Configuration: **OS:** Linux 4.8.0-34-generic x86_64
Compiler: gcc 5.4.0
CPU: Intel® Core™ i7-6700K CPU @ 4.00GHz x 8
GPU: Intel® HD Graphics 530 (Skylake GT2)

	State
OpenCV	https://github.com/opencv/opencv/commit/d3a124c820807e6f20f22075575731a53e6b5674
Intel-Caffe	https://github.com/intel/caffe/commit/f6a2a6b05defab4b637028ce4f7719cac340a86d
clCaffe	https://github.com/BVLC/caffe/commit/2e7138570d324564c80c040b83ef6b1d4b39324d
TensorFlow	https://github.com/tensorflow/tensorflow/commit/438604fc885208ee05f9eef2d0f2c630e1360a83
Torch	https://github.com/torch/distro/tree/748f5e3c5c804eebf5715c0b47b1519d60ef4409
Halide	https://github.com/halide/Halide/commit/dac950a610ab01e9052541af34a150dc04e4fb93
LLVM/Clang	4.0.1
MKL	Build date 2017.04.13

The best observed median time of single image forward pass (in milliseconds):

CPU   All calculations are done in float32.

Model	DNN, C++	DNN, Halide	Intel-Caffe, MKLDNN	TensorFlow	Torch w. MKL
AlexNet	14.52	22.31	**11.95**		
GoogLeNet	17.37	32.43	**9.43**		
ResNet-50	40.01	76.13	**22.75**		
SqueezeNet v1.1	4.68	6.61	**3.05**		
Inception-5h	19.30	35.27		**14.6**	
ENet @ 512x256	65.93	**42.16**			226
OpenFace (nn4.small2)	**4.20**	8.14			25.44
MobileNet-SSD @ 300x300 20 classes, Caffe	**22.71**	54.36	27.79		
MobileNet-SSD @ 300x300 90 classes, TensorFlow	**25.15**	60.95		35.86	

GPU (OpenCL 2.0):   All computations in float-32.

Model	DNN, OpenCL backend	DNN, Halide	clCaffe, MKL
AlexNet	15.81	48.45	**15.16**
GoogLeNet	20.59	89.53	**19.56**
ResNet-50	**37.19**	183.67	63.26
SqueezeNet v1.1	6.50	15.7	**6.05**
Inception-5h	**22.68**	92.33	
ENet @ 512x256	**34.89**	48.92	
OpenFace (nn4.small2)	**10.55**	37.59	
MobileNet-SSD @ 300x30020 classes, Caffe	*172.13* (before #10341)***26.66*** (with #10341)	100.31	369.91
MobileNet-SSD @ 300x30090 classes, TensorFlow	*203.47* (before #10341)***45.11*** (with #10341)	93.34	

Scripts

TensorFlow

```python
import numpy as np
import tensorflow as tf
import time

with tf.gfile.FastGFile('opencv_extra/testdata/dnn/ssd_mobilenet_v1_coco.pb') as f:
    graph_def = tf.GraphDef()
    graph_def.ParseFromString(f.read())

with tf.Session() as sess:
    sess.graph.as_default()
    tf.import_graph_def(graph_def, name='')

    # Generate input
    np.random.seed(2701)
    inp = np.random.standard_normal([1, 300, 300, 3]).astype(np.float32)

    # Get output tensor
    outTensors = [sess.graph.get_tensor_by_name('num_detections:0'),
                  sess.graph.get_tensor_by_name('detection_scores:0'),
                  sess.graph.get_tensor_by_name('detection_boxes:0'),
                  sess.graph.get_tensor_by_name('detection_classes:0')]

    def run():
        out = sess.run(outTensors, feed_dict={'image_tensor:0': inp})
```

```python
# Warm up
for _ in range(3):
    run()

# Measure
N = 10
start = time.time()
for _ in range(N):
    run()
print 1e+3 * (time.time() - start) / N
```

Torch

```lua
require 'nn'
require 'dpnn'
require 'image'

torch.setdefaulttensortype('torch.FloatTensor')

net = torch.load('opencv_extra/testdata/dnn/openface_nn4.small2.v1.t7')

input = torch.FloatTensor(torch.LongStorage({1, 3, 96, 96}))

net:evaluate()
-- Warm up
for i = 1,3 do
  output = net:forward(input)
end

N = 10
timer = torch.Timer()
start = timer:time().real
for i = 1,N do
  output = net:forward(input)
end
print(1000 * (timer:time().real - start) / N)
```

References

- OpenCV's deep learning module, https://github.com/opencv/opencv/tree/master/modules/dnn.
- Intel-Caffe, https://github.com/intel/caffe.
- clCaffe, https://github.com/01org/caffe.
- TensorFlow, https://www.tensorflow.org/.
- Torch, http://torch.ch/.
- Halide, http://halide-lang.org/.