Project info

Name: OpenCV dnn module enhancement
Organization: OpenCV
Student: Vitaliy Lyudvichenko (GitHub) (Email)
Mentor: Anatoly Baksheev (GitHub)
Useful docs: (Project weekly log) (Ideas)

Links to made commits and pull requests

Pull requests:

- The first merged PR for dnn module https://github.com/opencv/opencv_contrib/pull/707
- PR with required bugfix in core module https://github.com/opencv/opencv/pull/6747
- PR with bugfixes and enhancements of python bindings generator https://github.com/opencv/opencv/pull/7056
- PR adding Python bindings https://github.com/opencv/opencv_contrib/pull/750 and new layers

My commits into opencv contrib repo (main contribution):

- https://github.com/opencv/opencv_contrib/commits/master?author=ludv1x

My commits into main opencv repo (minor contribution):

- https://github.com/opencv/opencv/commits/master?author=ludv1x

My commits into opencv extra repo (minor contribution):

- https://github.com/opencv/opencv_extra/commits/master?author=ludv1x

My fork with work-in-progress including unmerged changes (`dnn*` branches):

- https://github.com/ludv1x/opencv_contrib

Description of contribution

- Added support of custom BLAS libraries.
- Added support of most-useful BLAS libraries: MKL, OpenBLAS, Atlas.
- Evaluated performance of above mentioned libraries (results).
- Added recurrent layers LSTM and RNN
- Extended Blob API, added logical switching between CPU and GPU data and new methods
- Using new API added OpenCL-accelerated implementations of some layers in “single code - two implementations” fashion which leverage existing OpenCV functionality
- Added OpenCL implementations for remaining layers using kernels from Caffe-OpenCL project
- Refactored layers implementations:
    a. Added publicly interface for all layers

    b. Separated layers code and code importing layer's parameters from supporting dnn-frameworks
    c. Added factory for creating layers from Caffe parameters
- Bugfixes and enhancements of OpenCV python bindings generator:
    a. Support of custom Python—C++ type conversions defined in each module
    b. Fixed of invalid initialization and memory leaks in code generator for simple classes
- Added full-scale Python bindings for dnn module, extended module API for more convenient usage from Python (and C++ also).