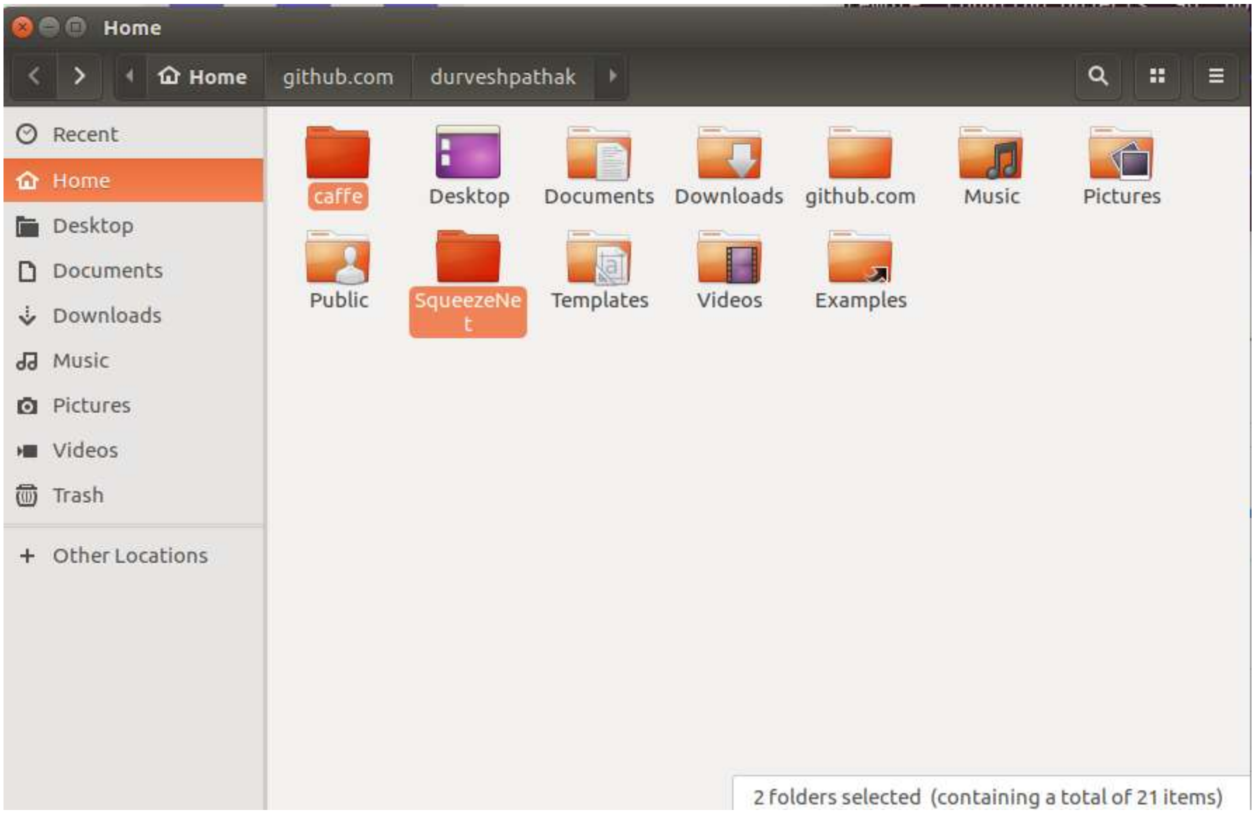
Fig: 2



```
fatal: unable to access 'https://': Could not resolve host: info
caffe@caffe-VirtualBox:~$ git clone https:// github.com/durveshpathak/SqueezeNet
.git
Cloning into 'github.com/durveshpathak/SqueezeNet.git'...
fatal: unable to access 'https://': Could not resolve host: info
caffe@caffe-VirtualBox:~$ git clone https:// github.com/durveshpathak/SqueezeNet
.git
Cloning into 'github.com/durveshpathak/SqueezeNet.git'...
fatal: unable to access 'https://': Could not resolve host: info
caffe@caffe-VirtualBox:~$ git clone https://github.com/durveshpathak/caffe.git
fatal: destination path 'caffe' already exists and is not an empty directory.
caffe@caffe-VirtualBox:~$ https://github.com/durveshpathak/SqueezeNet.git
bash: https://github.com/durveshpathak/SqueezeNet.git: No such file or directory
caffe@caffe-VirtualBox:~$ git clonehttps://github.com/durveshpathak/SqueezeNet.g
it
git: 'clonehttps://github.com/durveshpathak/SqueezeNet.git' is not a git command
. See 'git --help'.
caffe@caffe-VirtualBox:~$ git clone https://github.com/durveshpathak/SqueezeNet.
git
Cloning into 'SqueezeNet'...
remote: Counting objects: 90, done.
remote: Total 90 (delta 0), reused 0 (delta 0), pack-reused 90
Unpacking objects: 100% (90/90), done.
caffe@caffe-VirtualBox:~$
```

Fig: 3

Once you have cloned the above mentioned Repos from GitHub you should be able to see the following folders in your home directory home-→ caffe & Squeezenet



2 folders selected (containing a total of 21 items)

Fig: 4

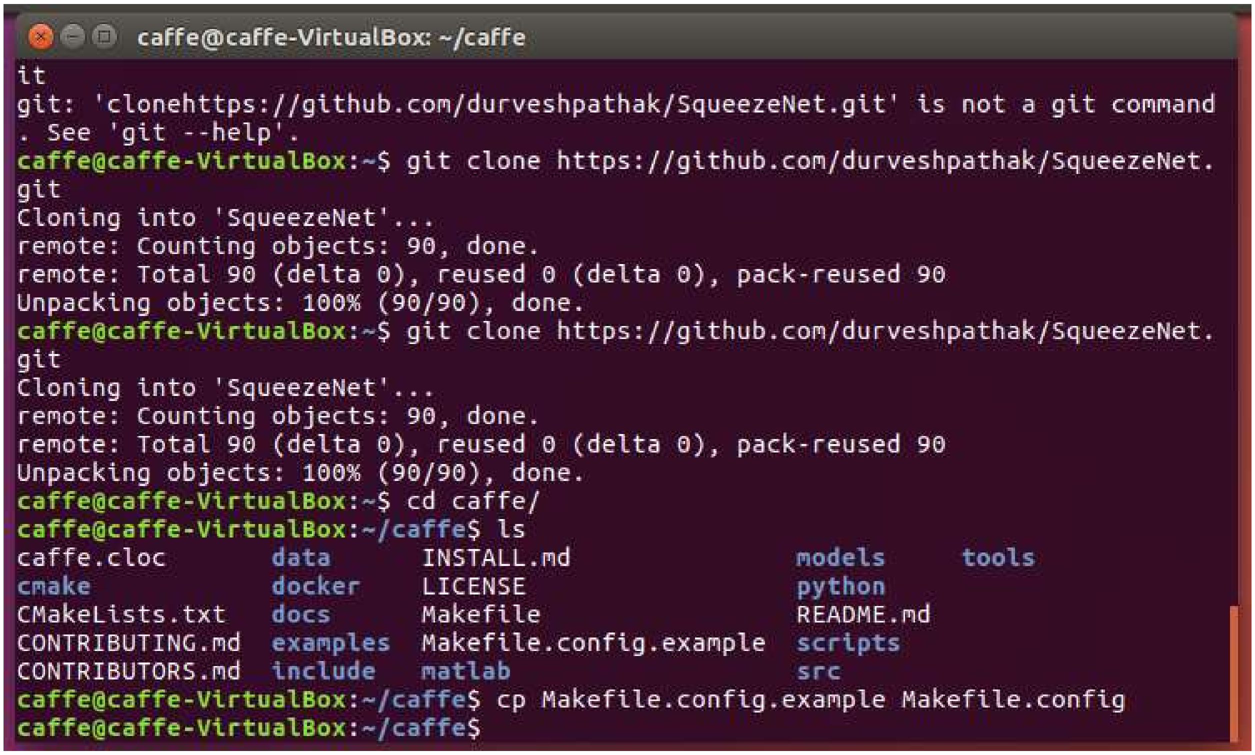
Compilation:

Run the following commands in your terminal.

Change directory to caffe directory

$ cd caffe

$ ls   (this will list all the directories & files.)



```
it
git: 'clonehttps://github.com/durveshpathak/SqueezeNet.git' is not a git command
. See 'git --help'.
caffe@caffe-VirtualBox:~$ git clone https://github.com/durveshpathak/SqueezeNet.
git
Cloning into 'SqueezeNet'...
remote: Counting objects: 90, done.
remote: Total 90 (delta 0), reused 0 (delta 0), pack-reused 90
Unpacking objects: 100% (90/90), done.
caffe@caffe-VirtualBox:~$ git clone https://github.com/durveshpathak/SqueezeNet.
git
Cloning into 'SqueezeNet'...
remote: Counting objects: 90, done.
remote: Total 90 (delta 0), reused 0 (delta 0), pack-reused 90
Unpacking objects: 100% (90/90), done.
caffe@caffe-VirtualBox:~$ cd caffe/
caffe@caffe-VirtualBox:~/caffe$ ls
caffe.cloc        data      INSTALL.md                models     tools
cmake             docker    LICENSE                   python
CMakeLists.txt    docs      Makefile                  README.md
CONTRIBUTING.md   examples  Makefile.config.example   scripts
CONTRIBUTORS.md   include   matlab                    src
caffe@caffe-VirtualBox:~/caffe$ cp Makefile.config.example Makefile.config
caffe@caffe-VirtualBox:~/caffe$
```

Fig: 5

Next we have to generate a MakeFile.config the default make file that comes with the repo cannot be directly used for CPU only modes.

The following command will make a copy of default MakeFile.config.example and save it as Makefile.config ← this is the file that is of utmost importance.

$ cp Makefile.config.example Makefile.config

After you create a makefile.config you might want to edit it based on whether you want to compile caffe for CPU or GPU. Since we are building caffe for CPU version we will change the following things in makefile.config & save the make file

Uncomment CPU_ONLY =1

Uncomment USE_LEVELDB =1

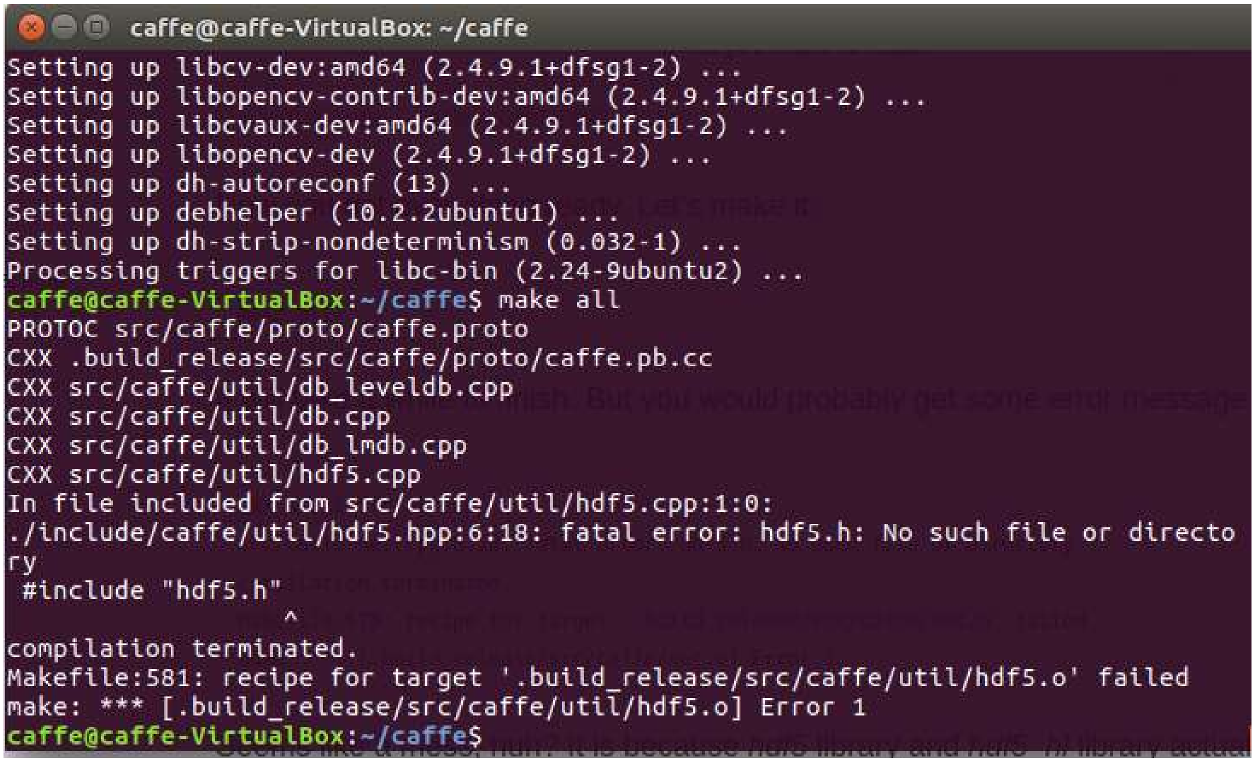
Uncomment USE_OPENCV =1

Uncomment USE_LMDB =1

Make sure when you are doing this you are in /caffe directory. The following command will start building caffe on you machine.

$ make all

You might receive an error!!!



```
Setting up libcv-dev:amd64 (2.4.9.1+dfsg1-2) ...
Setting up libopencv-contrib-dev:amd64 (2.4.9.1+dfsg1-2) ...
Setting up libcvaux-dev:amd64 (2.4.9.1+dfsg1-2) ...
Setting up libopencv-dev (2.4.9.1+dfsg1-2) ...
Setting up dh-autoreconf (13) ...
Setting up debhelper (10.2.2ubuntu1) ...
Setting up dh-strip-nondeterminism (0.032-1) ...
Processing triggers for libc-bin (2.24-9ubuntu2) ...
caffe@caffe-VirtualBox:~/caffe$ make all
PROTOC src/caffe/proto/caffe.proto
CXX .build_release/src/caffe/proto/caffe.pb.cc
CXX src/caffe/util/db_leveldb.cpp
CXX src/caffe/util/db.cpp
CXX src/caffe/util/db_lmdb.cpp
CXX src/caffe/util/hdf5.cpp
In file included from src/caffe/util/hdf5.cpp:1:0:
./include/caffe/util/hdf5.hpp:6:18: fatal error: hdf5.h: No such file or directo
ry
 #include "hdf5.h"
                  ^
compilation terminated.
Makefile:581: recipe for target '.build_release/src/caffe/util/hdf5.o' failed
make: *** [.build_release/src/caffe/util/hdf5.o] Error 1
caffe@caffe-VirtualBox:~/caffe$
```

Fig: 6

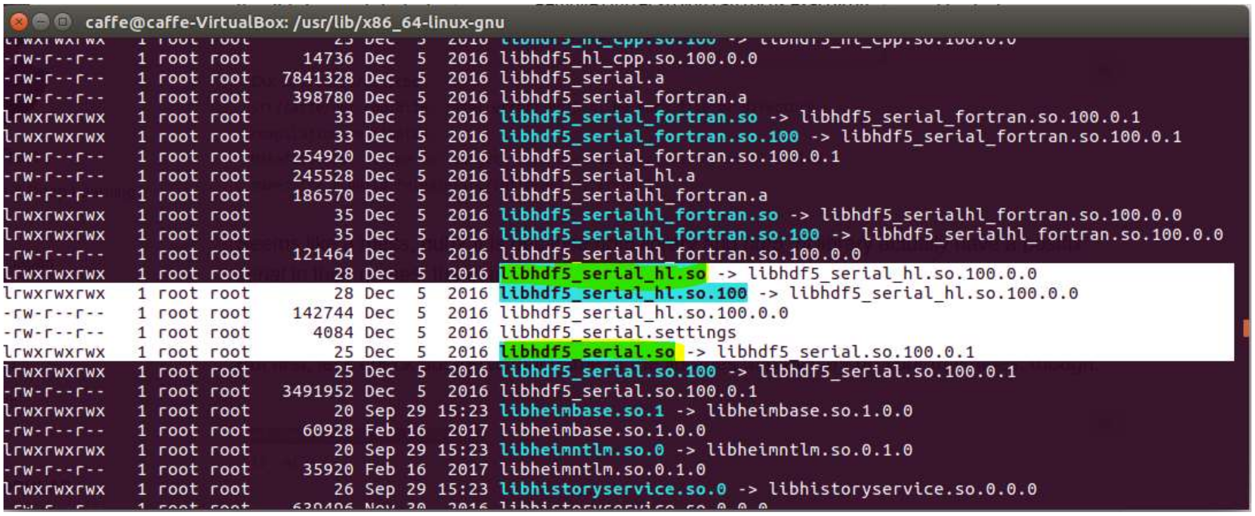
```
CXX src/caffe/net.cpp
src/caffe/net.cpp:8:18: fatal error: hdf5.h: No such file or directory
compilation terminated.
Makefile:575: recipe for target '.build_release/src/caffe/net.o' failed
make: *** [.build_release/src/caffe/net.o] Error 1
```

Seems like a mess, huh? It is because hdf5 library and hdf5_hl library actually have a postfix serial in their names, the compiler cannot find them. To fix this, we just have to make a link to the actual files. Remember, we are not changing their names! But first, let’s check out the actual name of the libraries. It may vary on your machines, though. To fix the issue you need to create a link file.

The following command will help you find the file.

$ cd /usr/lib/x86_64-linux-gnu/

$ ls -al



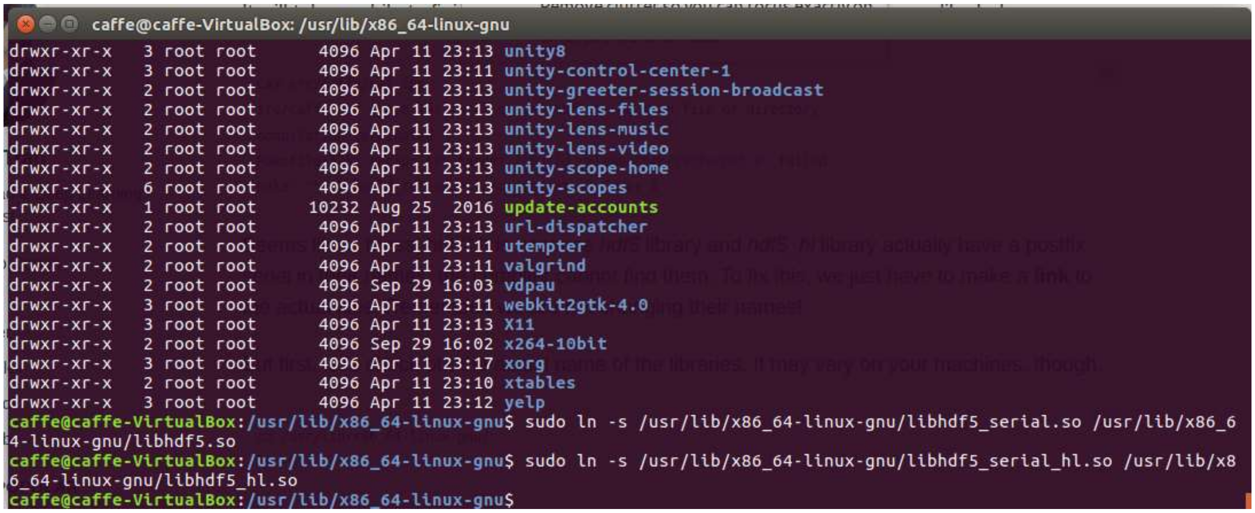
```
lrwxrwxrwx   1 root root       23 Dec  5  2016 libhdf5_hl_cpp.so.100 -> libhdf5_hl_cpp.so.100.0.0
-rw-r--r--   1 root root    14736 Dec  5  2016 libhdf5_hl_cpp.so.100.0.0
-rw-r--r--   1 root root  7841328 Dec  5  2016 libhdf5_serial.a
-rw-r--r--   1 root root   398780 Dec  5  2016 libhdf5_serial_fortran.a
lrwxrwxrwx   1 root root       33 Dec  5  2016 libhdf5_serial_fortran.so -> libhdf5_serial_fortran.so.100.0.1
lrwxrwxrwx   1 root root       33 Dec  5  2016 libhdf5_serial_fortran.so.100 -> libhdf5_serial_fortran.so.100.0.1
-rw-r--r--   1 root root   254920 Dec  5  2016 libhdf5_serial_fortran.so.100.0.1
-rw-r--r--   1 root root   245528 Dec  5  2016 libhdf5_serial_hl.a
-rw-r--r--   1 root root   186570 Dec  5  2016 libhdf5_serialhl_fortran.a
lrwxrwxrwx   1 root root       35 Dec  5  2016 libhdf5_serialhl_fortran.so -> libhdf5_serialhl_fortran.so.100.0.0
lrwxrwxrwx   1 root root       35 Dec  5  2016 libhdf5_serialhl_fortran.so.100 -> libhdf5_serialhl_fortran.so.100.0.0
-rw-r--r--   1 root root   121464 Dec  5  2016 libhdf5_serialhl_fortran.so.100.0.0
lrwxrwxrwx   1 root root       28 Dec  5  2016 libhdf5_serial_hl.so -> libhdf5_serial_hl.so.100.0.0
lrwxrwxrwx   1 root root       28 Dec  5  2016 libhdf5_serial_hl.so.100 -> libhdf5_serial_hl.so.100.0.0
-rw-r--r--   1 root root   142744 Dec  5  2016 libhdf5_serial_hl.so.100.0.0
-rw-r--r--   1 root root     4084 Dec  5  2016 libhdf5_serial.settings
lrwxrwxrwx   1 root root       25 Dec  5  2016 libhdf5_serial.so -> libhdf5_serial.so.100.0.1
lrwxrwxrwx   1 root root       25 Dec  5  2016 libhdf5_serial.so.100 -> libhdf5_serial.so.100.0.1
-rw-r--r--   1 root root  3491952 Dec  5  2016 libhdf5_serial.so.100.0.1
lrwxrwxrwx   1 root root       20 Sep 29 15:23 libheimbase.so.1 -> libheimbase.so.1.0.0
-rw-r--r--   1 root root    60928 Feb 16  2017 libheimbase.so.1.0.0
lrwxrwxrwx   1 root root       20 Sep 29 15:23 libheimntlm.so.0 -> libheimntlm.so.0.1.0
-rw-r--r--   1 root root    35920 Feb 16  2017 libheimntlm.so.0.1.0
lrwxrwxrwx   1 root root       26 Sep 29 15:23 libhistoryservice.so.0 -> libhistoryservice.so.0.0.0
-rw-r--r--   1 root root   639496 Nov 30  2016 libhistoryservice.so.0.0.0
```

Fig: 7

You may find the two files like above. Note again that the version may be different. Just take note the ones you saw. Then we will make a link to them

$ sudo ln -s /usr/lib/x86_64-linux-gnu/libhdf5_serial.so. /usr/lib/x86_64-linux-gnu/libhdf5.so
$sudo ln -s /usr/lib/x86_64-linux-gnu/libhdf5_serial_hl.so.10.0.2 /usr/lib/x86_64-linux-gnu/libhdf5_hl.so

Please note: paths might differ a bit.



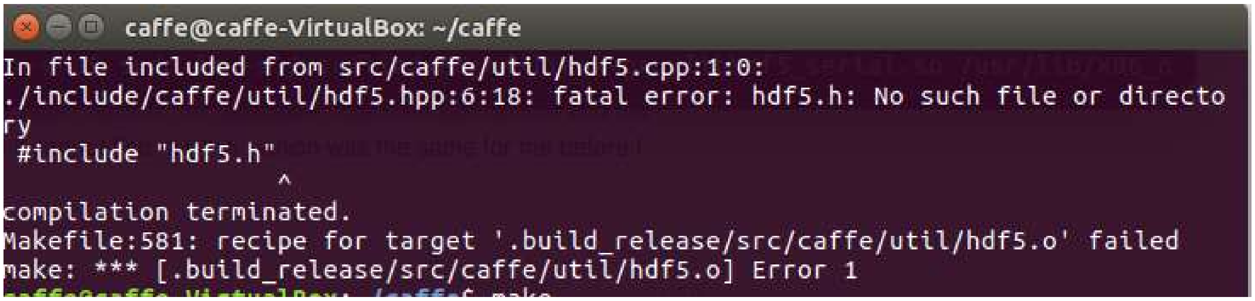
```
drwxr-xr-x   3 root root       4096 Apr 11 23:13 unity8
drwxr-xr-x   3 root root       4096 Apr 11 23:11 unity-control-center-1
drwxr-xr-x   2 root root       4096 Apr 11 23:13 unity-greeter-session-broadcast
drwxr-xr-x   2 root root       4096 Apr 11 23:13 unity-lens-files
drwxr-xr-x   2 root root       4096 Apr 11 23:13 unity-lens-music
drwxr-xr-x   2 root root       4096 Apr 11 23:13 unity-lens-video
drwxr-xr-x   2 root root       4096 Apr 11 23:13 unity-scope-home
drwxr-xr-x   6 root root       4096 Apr 11 23:13 unity-scopes
-rwxr-xr-x   1 root root      10232 Aug 25  2016 update-accounts
drwxr-xr-x   2 root root       4096 Apr 11 23:13 url-dispatcher
drwxr-xr-x   2 root root       4096 Apr 11 23:11 utempter
drwxr-xr-x   2 root root       4096 Apr 11 23:11 valgrind
drwxr-xr-x   2 root root       4096 Sep 29 16:03 vdpau
drwxr-xr-x   3 root root       4096 Apr 11 23:11 webkit2gtk-4.0
drwxr-xr-x   3 root root       4096 Apr 11 23:13 X11
drwxr-xr-x   2 root root       4096 Sep 29 16:02 x264-10bit
drwxr-xr-x   3 root root       4096 Apr 11 23:17 xorg
drwxr-xr-x   2 root root       4096 Apr 11 23:10 xtables
drwxr-xr-x   3 root root       4096 Apr 11 23:12 yelp
caffe@caffe-VirtualBox:/usr/lib/x86_64-linux-gnu$ sudo ln -s /usr/lib/x86_64-linux-gnu/libhdf5_serial.so /usr/lib/x86_6
4-linux-gnu/libhdf5.so
caffe@caffe-VirtualBox:/usr/lib/x86_64-linux-gnu$ sudo ln -s /usr/lib/x86_64-linux-gnu/libhdf5_serial_hl.so /usr/lib/x8
6_64-linux-gnu/libhdf5_hl.so
caffe@caffe-VirtualBox:/usr/lib/x86_64-linux-gnu$
```

Fig: 8

After creating a link file type the following command

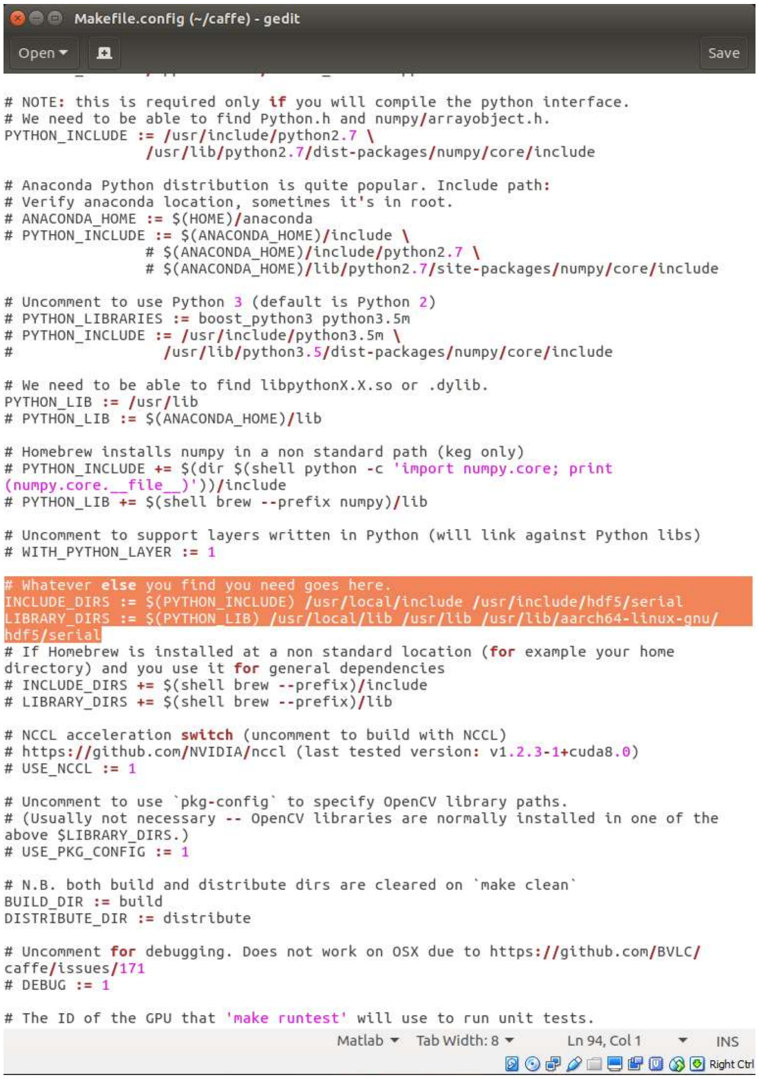
$ make all

If you get the following error, you need to make slight change in Makefile.config



```
In file included from src/caffe/util/hdf5.cpp:1:0:
./include/caffe/util/hdf5.hpp:6:18: fatal error: hdf5.h: No such file or directo
ry
 #include "hdf5.h"
                  ^
compilation terminated.
Makefile:581: recipe for target '.build_release/src/caffe/util/hdf5.o' failed
make: *** [.build_release/src/caffe/util/hdf5.o] Error 1
```

Fig:8



```
# NOTE: this is required only if you will compile the python interface.
# We need to be able to find Python.h and numpy/arrayobject.h.
PYTHON_INCLUDE := /usr/include/python2.7 \
                /usr/lib/python2.7/dist-packages/numpy/core/include

# Anaconda Python distribution is quite popular. Include path:
# Verify anaconda location, sometimes it's in root.
# ANACONDA_HOME := $(HOME)/anaconda
# PYTHON_INCLUDE := $(ANACONDA_HOME)/include \
                # $(ANACONDA_HOME)/include/python2.7 \
                # $(ANACONDA_HOME)/lib/python2.7/site-packages/numpy/core/include

# Uncomment to use Python 3 (default is Python 2)
# PYTHON_LIBRARIES := boost_python3 python3.5m
# PYTHON_INCLUDE := /usr/include/python3.5m \
#                 /usr/lib/python3.5/dist-packages/numpy/core/include

# We need to be able to find libpythonX.X.so or .dylib.
PYTHON_LIB := /usr/lib
# PYTHON_LIB := $(ANACONDA_HOME)/lib

# Homebrew installs numpy in a non standard path (keg only)
# PYTHON_INCLUDE += $(dir $(shell python -c 'import numpy.core; print
(numpy.core.__file__)'))/include
# PYTHON_LIB += $(shell brew --prefix numpy)/lib

# Uncomment to support layers written in Python (will link against Python libs)
# WITH_PYTHON_LAYER := 1

# Whatever else you find you need goes here.
INCLUDE_DIRS := $(PYTHON_INCLUDE) /usr/local/include /usr/include/hdf5/serial
LIBRARY_DIRS := $(PYTHON_LIB) /usr/local/lib /usr/lib /usr/lib/aarch64-linux-gnu/
hdf5/serial
# If Homebrew is installed at a non standard location (for example your home
directory) and you use it for general dependencies
# INCLUDE_DIRS += $(shell brew --prefix)/include
# LIBRARY_DIRS += $(shell brew --prefix)/lib

# NCCL acceleration switch (uncomment to build with NCCL)
# https://github.com/NVIDIA/nccl (last tested version: v1.2.3-1+cuda8.0)
# USE_NCCL := 1

# Uncomment to use `pkg-config` to specify OpenCV library paths.
# (Usually not necessary -- OpenCV libraries are normally installed in one of the
above $LIBRARY_DIRS.)
# USE_PKG_CONFIG := 1

# N.B. both build and distribute dirs are cleared on `make clean`
BUILD_DIR := build
DISTRIBUTE_DIR := distribute

# Uncomment for debugging. Does not work on OSX due to https://github.com/BVLC/
caffe/issues/171
# DEBUG := 1

# The ID of the GPU that 'make runtest' will use to run unit tests.
```

Fig: 9

Add the following line to makefile.config where it’s highlighted

```
INCLUDE_DIRS := $(PYTHON_INCLUDE) /usr/local/include /usr/include/hdf5/serial/ and
LIBRARY_DIRS := $(PYTHON_LIB) /usr/local/lib /usr/lib /usr/lib/x86_64-linux-gnu/hdf5/serial/
```

Try running $ make all again. The caffe should compile.

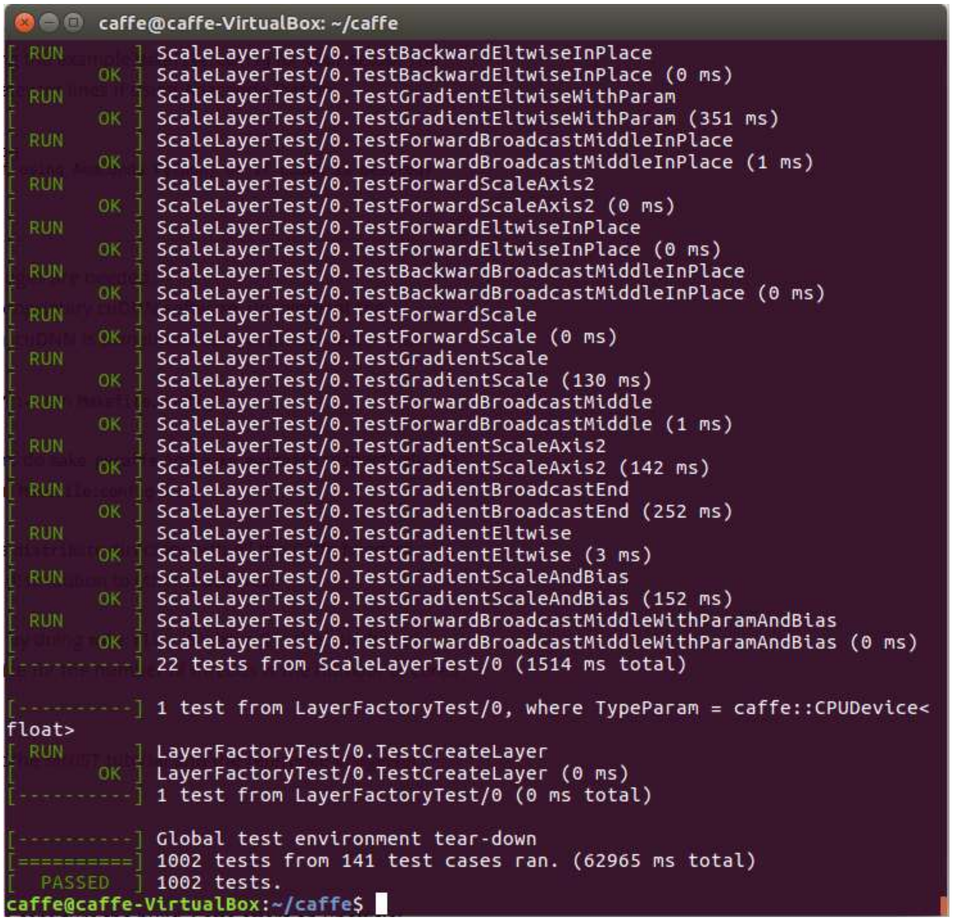
Testing:

To test the caffe first type in the following commands

$ make test

And then

$ make runtest



caffe@caffe-VirtualBox: ~/caffe

```
[ RUN      ] ScaleLayerTest/0.TestBackwardEltwiseInPlace
[       OK ] ScaleLayerTest/0.TestBackwardEltwiseInPlace (0 ms)
[ RUN      ] ScaleLayerTest/0.TestGradientEltwiseWithParam
[       OK ] ScaleLayerTest/0.TestGradientEltwiseWithParam (351 ms)
[ RUN      ] ScaleLayerTest/0.TestForwardBroadcastMiddleInPlace
[       OK ] ScaleLayerTest/0.TestForwardBroadcastMiddleInPlace (1 ms)
[ RUN      ] ScaleLayerTest/0.TestForwardScaleAxis2
[       OK ] ScaleLayerTest/0.TestForwardScaleAxis2 (0 ms)
[ RUN      ] ScaleLayerTest/0.TestForwardEltwiseInPlace
[       OK ] ScaleLayerTest/0.TestForwardEltwiseInPlace (0 ms)
[ RUN      ] ScaleLayerTest/0.TestBackwardBroadcastMiddleInPlace
[       OK ] ScaleLayerTest/0.TestBackwardBroadcastMiddleInPlace (0 ms)
[ RUN      ] ScaleLayerTest/0.TestForwardScale
[       OK ] ScaleLayerTest/0.TestForwardScale (0 ms)
[ RUN      ] ScaleLayerTest/0.TestGradientScale
[       OK ] ScaleLayerTest/0.TestGradientScale (130 ms)
[ RUN      ] ScaleLayerTest/0.TestForwardBroadcastMiddle
[       OK ] ScaleLayerTest/0.TestForwardBroadcastMiddle (1 ms)
[ RUN      ] ScaleLayerTest/0.TestGradientScaleAxis2
[       OK ] ScaleLayerTest/0.TestGradientScaleAxis2 (142 ms)
[ RUN      ] ScaleLayerTest/0.TestGradientBroadcastEnd
[       OK ] ScaleLayerTest/0.TestGradientBroadcastEnd (252 ms)
[ RUN      ] ScaleLayerTest/0.TestGradientEltwise
[       OK ] ScaleLayerTest/0.TestGradientEltwise (3 ms)
[ RUN      ] ScaleLayerTest/0.TestGradientScaleAndBias
[       OK ] ScaleLayerTest/0.TestGradientScaleAndBias (152 ms)
[ RUN      ] ScaleLayerTest/0.TestForwardBroadcastMiddleWithParamAndBias
[       OK ] ScaleLayerTest/0.TestForwardBroadcastMiddleWithParamAndBias (0 ms)
[----------] 22 tests from ScaleLayerTest/0 (1514 ms total)

[----------] 1 test from LayerFactoryTest/0, where TypeParam = caffe::CPUDevice<
float>
[ RUN      ] LayerFactoryTest/0.TestCreateLayer
[       OK ] LayerFactoryTest/0.TestCreateLayer (0 ms)
[----------] 1 test from LayerFactoryTest/0 (0 ms total)

[----------] Global test environment tear-down
[==========] 1002 tests from 141 test cases ran. (62965 ms total)
[  PASSED  ] 1002 tests.
caffe@caffe-VirtualBox:~/caffe$
```

Fig: 10

If you saw something similar, then Congratulations! You have successfully installed Caffe! Now you can get your hands dirty with some real Deep Neural Network projects and become a part of Caffe community!

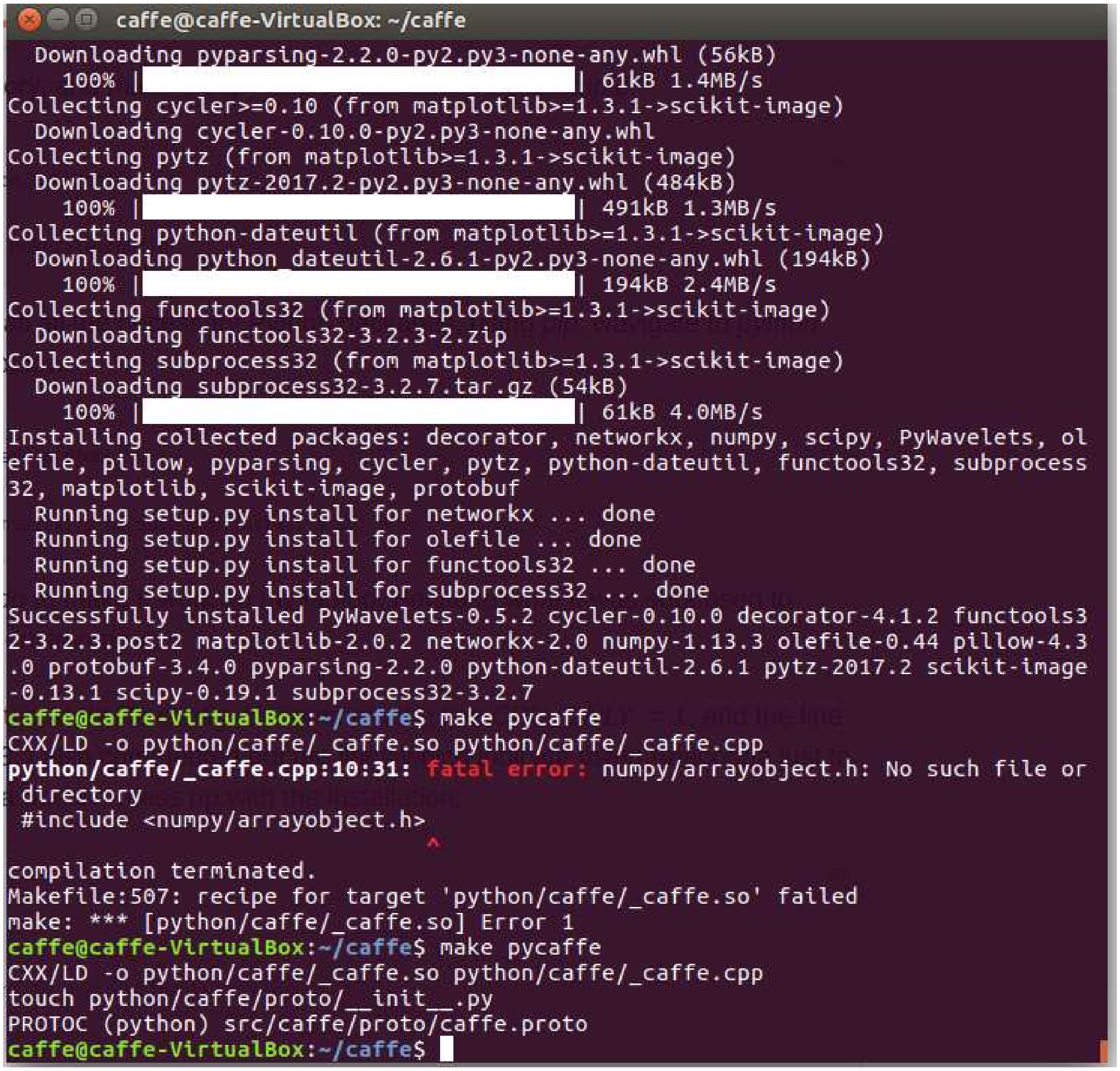
Next step is optional but I highly recommend because we are using Python for our works. We will compile the Python layer so that we can use *caffe* directly in our Python source code

Building Pycaffe:

Next step is to build pycaffe, for that type in the following command

$ make pycaffe

If you observe this error.



```
  Downloading pyparsing-2.2.0-py2.py3-none-any.whl (56kB)
    100% |████████████████████████████████| 61kB 1.4MB/s
Collecting cycler>=0.10 (from matplotlib>=1.3.1->scikit-image)
  Downloading cycler-0.10.0-py2.py3-none-any.whl
Collecting pytz (from matplotlib>=1.3.1->scikit-image)
  Downloading pytz-2017.2-py2.py3-none-any.whl (484kB)
    100% |████████████████████████████████| 491kB 1.3MB/s
Collecting python-dateutil (from matplotlib>=1.3.1->scikit-image)
  Downloading python_dateutil-2.6.1-py2.py3-none-any.whl (194kB)
    100% |████████████████████████████████| 194kB 2.4MB/s
Collecting functools32 (from matplotlib>=1.3.1->scikit-image)
  Downloading functools32-3.2.3-2.zip
Collecting subprocess32 (from matplotlib>=1.3.1->scikit-image)
  Downloading subprocess32-3.2.7.tar.gz (54kB)
    100% |████████████████████████████████| 61kB 4.0MB/s
Installing collected packages: decorator, networkx, numpy, scipy, PyWavelets, ol
efile, pillow, pyparsing, cycler, pytz, python-dateutil, functools32, subprocess
32, matplotlib, scikit-image, protobuf
  Running setup.py install for networkx ... done
  Running setup.py install for olefile ... done
  Running setup.py install for functools32 ... done
  Running setup.py install for subprocess32 ... done
Successfully installed PyWavelets-0.5.2 cycler-0.10.0 decorator-4.1.2 functools3
2-3.2.3.post2 matplotlib-2.0.2 networkx-2.0 numpy-1.13.3 olefile-0.44 pillow-4.3
.0 protobuf-3.4.0 pyparsing-2.2.0 python-dateutil-2.6.1 pytz-2017.2 scikit-image
-0.13.1 scipy-0.19.1 subprocess32-3.2.7
caffe@caffe-VirtualBox:~/caffe$ make pycaffe
CXX/LD -o python/caffe/_caffe.so python/caffe/_caffe.cpp
python/caffe/_caffe.cpp:10:31: fatal error: numpy/arrayobject.h: No such file or
 directory
 #include <numpy/arrayobject.h>
                               ^

compilation terminated.
Makefile:507: recipe for target 'python/caffe/_caffe.so' failed
make: *** [python/caffe/_caffe.so] Error 1
caffe@caffe-VirtualBox:~/caffe$ make pycaffe
CXX/LD -o python/caffe/_caffe.so python/caffe/_caffe.cpp
touch python/caffe/proto/__init__.py
PROTOC (python) src/caffe/proto/caffe.proto
caffe@caffe-VirtualBox:~/caffe$
```

Fig: 11

```
CXX/LD -o python/caffe/_caffe.so python/caffe/_caffe.cpp
python/caffe/_caffe.cpp:10:31: fatal error: numpy/arrayobject.h: No such file or
directory
compilation terminated.
Makefile:501: recipe for target 'python/caffe/_caffe.so' failed
make: *** [python/caffe/_caffe.so] Error 1
```

make the following changes to makefile.config

```
PYTHON_INCLUDE := /usr/include/python2.7 \
/usr/lib/python2.7/dist-packages/numpy/core/include
```

to

```
PYTHON_INCLUDE := /usr/include/python2.7 \
/usr/local/lib/python2.7/dist-packages/numpy/core/include
```

Build pycaffe again!

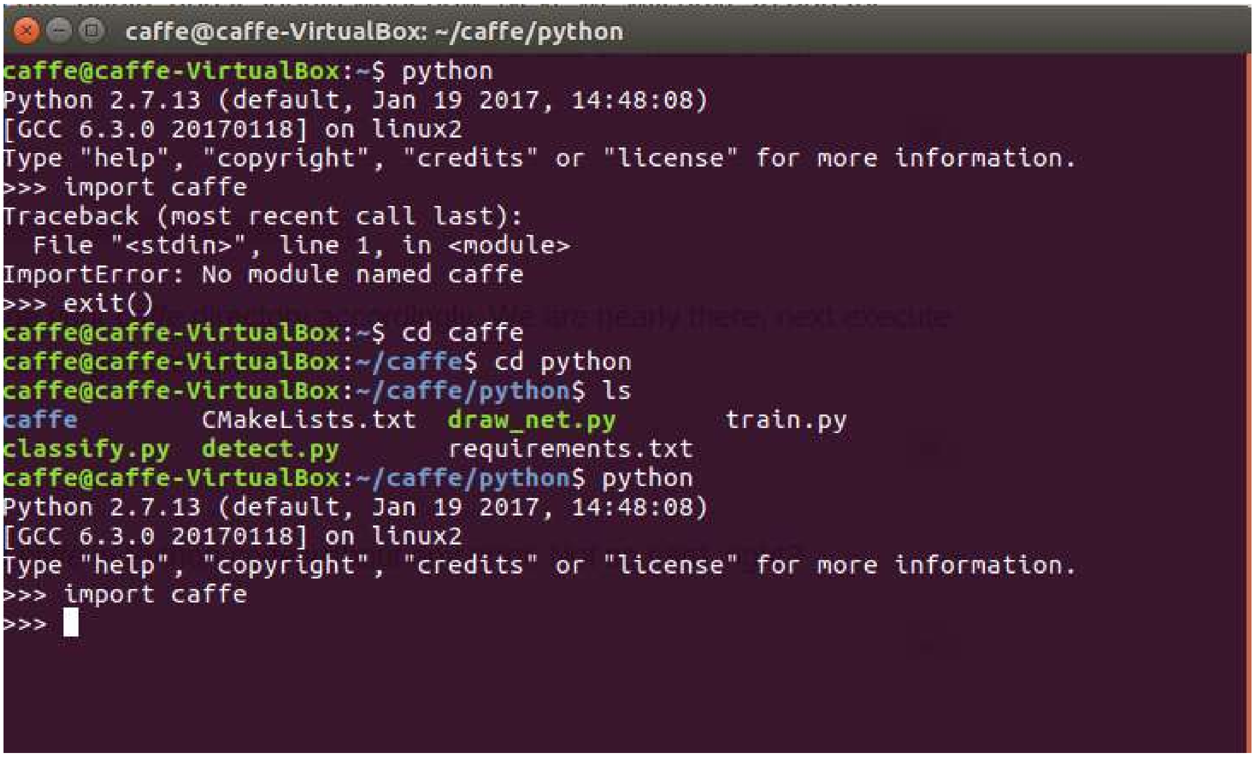
To test the pycaffe move to python directory in caffe and start python environment and type import python this should not give you any error. You can also set this path to your environment variable.

```
$ cd caffe/python
```

```
$ python
```

```
>>  import caffe
```

```
>>
```



caffe@caffe-VirtualBox: ~/caffe/python

```
caffe@caffe-VirtualBox:~$ python
Python 2.7.13 (default, Jan 19 2017, 14:48:08)
[GCC 6.3.0 20170118] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import caffe
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ImportError: No module named caffe
>>> exit()
caffe@caffe-VirtualBox:~$ cd caffe
caffe@caffe-VirtualBox:~/caffe$ cd python
caffe@caffe-VirtualBox:~/caffe/python$ ls
caffe         CMakeLists.txt  draw_net.py        train.py
classify.py   detect.py       requirements.txt
caffe@caffe-VirtualBox:~/caffe/python$ python
Python 2.7.13 (default, Jan 19 2017, 14:48:08)
[GCC 6.3.0 20170118] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import caffe
>>> 
```

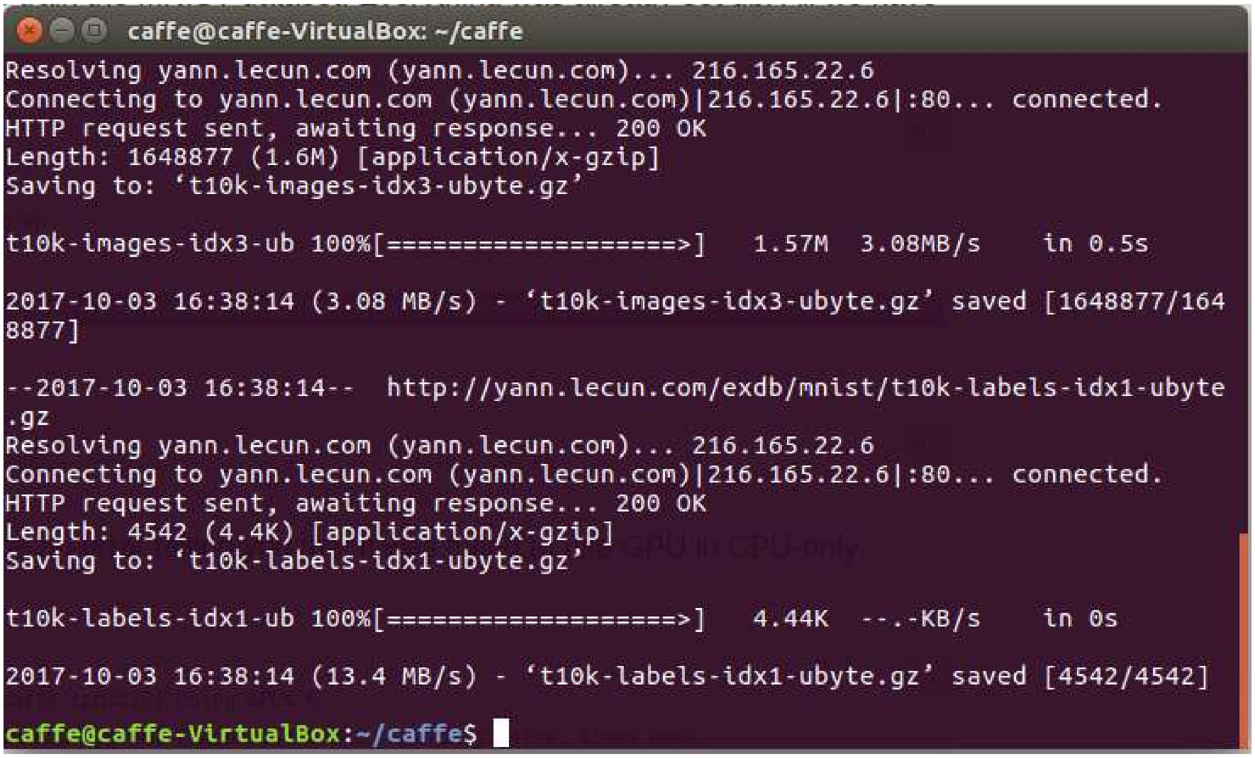
Fig: 12

Running examples:

But we are not done yet. Caffe provides us some examples of the most well-known models. We will use the LeNet model to train the MNIST dataset. Everything was already set up. All we have to do is just make it work:

```
$ cd /Downloads/caffe
```

```
$ ./data/mnist/get_mnist.sh
```



caffe@caffe-VirtualBox: ~/caffe

```
Resolving yann.lecun.com (yann.lecun.com)... 216.165.22.6
Connecting to yann.lecun.com (yann.lecun.com)|216.165.22.6|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1648877 (1.6M) [application/x-gzip]
Saving to: ‘t10k-images-idx3-ubyte.gz’

t10k-images-idx3-ub 100%[===================>]   1.57M  3.08MB/s    in 0.5s

2017-10-03 16:38:14 (3.08 MB/s) - ‘t10k-images-idx3-ubyte.gz’ saved [1648877/164
8877]

--2017-10-03 16:38:14--  http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte
.gz
Resolving yann.lecun.com (yann.lecun.com)... 216.165.22.6
Connecting to yann.lecun.com (yann.lecun.com)|216.165.22.6|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4542 (4.4K) [application/x-gzip]
Saving to: ‘t10k-labels-idx1-ubyte.gz’

t10k-labels-idx1-ub 100%[===================>]   4.44K  --.-KB/s    in 0s

2017-10-03 16:38:14 (13.4 MB/s) - ‘t10k-labels-idx1-ubyte.gz’ saved [4542/4542]

caffe@caffe-VirtualBox:~/caffe$
```

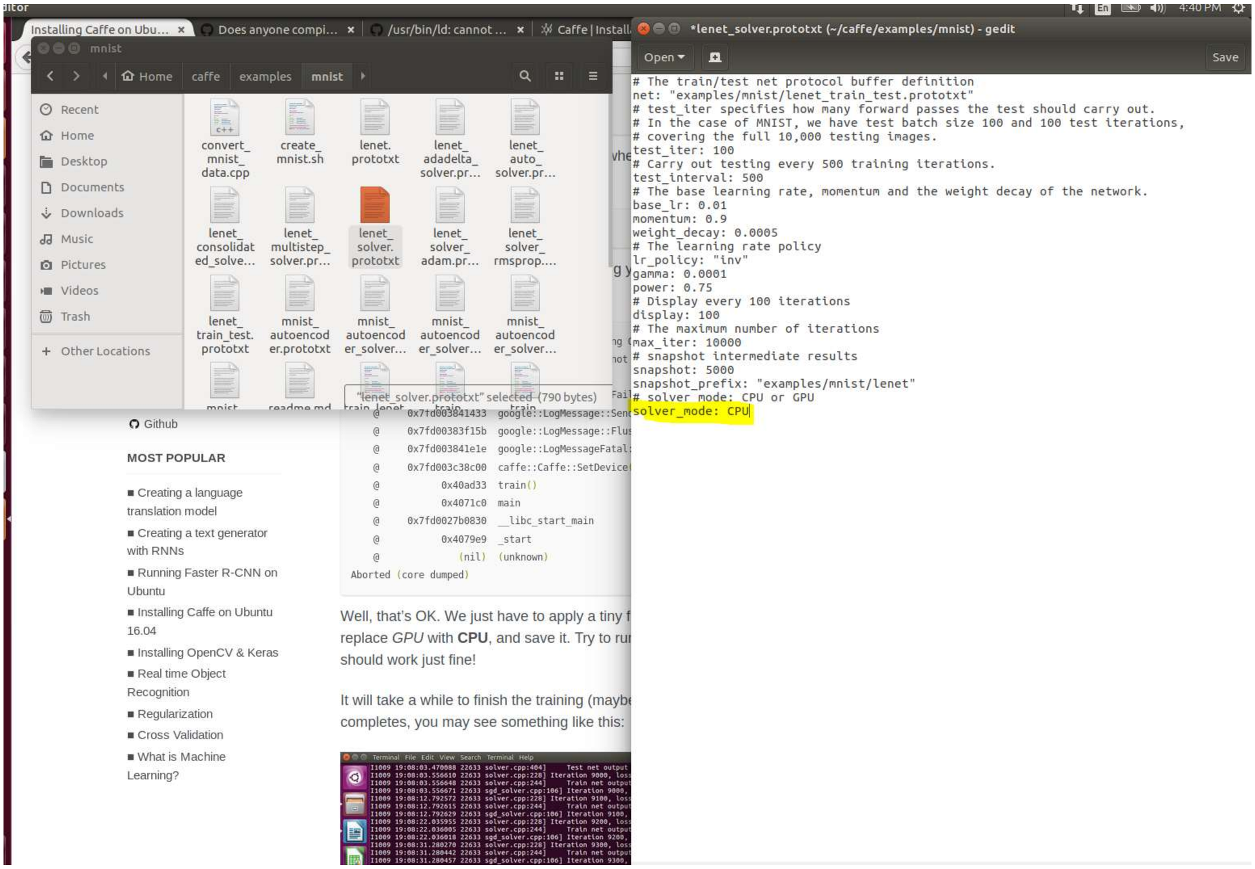
Fig: 13

```
$ ./examples/mnist/create_mnist.sh
```

```
$ ./examples/mnist/train_lenet.sh
```

If the above train command gives you the following error. Well, that's OK. We just have to apply a tiny fix to the file *examples/mnist/lenet_solver.prototxt*, replace *GPU* with **CPU**, and save it. Try to run the command above again, then everything should work just fine!

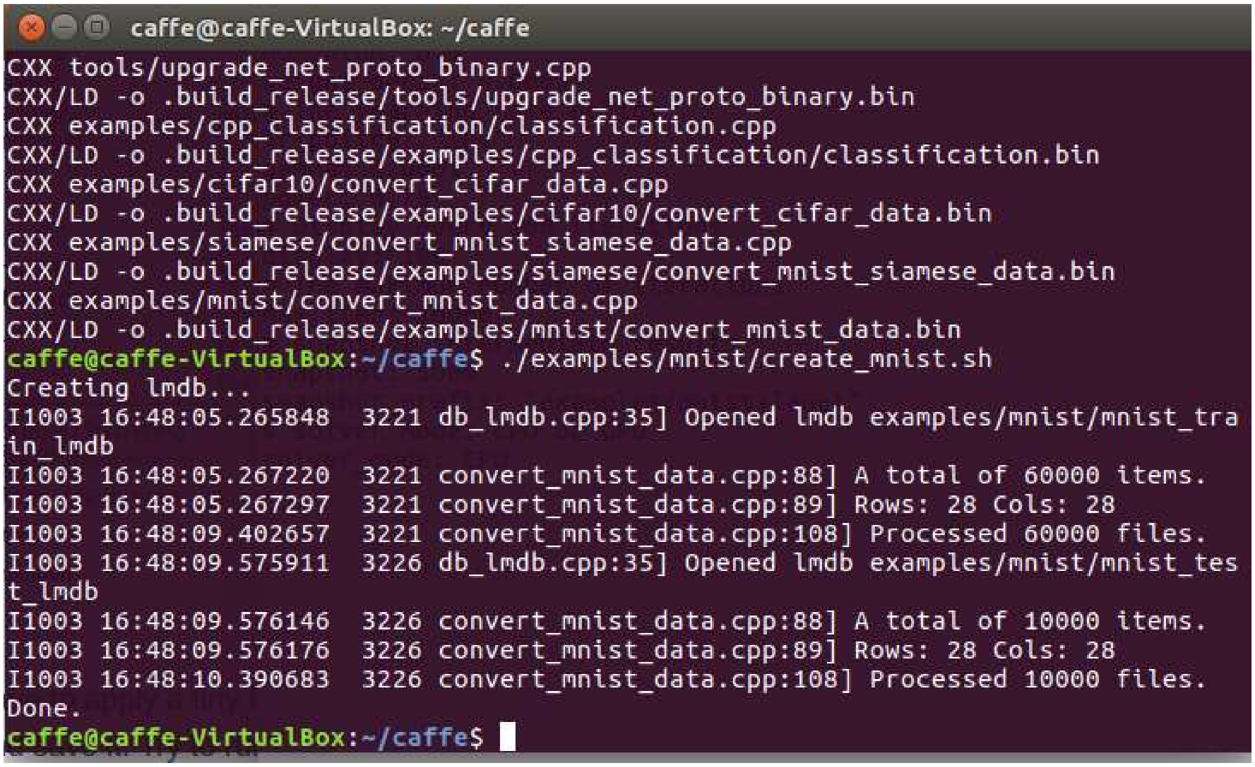
```
I1009 18:51:42.646926 22536 caffe.cpp:217] Using GPUs 0
F1009 18:51:42.647065 22536 common.cpp:66] Cannot use GPU in CPU-only Caffe: check
mode.
*** Check failure stack trace: ***
    @     0x7fd00383f5cd  google::LogMessage::Fail()
    @     0x7fd003841433  google::LogMessage::SendToLog()
    @     0x7fd00383f15b  google::LogMessage::Flush()
    @     0x7fd003841e1e  google::LogMessageFatal::~LogMessageFatal()
    @     0x7fd003c38c00  caffe::Caffe::SetDevice()
    @           0x40ad33  train()
    @           0x4071c0  main
    @     0x7fd0027b0830  __libc_start_main
    @           0x4079e9  _start
    @              (nil)  (unknown)
Aborted (core dumped)
```


Installing Caffe on Ubu…
Does anyone compi…
/usr/bin/ld: cannot …
Caffe | Install
mnist
Home
caffe
examples
mnist
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
convert_mnist_data.cpp
create_mnist.sh
lenet.prototxt
lenet_adadelta_solver.pr…
lenet_auto_solver.pr…
lenet_consolidated_solve…
lenet_multistep_solver.pr…
lenet_solver.prototxt
lenet_solver_adam.pr…
lenet_solver_rmsprop.…
lenet_train_test.prototxt
mnist_autoencoder.prototxt
mnist_autoencoder_solver…
mnist_autoencoder_solver…
mnist_autoencoder_solver…
readme.md
"lenet_solver.prototxt" selected (790 bytes)
Github
MOST POPULAR
Creating a language translation model
Creating a text generator with RNNs
Running Faster R-CNN on Ubuntu
Installing Caffe on Ubuntu 16.04
Installing OpenCV & Keras
Real time Object Recognition
Regularization
Cross Validation
What is Machine Learning?
Terminal File Edit View Search Terminal Help
4:40 PM
Open
Save

*lenet_solver.prototxt (~/caffe/examples/mnist) - gedit

```
# The train/test net protocol buffer definition
net: "examples/mnist/lenet_train_test.prototxt"
# test_iter specifies how many forward passes the test should carry out.
# In the case of MNIST, we have test batch size 100 and 100 test iterations,
# covering the full 10,000 testing images.
test_iter: 100
# Carry out testing every 500 training iterations.
test_interval: 500
# The base learning rate, momentum and the weight decay of the network.
base_lr: 0.01
momentum: 0.9
weight_decay: 0.0005
# The learning rate policy
lr_policy: "inv"
gamma: 0.0001
power: 0.75
# Display every 100 iterations
display: 100
# The maximum number of iterations
max_iter: 10000
# snapshot intermediate results
snapshot: 5000
snapshot_prefix: "examples/mnist/lenet"
# solver_mode: CPU or GPU
solver_mode: CPU
```

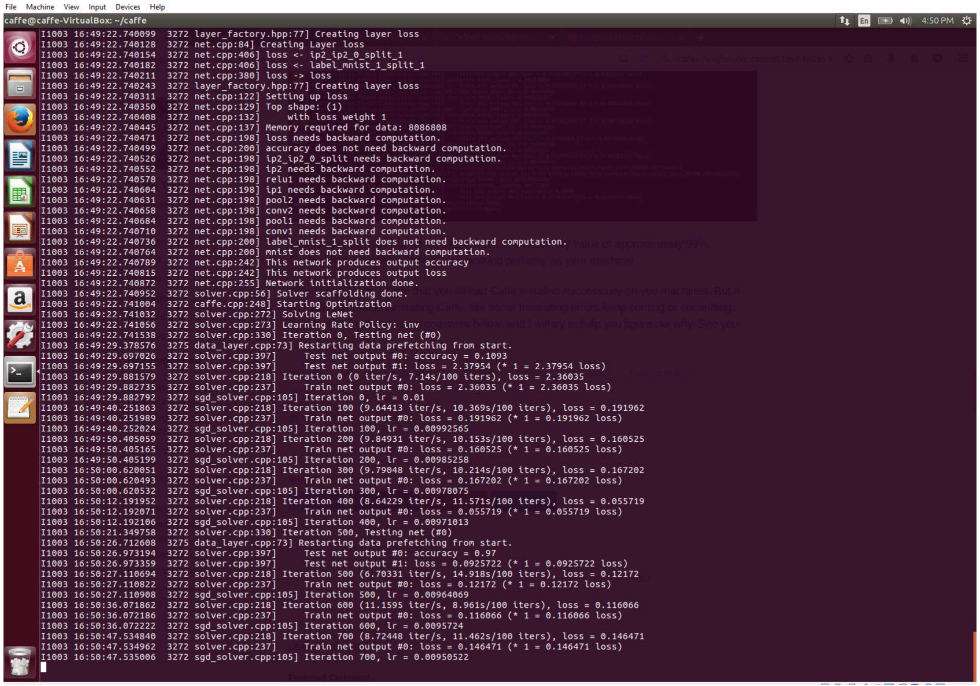
Fig: 14



```
CXX tools/upgrade_net_proto_binary.cpp
CXX/LD -o .build_release/tools/upgrade_net_proto_binary.bin
CXX examples/cpp_classification/classification.cpp
CXX/LD -o .build_release/examples/cpp_classification/classification.bin
CXX examples/cifar10/convert_cifar_data.cpp
CXX/LD -o .build_release/examples/cifar10/convert_cifar_data.bin
CXX examples/siamese/convert_mnist_siamese_data.cpp
CXX/LD -o .build_release/examples/siamese/convert_mnist_siamese_data.bin
CXX examples/mnist/convert_mnist_data.cpp
CXX/LD -o .build_release/examples/mnist/convert_mnist_data.bin
caffe@caffe-VirtualBox:~/caffe$ ./examples/mnist/create_mnist.sh
Creating lmdb...
I1003 16:48:05.265848  3221 db_lmdb.cpp:35] Opened lmdb examples/mnist/mnist_tra
in_lmdb
I1003 16:48:05.267220  3221 convert_mnist_data.cpp:88] A total of 60000 items.
I1003 16:48:05.267297  3221 convert_mnist_data.cpp:89] Rows: 28 Cols: 28
I1003 16:48:09.402657  3221 convert_mnist_data.cpp:108] Processed 60000 files.
I1003 16:48:09.575911  3226 db_lmdb.cpp:35] Opened lmdb examples/mnist/mnist_tes
t_lmdb
I1003 16:48:09.576146  3226 convert_mnist_data.cpp:88] A total of 10000 items.
I1003 16:48:09.576176  3226 convert_mnist_data.cpp:89] Rows: 28 Cols: 28
I1003 16:48:10.390683  3226 convert_mnist_data.cpp:108] Processed 10000 files.
Done.
caffe@caffe-VirtualBox:~/caffe$
```

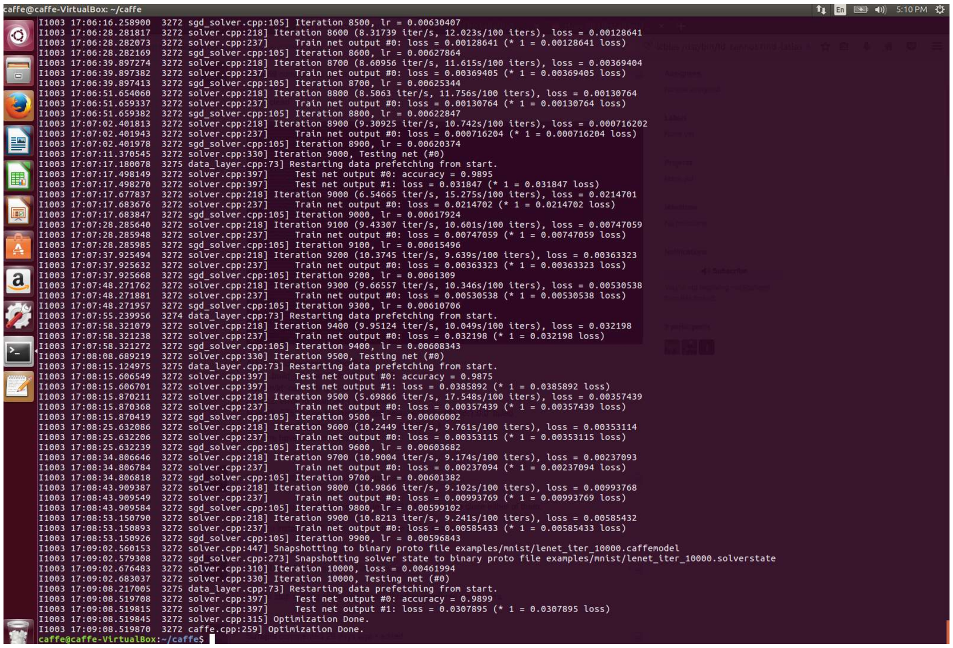
Fig: 15

It will take a while to finish the training (maybe long since we are using CPU). When it completes, you may see something like this:



```
I1003 16:49:22.740099  3272 layer_factory.hpp:77] Creating layer loss
I1003 16:49:22.740128  3272 net.cpp:84] Creating Layer loss
I1003 16:49:22.740154  3272 net.cpp:406] loss <- ip2_ip2_0_split_1
I1003 16:49:22.740182  3272 net.cpp:406] loss <- label_mnist_1_split_1
I1003 16:49:22.740211  3272 net.cpp:380] loss -> loss
I1003 16:49:22.740243  3272 layer_factory.hpp:77] Creating layer loss
I1003 16:49:22.740311  3272 net.cpp:122] Setting up loss
I1003 16:49:22.740350  3272 net.cpp:129] Top shape: (1)
I1003 16:49:22.740408  3272 net.cpp:132]     with loss weight 1
I1003 16:49:22.740445  3272 net.cpp:137] Memory required for data: 8086808
I1003 16:49:22.740471  3272 net.cpp:198] loss needs backward computation.
I1003 16:49:22.740499  3272 net.cpp:200] accuracy does not need backward computation.
I1003 16:49:22.740526  3272 net.cpp:198] ip2_ip2_0_split needs backward computation.
I1003 16:49:22.740552  3272 net.cpp:198] ip2 needs backward computation.
I1003 16:49:22.740578  3272 net.cpp:198] relu1 needs backward computation.
I1003 16:49:22.740604  3272 net.cpp:198] ip1 needs backward computation.
I1003 16:49:22.740631  3272 net.cpp:198] pool2 needs backward computation.
I1003 16:49:22.740658  3272 net.cpp:198] conv2 needs backward computation.
I1003 16:49:22.740684  3272 net.cpp:198] pool1 needs backward computation.
I1003 16:49:22.740710  3272 net.cpp:198] conv1 needs backward computation.
I1003 16:49:22.740736  3272 net.cpp:200] label_mnist_1_split does not need backward computation.
I1003 16:49:22.740764  3272 net.cpp:200] mnist does not need backward computation.
I1003 16:49:22.740789  3272 net.cpp:242] This network produces output accuracy
I1003 16:49:22.740815  3272 net.cpp:242] This network produces output loss
I1003 16:49:22.740872  3272 net.cpp:255] Network initialization done.
I1003 16:49:22.740952  3272 solver.cpp:56] Solver scaffolding done.
I1003 16:49:22.741004  3272 caffe.cpp:248] Starting Optimization
I1003 16:49:22.741032  3272 solver.cpp:272] Solving LeNet
I1003 16:49:22.741056  3272 solver.cpp:273] Learning Rate Policy: inv
I1003 16:49:22.741538  3272 solver.cpp:330] Iteration 0, Testing net (#0)
I1003 16:49:29.378576  3275 data_layer.cpp:73] Restarting data prefetching from start.
I1003 16:49:29.697026  3272 solver.cpp:397]     Test net output #0: accuracy = 0.1093
I1003 16:49:29.697155  3272 solver.cpp:397]     Test net output #1: loss = 2.37954 (* 1 = 2.37954 loss)
I1003 16:49:29.881579  3272 solver.cpp:218] Iteration 0 (0 iter/s, 7.14s/100 iters), loss = 2.36035
I1003 16:49:29.882735  3272 solver.cpp:237]     Train net output #0: loss = 2.36035 (* 1 = 2.36035 loss)
I1003 16:49:29.882792  3272 sgd_solver.cpp:105] Iteration 0, lr = 0.01
I1003 16:49:40.251863  3272 solver.cpp:218] Iteration 100 (9.64413 iter/s, 10.369s/100 iters), loss = 0.191962
I1003 16:49:40.251989  3272 solver.cpp:237]     Train net output #0: loss = 0.191962 (* 1 = 0.191962 loss)
I1003 16:49:40.252024  3272 sgd_solver.cpp:105] Iteration 100, lr = 0.00992565
I1003 16:49:50.405059  3272 solver.cpp:218] Iteration 200 (9.84931 iter/s, 10.153s/100 iters), loss = 0.160525
I1003 16:49:50.405165  3272 solver.cpp:237]     Train net output #0: loss = 0.160525 (* 1 = 0.160525 loss)
I1003 16:49:50.405199  3272 sgd_solver.cpp:105] Iteration 200, lr = 0.00985258
I1003 16:50:00.620051  3272 solver.cpp:218] Iteration 300 (9.79048 iter/s, 10.214s/100 iters), loss = 0.167202
I1003 16:50:00.620493  3272 solver.cpp:237]     Train net output #0: loss = 0.167202 (* 1 = 0.167202 loss)
I1003 16:50:00.620532  3272 sgd_solver.cpp:105] Iteration 300, lr = 0.00978075
I1003 16:50:12.191952  3272 solver.cpp:218] Iteration 400 (8.64229 iter/s, 11.571s/100 iters), loss = 0.055719
I1003 16:50:12.192071  3272 solver.cpp:237]     Train net output #0: loss = 0.055719 (* 1 = 0.055719 loss)
I1003 16:50:12.192106  3272 sgd_solver.cpp:105] Iteration 400, lr = 0.00971013
I1003 16:50:21.349758  3272 solver.cpp:330] Iteration 500, Testing net (#0)
I1003 16:50:26.712608  3275 data_layer.cpp:73] Restarting data prefetching from start.
I1003 16:50:26.973194  3272 solver.cpp:397]     Test net output #0: accuracy = 0.97
I1003 16:50:26.973359  3272 solver.cpp:397]     Test net output #1: loss = 0.0925722 (* 1 = 0.0925722 loss)
I1003 16:50:27.110694  3272 solver.cpp:218] Iteration 500 (6.70331 iter/s, 14.918s/100 iters), loss = 0.12172
I1003 16:50:27.110822  3272 solver.cpp:237]     Train net output #0: loss = 0.12172 (* 1 = 0.12172 loss)
I1003 16:50:27.110908  3272 sgd_solver.cpp:105] Iteration 500, lr = 0.00964069
I1003 16:50:36.071862  3272 solver.cpp:218] Iteration 600 (11.1595 iter/s, 8.961s/100 iters), loss = 0.116066
I1003 16:50:36.072186  3272 solver.cpp:237]     Train net output #0: loss = 0.116066 (* 1 = 0.116066 loss)
I1003 16:50:36.072222  3272 sgd_solver.cpp:105] Iteration 600, lr = 0.0095724
I1003 16:50:47.534840  3272 solver.cpp:218] Iteration 700 (8.72448 iter/s, 11.462s/100 iters), loss = 0.146471
I1003 16:50:47.534962  3272 solver.cpp:237]     Train net output #0: loss = 0.146471 (* 1 = 0.146471 loss)
I1003 16:50:47.535006  3272 sgd_solver.cpp:105] Iteration 700, lr = 0.00950522
```

Fig: 16



```
I1003 17:06:16.258900  3272 sgd_solver.cpp:105] Iteration 8500, lr = 0.00630407
I1003 17:06:28.281817  3272 solver.cpp:218] Iteration 8600 (8.31739 iter/s, 12.023s/100 iters), loss = 0.00128641
I1003 17:06:28.282073  3272 solver.cpp:237]     Train net output #0: loss = 0.00128641 (* 1 = 0.00128641 loss)
I1003 17:06:28.282169  3272 sgd_solver.cpp:105] Iteration 8600, lr = 0.00627864
I1003 17:06:39.897274  3272 solver.cpp:218] Iteration 8700 (8.60956 iter/s, 11.615s/100 iters), loss = 0.00369404
I1003 17:06:39.897382  3272 solver.cpp:237]     Train net output #0: loss = 0.00369405 (* 1 = 0.00369405 loss)
I1003 17:06:39.897413  3272 sgd_solver.cpp:105] Iteration 8700, lr = 0.00625344
I1003 17:06:51.654060  3272 solver.cpp:218] Iteration 8800 (8.5063 iter/s, 11.756s/100 iters), loss = 0.00130764
I1003 17:06:51.659337  3272 solver.cpp:237]     Train net output #0: loss = 0.00130764 (* 1 = 0.00130764 loss)
I1003 17:06:51.659382  3272 sgd_solver.cpp:105] Iteration 8800, lr = 0.00622847
I1003 17:07:02.401813  3272 solver.cpp:218] Iteration 8900 (9.30925 iter/s, 10.742s/100 iters), loss = 0.000716202
I1003 17:07:02.401943  3272 solver.cpp:237]     Train net output #0: loss = 0.000716204 (* 1 = 0.000716204 loss)
I1003 17:07:02.401978  3272 sgd_solver.cpp:105] Iteration 8900, lr = 0.00620374
I1003 17:07:11.370545  3272 solver.cpp:330] Iteration 9000, Testing net (#0)
I1003 17:07:17.180078  3275 data_layer.cpp:73] Restarting data prefetching from start.
I1003 17:07:17.498149  3272 solver.cpp:397]     Test net output #0: accuracy = 0.9895
I1003 17:07:17.498270  3272 solver.cpp:397]     Test net output #1: loss = 0.031847 (* 1 = 0.031847 loss)
I1003 17:07:17.677837  3272 solver.cpp:218] Iteration 9000 (6.54665 iter/s, 15.275s/100 iters), loss = 0.0214701
I1003 17:07:17.683676  3272 solver.cpp:237]     Train net output #0: loss = 0.0214702 (* 1 = 0.0214702 loss)
I1003 17:07:17.683847  3272 sgd_solver.cpp:105] Iteration 9000, lr = 0.00617924
I1003 17:07:28.285640  3272 solver.cpp:218] Iteration 9100 (9.43307 iter/s, 10.601s/100 iters), loss = 0.00747059
I1003 17:07:28.285948  3272 solver.cpp:237]     Train net output #0: loss = 0.00747059 (* 1 = 0.00747059 loss)
I1003 17:07:28.285985  3272 sgd_solver.cpp:105] Iteration 9100, lr = 0.00615496
I1003 17:07:37.925494  3272 solver.cpp:218] Iteration 9200 (10.3745 iter/s, 9.639s/100 iters), loss = 0.00363323
I1003 17:07:37.925632  3272 solver.cpp:237]     Train net output #0: loss = 0.00363323 (* 1 = 0.00363323 loss)
I1003 17:07:37.925668  3272 sgd_solver.cpp:105] Iteration 9200, lr = 0.0061309
I1003 17:07:48.271762  3272 solver.cpp:218] Iteration 9300 (9.66557 iter/s, 10.346s/100 iters), loss = 0.00530538
I1003 17:07:48.271881  3272 solver.cpp:237]     Train net output #0: loss = 0.00530538 (* 1 = 0.00530538 loss)
I1003 17:07:48.271957  3272 sgd_solver.cpp:105] Iteration 9300, lr = 0.00610706
I1003 17:07:55.239956  3274 data_layer.cpp:73] Restarting data prefetching from start.
I1003 17:07:58.321079  3272 solver.cpp:218] Iteration 9400 (9.95124 iter/s, 10.049s/100 iters), loss = 0.032198
I1003 17:07:58.321238  3272 solver.cpp:237]     Train net output #0: loss = 0.032198 (* 1 = 0.032198 loss)
I1003 17:07:58.321272  3272 sgd_solver.cpp:105] Iteration 9400, lr = 0.00608343
I1003 17:08:08.689219  3272 solver.cpp:330] Iteration 9500, Testing net (#0)
I1003 17:08:15.124975  3275 data_layer.cpp:73] Restarting data prefetching from start.
I1003 17:08:15.606549  3272 solver.cpp:397]     Test net output #0: accuracy = 0.9875
I1003 17:08:15.606701  3272 solver.cpp:397]     Test net output #1: loss = 0.0385892 (* 1 = 0.0385892 loss)
I1003 17:08:15.870211  3272 solver.cpp:218] Iteration 9500 (5.69866 iter/s, 17.548s/100 iters), loss = 0.00357439
I1003 17:08:15.870368  3272 solver.cpp:237]     Train net output #0: loss = 0.00357439 (* 1 = 0.00357439 loss)
I1003 17:08:15.870419  3272 sgd_solver.cpp:105] Iteration 9500, lr = 0.00606002
I1003 17:08:25.632086  3272 solver.cpp:218] Iteration 9600 (10.2449 iter/s, 9.761s/100 iters), loss = 0.00353114
I1003 17:08:25.632206  3272 solver.cpp:237]     Train net output #0: loss = 0.00353115 (* 1 = 0.00353115 loss)
I1003 17:08:25.632239  3272 sgd_solver.cpp:105] Iteration 9600, lr = 0.00603682
I1003 17:08:34.806646  3272 solver.cpp:218] Iteration 9700 (10.9004 iter/s, 9.174s/100 iters), loss = 0.00237093
I1003 17:08:34.806784  3272 solver.cpp:237]     Train net output #0: loss = 0.00237094 (* 1 = 0.00237094 loss)
I1003 17:08:34.806818  3272 sgd_solver.cpp:105] Iteration 9700, lr = 0.00601382
I1003 17:08:43.909387  3272 solver.cpp:218] Iteration 9800 (10.9866 iter/s, 9.102s/100 iters), loss = 0.00993768
I1003 17:08:43.909549  3272 solver.cpp:237]     Train net output #0: loss = 0.00993769 (* 1 = 0.00993769 loss)
I1003 17:08:43.909584  3272 sgd_solver.cpp:105] Iteration 9800, lr = 0.00599102
I1003 17:08:53.150790  3272 solver.cpp:218] Iteration 9900 (10.8213 iter/s, 9.241s/100 iters), loss = 0.00585432
I1003 17:08:53.150893  3272 solver.cpp:237]     Train net output #0: loss = 0.00585433 (* 1 = 0.00585433 loss)
I1003 17:08:53.150926  3272 sgd_solver.cpp:105] Iteration 9900, lr = 0.00596843
I1003 17:09:02.560153  3272 solver.cpp:447] Snapshotting to binary proto file examples/mnist/lenet_iter_10000.caffemodel
I1003 17:09:02.579308  3272 sgd_solver.cpp:273] Snapshotting solver state to binary proto file examples/mnist/lenet_iter_10000.solverstate
I1003 17:09:02.676483  3272 solver.cpp:310] Iteration 10000, loss = 0.00461994
I1003 17:09:02.683037  3272 solver.cpp:330] Iteration 10000, Testing net (#0)
I1003 17:09:08.217005  3275 data_layer.cpp:73] Restarting data prefetching from start.
I1003 17:09:08.519708  3272 solver.cpp:397]     Test net output #0: accuracy = 0.9899
I1003 17:09:08.519815  3272 solver.cpp:397]     Test net output #1: loss = 0.0307895 (* 1 = 0.0307895 loss)
I1003 17:09:08.519845  3272 solver.cpp:315] Optimization Done.
I1003 17:09:08.519870  3272 caffe.cpp:259] Optimization Done.
caffe@caffe-VirtualBox:~/caffe$
```

Fig: 17

Network is trained!!

So what have we trained!!!

I never explained what we trained, but since we have trained it let dig a little deeper to understand, what we have trained.

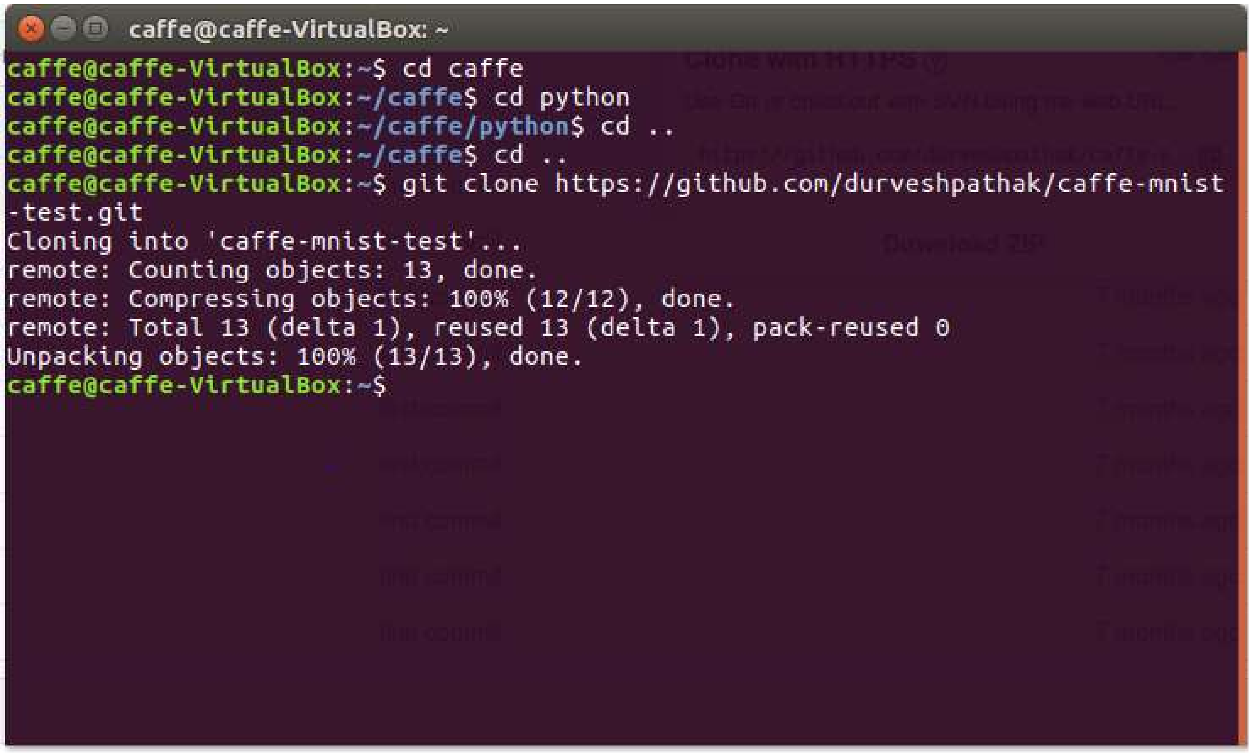
The simple answer is, we have trained Lenet network it’s a type of Convolutional Neural Network that is popularly used for classification secondly we have trained the network with mnist data base which contains 60,000 image of digits 0,1,2….9.

So, since we have trained the network its time to run some example that we can visualize the output.

We will use a simple python script

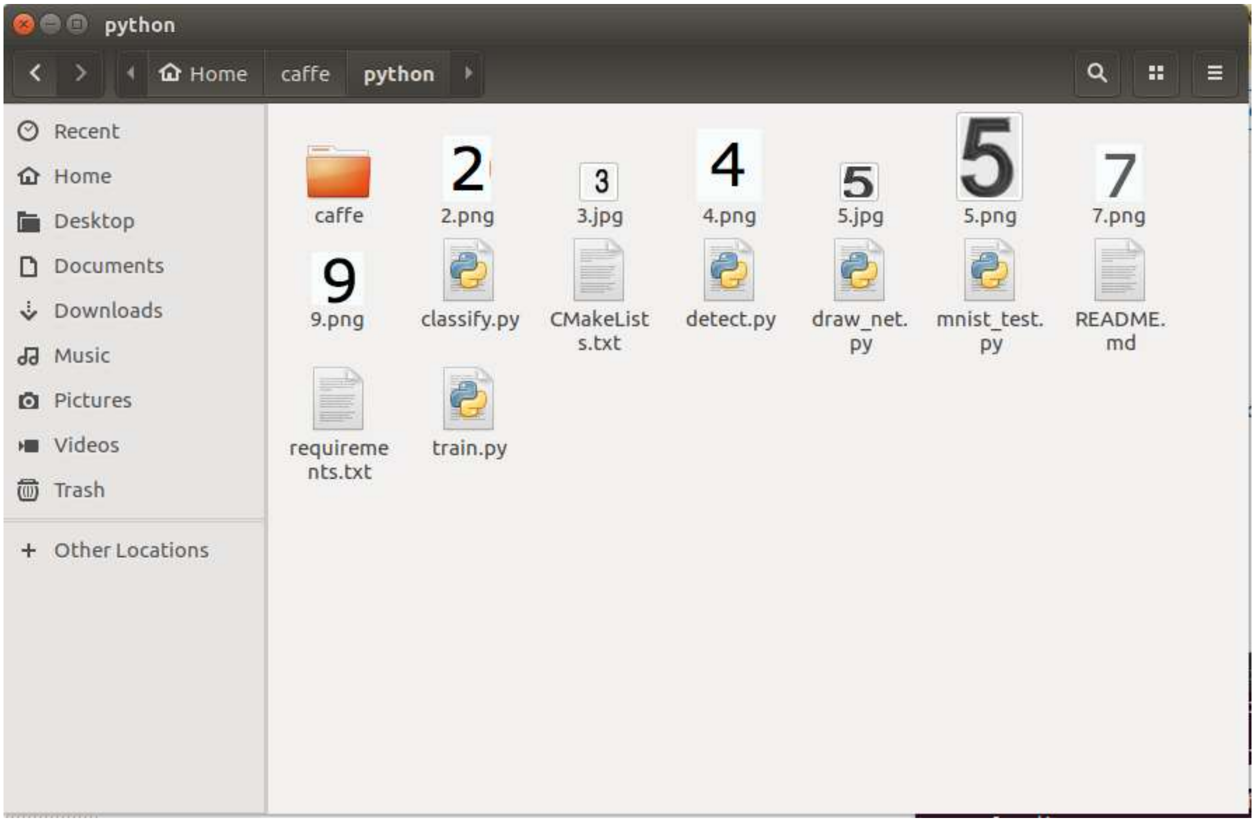
Step 1: open a text editor.

Step 2: clone the following repo from github https://github.com/durveshpathak/caffe-mnist-test.git this will download a python code and image files.



```
caffe@caffe-VirtualBox:~$ cd caffe
caffe@caffe-VirtualBox:~/caffe$ cd python
caffe@caffe-VirtualBox:~/caffe/python$ cd ..
caffe@caffe-VirtualBox:~/caffe$ cd ..
caffe@caffe-VirtualBox:~$ git clone https://github.com/durveshpathak/caffe-mnist
-test.git
Cloning into 'caffe-mnist-test'...
remote: Counting objects: 13, done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 13 (delta 1), reused 13 (delta 1), pack-reused 0
Unpacking objects: 100% (13/13), done.
caffe@caffe-VirtualBox:~$
```

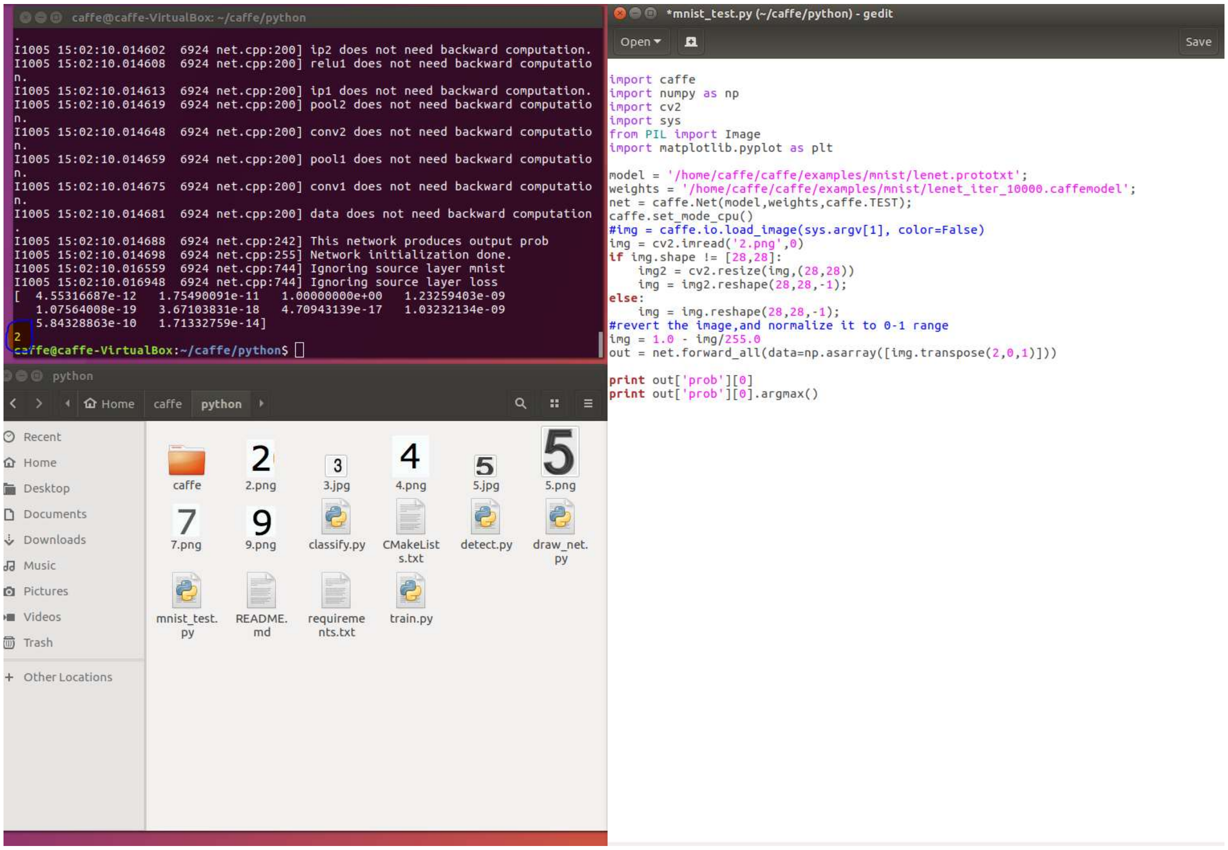
Step 3: Paste the files inside the downloaded folder in caffe/python directory



After pasting the files type in the following commands

```
$ python mnist_test.py
```

Conclusion : We can see that if we provide the png or jpg file for a specific digit the Lenet recognizes the digit and provides the prediction. Please make sure that you provide the right extension of the file. And Image and python file should be in same folder.

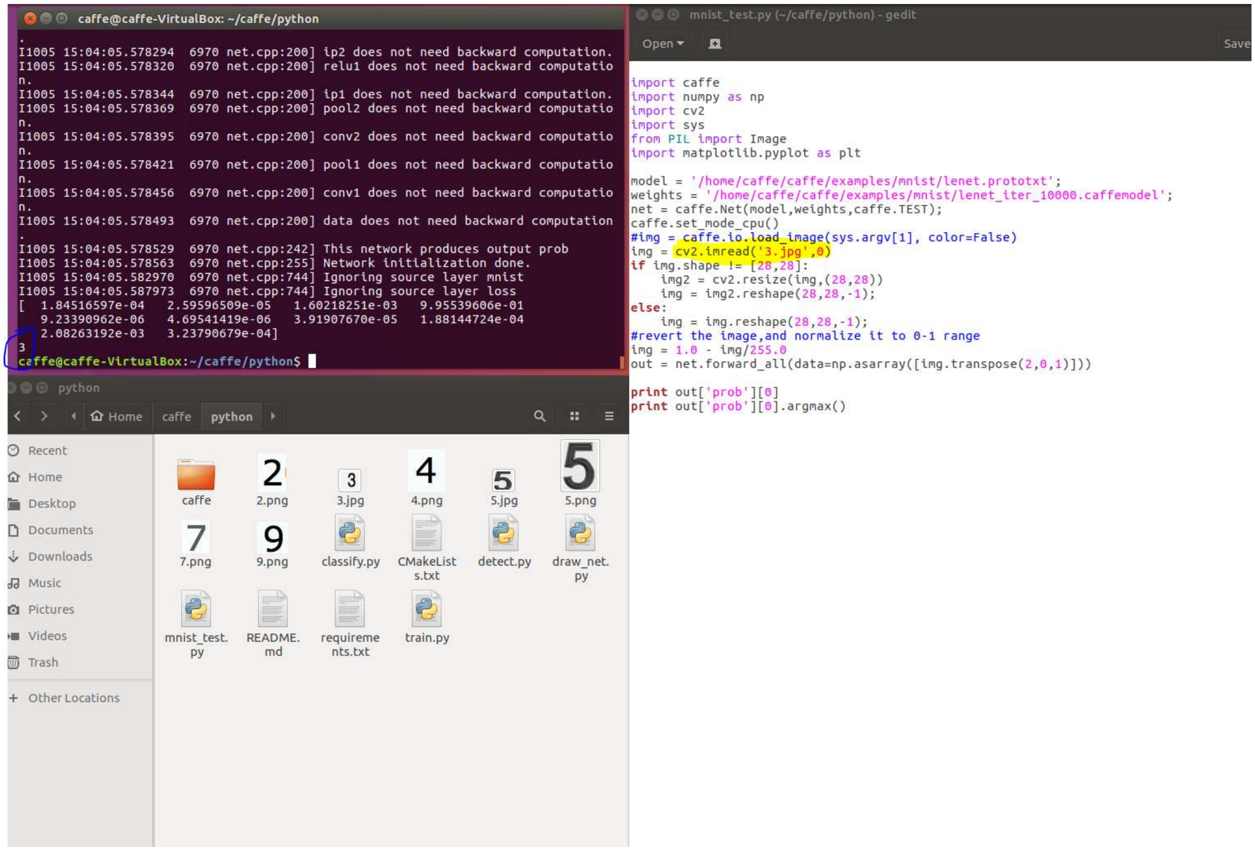


```
.
I1005 15:02:10.014602  6924 net.cpp:200] ip2 does not need backward computation.
I1005 15:02:10.014608  6924 net.cpp:200] relu1 does not need backward computatio
n.
I1005 15:02:10.014613  6924 net.cpp:200] ip1 does not need backward computation.
I1005 15:02:10.014619  6924 net.cpp:200] pool2 does not need backward computatio
n.
I1005 15:02:10.014648  6924 net.cpp:200] conv2 does not need backward computatio
n.
I1005 15:02:10.014659  6924 net.cpp:200] pool1 does not need backward computatio
n.
I1005 15:02:10.014675  6924 net.cpp:200] conv1 does not need backward computatio
n.
I1005 15:02:10.014681  6924 net.cpp:200] data does not need backward computation
.
I1005 15:02:10.014688  6924 net.cpp:242] This network produces output prob
I1005 15:02:10.014698  6924 net.cpp:255] Network initialization done.
I1005 15:02:10.016559  6924 net.cpp:744] Ignoring source layer mnist
I1005 15:02:10.016948  6924 net.cpp:744] Ignoring source layer loss
[  4.55316687e-12   1.75490091e-11   1.00000000e+00   1.23259403e-09
   1.07564008e-19   3.67103831e-18   4.70943139e-17   1.03232134e-09
   5.84328863e-10   1.71332759e-14]
2
caffe@caffe-VirtualBox:~/caffe/python$
```

```python
import caffe
import numpy as np
import cv2
import sys
from PIL import Image
import matplotlib.pyplot as plt

model = '/home/caffe/caffe/examples/mnist/lenet.prototxt';
weights = '/home/caffe/caffe/examples/mnist/lenet_iter_10000.caffemodel';
net = caffe.Net(model,weights,caffe.TEST);
caffe.set_mode_cpu()
#img = caffe.io.load_image(sys.argv[1], color=False)
img = cv2.imread('2.png',0)
if img.shape != [28,28]:
    img2 = cv2.resize(img,(28,28))
    img = img2.reshape(28,28,-1);
else:
    img = img.reshape(28,28,-1);
#revert the image,and normalize it to 0-1 range
img = 1.0 - img/255.0
out = net.forward_all(data=np.asarray([img.transpose(2,0,1)]))

print out['prob'][0]
print out['prob'][0].argmax()
```



```
.
I1005 15:04:05.578294  6970 net.cpp:200] ip2 does not need backward computation.
I1005 15:04:05.578320  6970 net.cpp:200] relu1 does not need backward computatio
n.
I1005 15:04:05.578344  6970 net.cpp:200] ip1 does not need backward computation.
I1005 15:04:05.578369  6970 net.cpp:200] pool2 does not need backward computatio
n.
I1005 15:04:05.578395  6970 net.cpp:200] conv2 does not need backward computatio
n.
I1005 15:04:05.578421  6970 net.cpp:200] pool1 does not need backward computatio
n.
I1005 15:04:05.578456  6970 net.cpp:200] conv1 does not need backward computatio
n.
I1005 15:04:05.578493  6970 net.cpp:200] data does not need backward computation
.
I1005 15:04:05.578529  6970 net.cpp:242] This network produces output prob
I1005 15:04:05.578563  6970 net.cpp:255] Network initialization done.
I1005 15:04:05.582970  6970 net.cpp:744] Ignoring source layer mnist
I1005 15:04:05.587973  6970 net.cpp:744] Ignoring source layer loss
[  1.84516597e-04   2.59596509e-05   1.60218251e-03   9.95539606e-01
   9.23390962e-06   4.69541419e-06   3.91907670e-05   1.88144724e-04
   2.08263192e-03   3.23790679e-04]
3
caffe@caffe-VirtualBox:~/caffe/python$
```

```python
import caffe
import numpy as np
import cv2
import sys
from PIL import Image
import matplotlib.pyplot as plt

model = '/home/caffe/caffe/examples/mnist/lenet.prototxt';
weights = '/home/caffe/caffe/examples/mnist/lenet_iter_10000.caffemodel';
net = caffe.Net(model,weights,caffe.TEST);
caffe.set_mode_cpu()
#img = caffe.io.load_image(sys.argv[1], color=False)
img = cv2.imread('3.jpg',0)
if img.shape != [28,28]:
    img2 = cv2.resize(img,(28,28))
    img = img2.reshape(28,28,-1);
else:
    img = img.reshape(28,28,-1);
#revert the image,and normalize it to 0-1 range
img = 1.0 - img/255.0
out = net.forward_all(data=np.asarray([img.transpose(2,0,1)]))

print out['prob'][0]
print out['prob'][0].argmax()
```