Example BVLC C++ classifier application using caffe

This application uses an already trained model from Caffe model Zoo called

https://github.com/BVLC/caffe/tree/master/models/bvlc_reference_caffenet

Which is licensed for unrestricted use .
The model is trained to classify 1000 objects in an image .
Description :

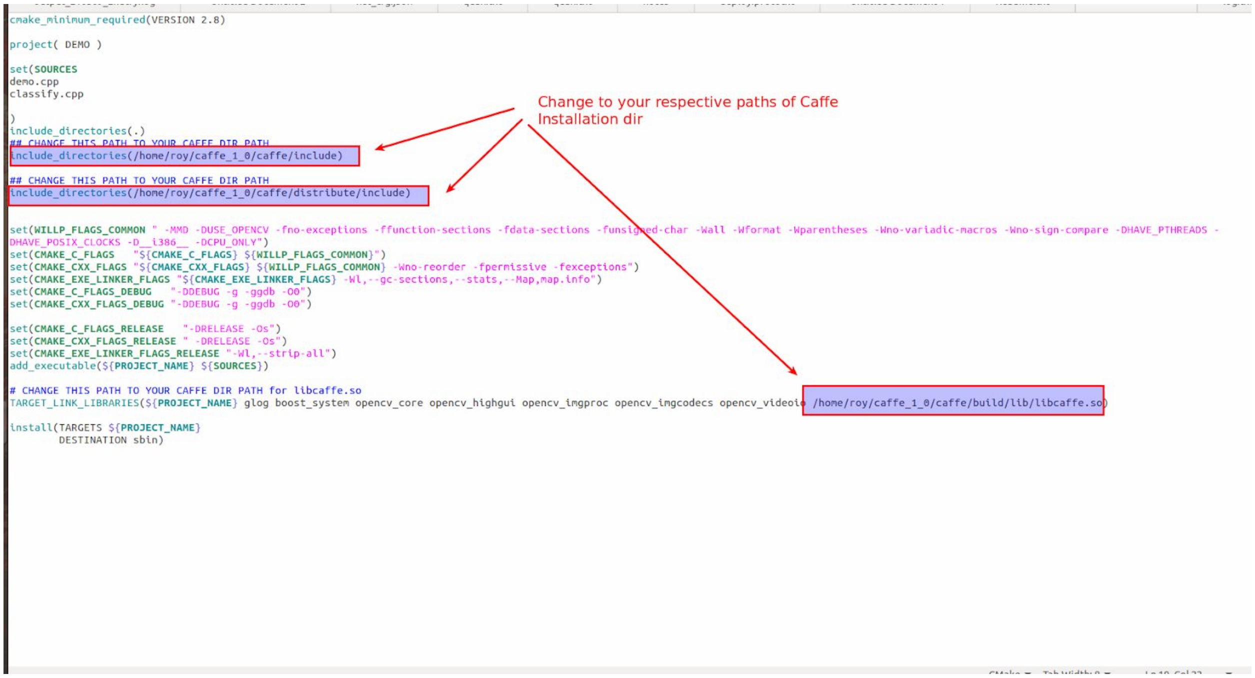
1) This sample first uses caffe_example/models/deploy.prototxt

Which has a Softmax layer defined as output layer and prints the probabilities

2) Then the sample uses/caffe_example/models/deploy_no_softmax.prototxt which has its softmax layer commented out and uses custom softmax function to print the probabilities .

To Compile and run the application

1. Go to folder caffe_example_catsAnddogs
2. Change the path in CMakeList.txt to your respective caffe install directory



```
cmake_minimum_required(VERSION 2.8)

project( DEMO )

set(SOURCES
demo.cpp
classify.cpp

)
include_directories(.)
## CHANGE THIS PATH TO YOUR CAFFE DIR PATH
include_directories(/home/roy/caffe_1_0/caffe/include)

## CHANGE THIS PATH TO YOUR CAFFE DIR PATH
include_directories(/home/roy/caffe_1_0/caffe/distribute/include)


set(WILLP_FLAGS_COMMON " -MMD -DUSE_OPENCV -fno-exceptions -ffunction-sections -fdata-sections -funsigned-char -Wall -Wformat -Wparentheses -Wno-variadic-macros -Wno-sign-compare -DHAVE_PTHREADS -
DHAVE_POSIX_CLOCKS -D__i386__ -DCPU_ONLY")
set(CMAKE_C_FLAGS   "${CMAKE_C_FLAGS} ${WILLP_FLAGS_COMMON}")
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} ${WILLP_FLAGS_COMMON} -Wno-reorder -fpermissive -fexceptions")
set(CMAKE_EXE_LINKER_FLAGS "${CMAKE_EXE_LINKER_FLAGS} -Wl,--gc-sections,--stats,--Map,map.info")
set(CMAKE_C_FLAGS_DEBUG   "-DDEBUG -g -ggdb -O0")
set(CMAKE_CXX_FLAGS_DEBUG "-DDEBUG -g -ggdb -O0")

set(CMAKE_C_FLAGS_RELEASE   "-DRELEASE -Os")
set(CMAKE_CXX_FLAGS_RELEASE " -DRELEASE -Os")
set(CMAKE_EXE_LINKER_FLAGS_RELEASE "-Wl,--strip-all")
add_executable(${PROJECT_NAME} ${SOURCES})

# CHANGE THIS PATH TO YOUR CAFFE DIR PATH for libcaffe.so
TARGET_LINK_LIBRARIES(${PROJECT_NAME} glog boost_system opencv_core opencv_highgui opencv_imgproc opencv_imgcodecs opencv_videoio /home/roy/caffe_1_0/caffe/build/lib/libcaffe.so)

install(TARGETS ${PROJECT_NAME}
        DESTINATION sbin)
```

Change to your respective paths of Caffe Installation dir

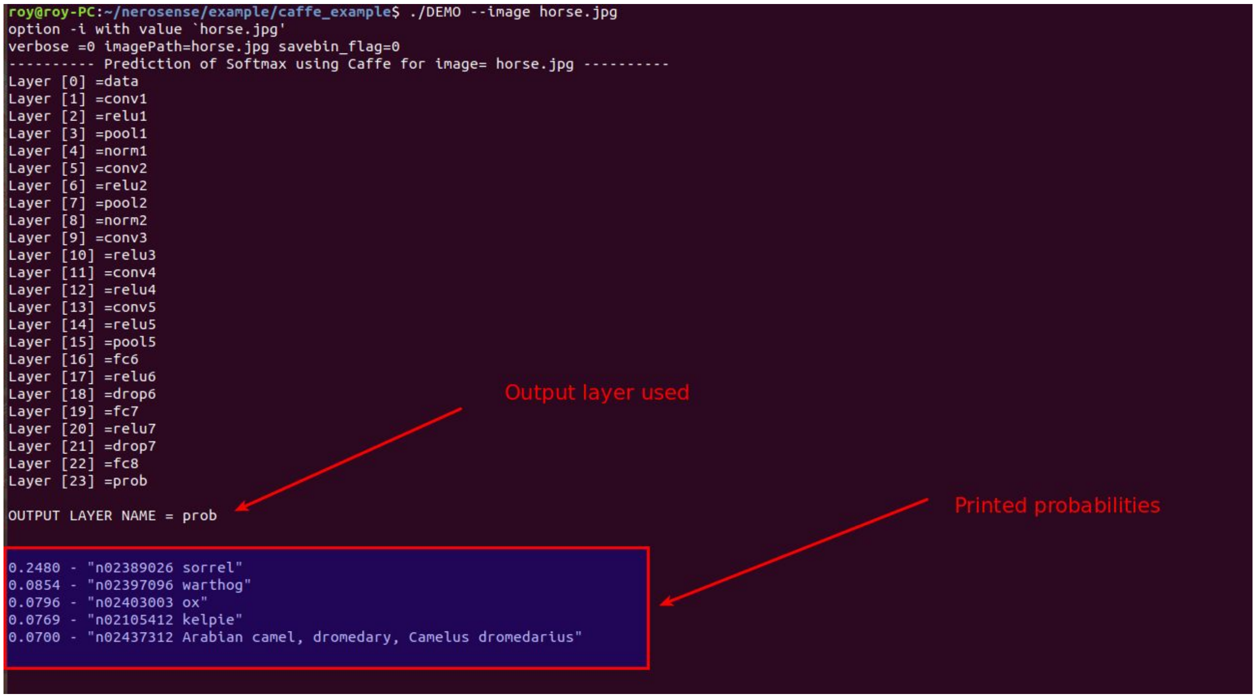
- Run cmake CMakeList.txt
- make

To classify a local image and compare probabities

Type **./DEMO --image <image.jpg>**

Below is a sample output from a sample image prediction

1. First application prints the output probability using Softmax layer within Caffe

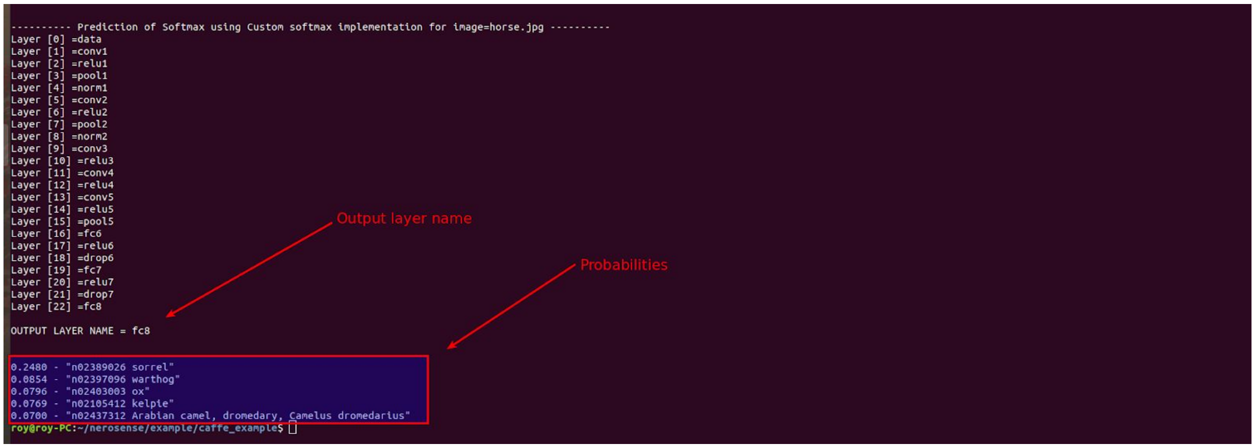


```
roy@roy-PC:~/nerosense/example/caffe_example$ ./DEMO --image horse.jpg
option -i with value `horse.jpg'
verbose =0 imagePath=horse.jpg savebin_flag=0
---------- Prediction of Softmax using Caffe for image= horse.jpg ----------
Layer [0] =data
Layer [1] =conv1
Layer [2] =relu1
Layer [3] =pool1
Layer [4] =norm1
Layer [5] =conv2
Layer [6] =relu2
Layer [7] =pool2
Layer [8] =norm2
Layer [9] =conv3
Layer [10] =relu3
Layer [11] =conv4
Layer [12] =relu4
Layer [13] =conv5
Layer [14] =relu5
Layer [15] =pool5
Layer [16] =fc6
Layer [17] =relu6
Layer [18] =drop6
Layer [19] =fc7
Layer [20] =relu7
Layer [21] =drop7
Layer [22] =fc8
Layer [23] =prob

OUTPUT LAYER NAME = prob

0.2480 - "n02389026 sorrel"
0.0854 - "n02397096 warthog"
0.0796 - "n02403003 ox"
0.0769 - "n02105412 kelpie"
0.0700 - "n02437312 Arabian camel, dromedary, Camelus dromedarius"
```

2. Then the application will print the probability using FC8 output and then performing a custom Softmax on the output



```
---------- Prediction of Softmax using Custom softmax implementation for image=horse.jpg ----------
Layer [0] =data
Layer [1] =conv1
Layer [2] =relu1
Layer [3] =pool1
Layer [4] =norm1
Layer [5] =conv2
Layer [6] =relu2
Layer [7] =pool2
Layer [8] =norm2
Layer [9] =conv3
Layer [10] =relu3
Layer [11] =conv4
Layer [12] =relu4
Layer [13] =conv5
Layer [14] =relu5
Layer [15] =pool5
Layer [16] =fc6
Layer [17] =relu6
Layer [18] =drop6
Layer [19] =fc7
Layer [20] =relu7
Layer [21] =drop7
Layer [22] =fc8

OUTPUT LAYER NAME = fc8

0.2480 - "n02389026 sorrel"
0.0854 - "n02397096 warthog"
0.0796 - "n02403003 ox"
0.0769 - "n02105412 kelpie"
0.0700 - "n02437312 Arabian camel, dromedary, Camelus dromedarius"
roy@roy-PC:~/nerosense/example/caffe_example$
```

Optional Arguments :

--verbose:

Use command **./DEMO --image <image.jpg> --verbose**

This command will print every step and also print the binary outputs of below float buffers :
When using using Softmax layer of Caffe

- prob Layer output

When using using FC8 layer of Caffe and performing custom Softmax

- Fc8 layer output
- Softmax function output

Below command can also be used to save the logs in a text file and for later viewing.

./DEMO --image <image.jpg> 2>&1 | tee ./log.txt

--savebin:

./DEMO --image <image.jpg> --savebin

This option is optional and will save the output as a binary file .
Below are the files it will generate .

- <imagename>prob_output.bin:prob Layer output When using using Softmax layer of Caffe
- <imagename>fc8_output.bin:Fc8 layer output
- <imagename>customsoftmax_output.bin:Custom Softmax function output

Results :

There was no major difference observed between the output of the Caffe Softmax layer and Custom Softmax function .