Intel® Math Kernel Library
(Intel® MKL)

Intel® Math Kernel Library (Intel® MKL) Introduction

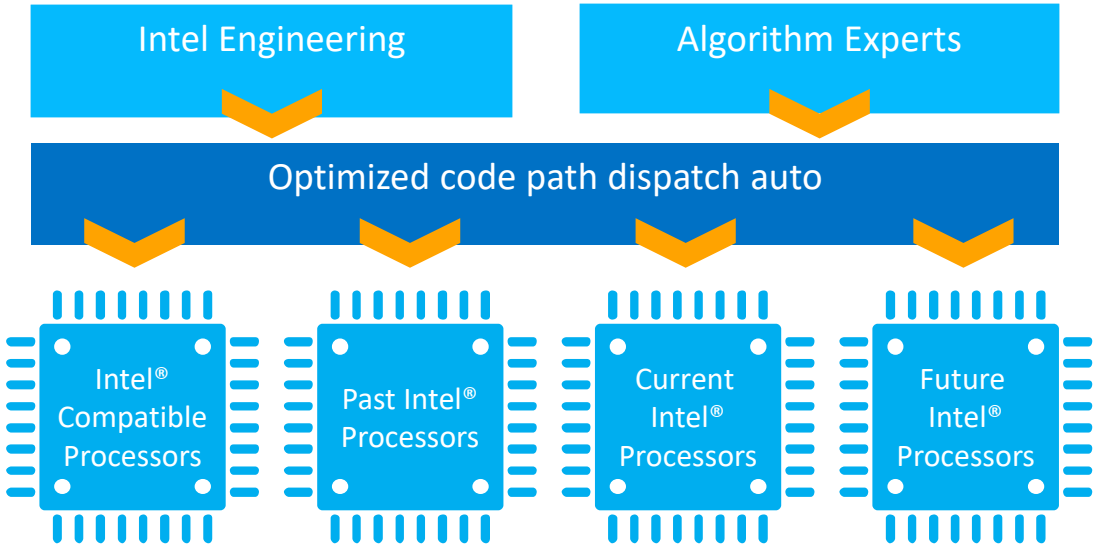
Highly optimized threaded math routines

- Performance, Performance, Performance!

Industry's leading math library

- Widely used in science, engineering, data processing

Tuned for Intel® processors – current and next generation



Evans Data Corporation EDC

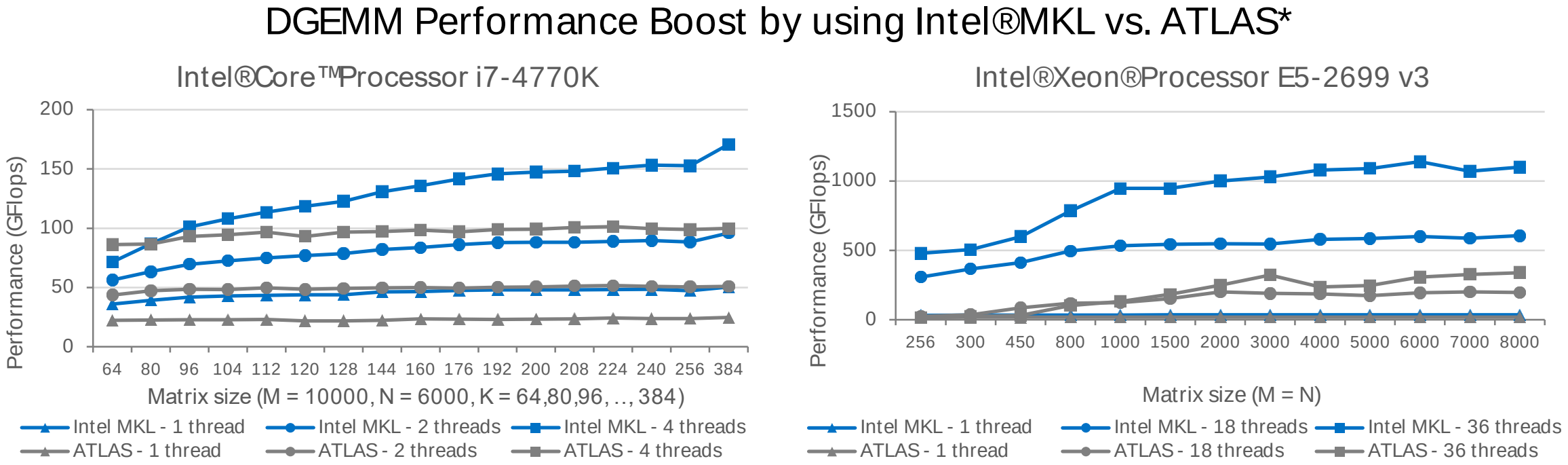
EDC North America Development Survey 2016, Volume I

More math library users depend on MKL than any other library

Be multiprocessor aware

- Cross-Platform Support
- Be vectorised , threaded, and distributed multiprocessor aware

Intel MKL unleashes the performance benefits of Intel architectures



Configuration Info - Versions: Intel®Math Kernel Library (Intel®MKL) 11.3, ATLAS* 3.10.2; Hardware: Intel®Xeon®Processor E5-2699v3, 2 Eighteen-core CPUs (45MB LLC, 2.3GHz), 64GB of RAM; Intel®Core™Processor i7-4770K, Quad-core CPU (8MB LLC, 3.5GHz), 8GB of RAM; Operating System: RHEL 6.4 GA x86_64;

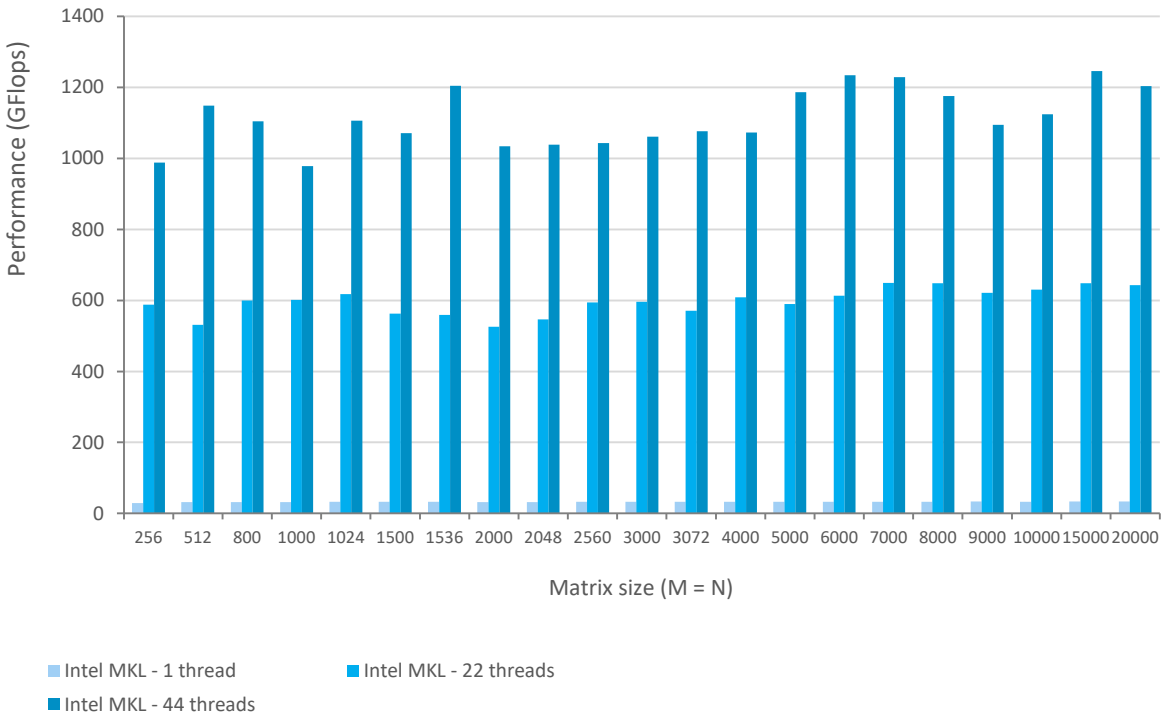
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Intel MKL 2017 Performance

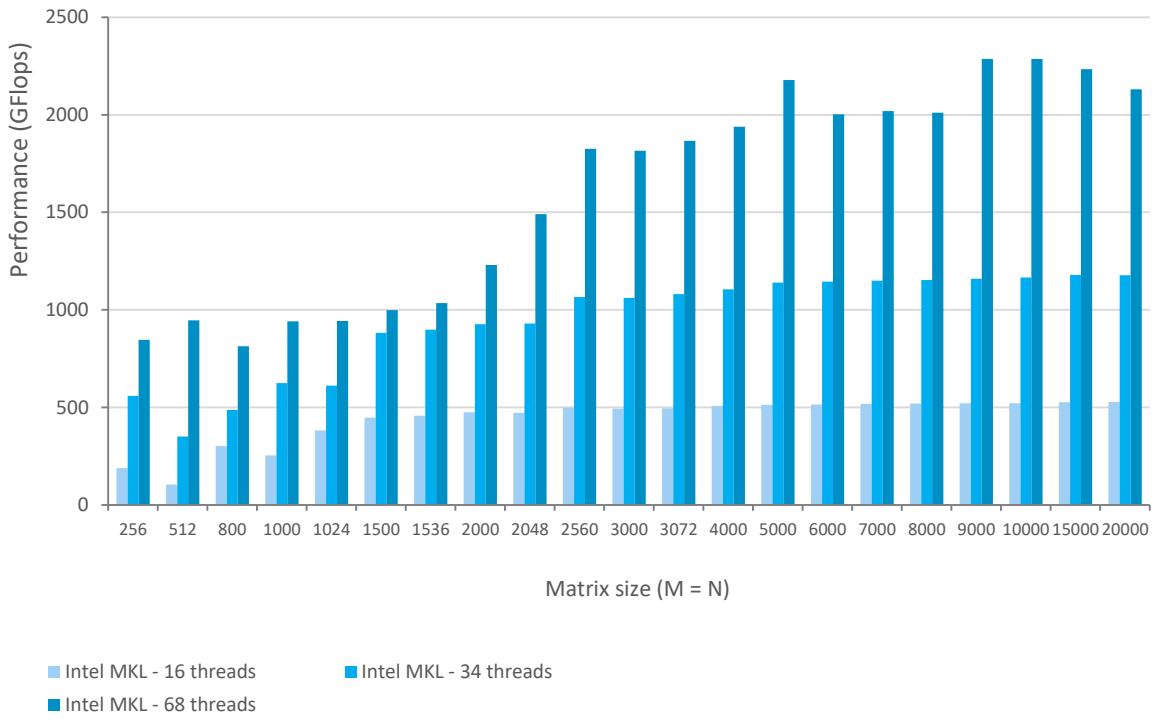
DGEMM Performance

On Intel® Xeon® Processor E5-2699 v4



DGEMM Performance

On Intel® Xeon Phi™ Processor 7250



Configuration Info - Versions: Intel® Math Kernel Library (Intel® MKL) 2017; Hardware:Hardware: Intel® Xeon® Processor E5-2699 v4, 2 Twenty-two-core CPU (55MB smart cache, 2.2GHz), 64GB of RAM; Intel® Xeon Phi™ Processor 7250, 68 cores (34MB L2 cache, 1.4GHz), 96 GB of DDR4 RAM and 16 GB MCDRAM; Operating System: RHEL 7.2 GA x86_64;

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Optimized Mathematical Building Blocks

Linear Algebra	Fast Fourier Transforms	Vector Math	Vector RNGs	Summary Statistics	Deep Neural Networks (DNN)
• BLAS • LAPACK • ScaLAPACK • Sparse BLAS • Sparse Solvers • Iterative • PARDISO* SMP & Cluster	• Multidimensional • FFTW interfaces • Cluster FFT	• Trigonometric • Hyperbolic • Exponential • Log • Power • Root	• Congruential • Wichmann-Hill • Mersenne Twister • Sobol • Neiderreiter • Non-deterministic	• Kurtosis • Variation coefficient • Order statistics • Min/max • Variance-covariance	• Convolution • Pooling • Normalization • ReLU • Softmax

BLAS – Basic Linear Algebra Subprograms

Defacto-standard APIs since the 1980s (Fortran 77)

- Level 1 – vector-vector operations
- Level 2 – matrix-vector operations
- Level 3 – matrix-matrix operations
- Precisions: single, double, single complex, double complex

Original BLAS available at
http://netlib.org/blas/

Operation	MKL Routine "D is for double"	Example	Computational complexity (work)
Vector Vector	D AXPY	$y = y + \alpha x$	$O(N)$
Matrix Vector	D GEMV	$y = \alpha Ax + \beta y$	$O(N^2)$
Matrix Matrix	D GEMM	$C = \alpha A * B + \beta C$	$O(N^3)$

LAPACK – Linear Algebra PACKage

Defacto-standard APIs since early 1990s

1000s of linear algebra functions

4 floating point precisions supported

Breadth of coverage:

- Matrix factorizations: the 3 Amigos – LU, Cholesky, QR
- Solving systems of linear equations
- Condition number estimates
- Singular value decomposition
- Symmetric and non-symmetric eigenvalue problems
- And much, much more

Original LAPACK is available at:
http://netlib.org/lapack/

Fast Fourier Transform (FFT)

Support multidimensional transforms

Multiple transforms on single call

Input/output strides supported

Allow FFT of a part of image, padding for better performance, transform combined with transposition, facilitates development of mixed-language applications.

Integrated FFTW interfaces

Source code of FFTW3 and FFTW2 wrappers in C/C++ and Fortran are provided.

FFTW3 wrappers are also built into the library.

Vector Math Functions

Example: $y(i) = e^{x(i)}$ for $i = 1$ to n

- Arithmetic
  - add/sub/sqrt/ ...
- Exponential and log
  - exp/pow/log/log10
- Trigonometric and hyperbolic
  - sin/cos/sincos/tan(h)
  - asin/acos/atan(h)
- Rounding
  - ceil, floor, round ...
- And many more ...

- Real and complex
- Single/double precision
- 3 accuracy modes
  - High accuracy
    - (Almost correctly rounded)
  - Low accuracy
    - (2 lowest bits in error)
  - Enhanced performance
    - (1/2 the bits correct)

Vector-based elementary functions allow developers to balance accuracy with performance

Vector Statistics

Random Number Generators (RNGs)	Psuedo-random, quasi-random, and non-deterministic generators
	Continuous and discrete distributions of various common distribution types
Summary Statistics (SS)	Parallelized algorithms for computation of statistical estimates for raw multi-dimensional datasets.
Convolution/ correlation	A set of routines intended to perform linear convolution and correlation transformations for single and double precision real and complex data.

Intel® MKL Sparse Solvers

PARDISO – Parallel Direct Sparse Solver	Support a wide range of matrix types.
	Based on BLAS level 3 update and pipelining parallelism.
	Supports out-of-core execution for huge problem sizes.
	New: Cluster support.
DSS – Direct Sparse Solver Interface for PARDISO	An alternative, simplified interface to PARDISO.
ISS – Iterative Sparse Solver	Symmetric positive definite: CG solver.
	Non-symmetric indefinite: Flexible generalized minimal residual solver.
	Based on Reverse Communication Interface (RCI).

More Intel® MKL Components

Data Fitting

1D linear, quadratic, cubic, step-wise const, and user-defined splines

Spline based interpolation/extrapolation

PDEs (Partial Differential Equations)

Solving Helmhotz, Poisson, and Laplace problems.

Optimization Solvers

Solvers for nonlinear least square problems with/without constraints

Support Functions

Memory management

Threading control

…

What are Intel MKL DNN Primitives?

A set of performance primitives to speed up image recognition topologies on existing or custom NN frameworks

- Topologies: AlexNet, VGG, GoogleNet, ResNet
- Frameworks: Caffe*, TensorFlow*, CNTK*, Torch*, MXNet*, ……

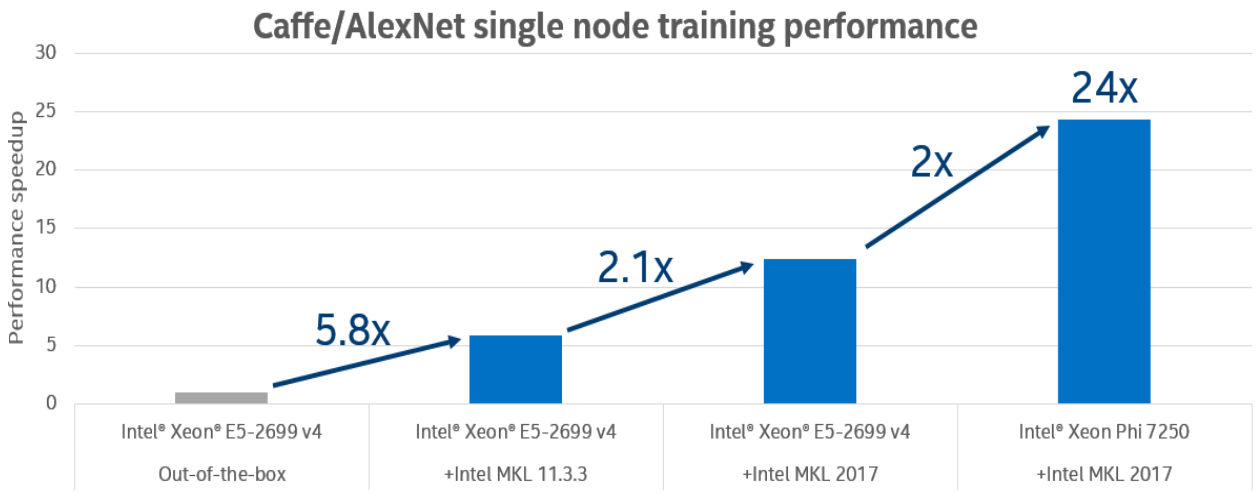
Operations (forward/backward)	Algorithms
Activation	ReLU
Normalization	batch, local response
Pooling	max, min, average
Convolutional	fully connected, direct batched convolution
Inner product	forward/backward propagation of inner product computation
Data manipulation	layout conversion, split, concat, sum, scale

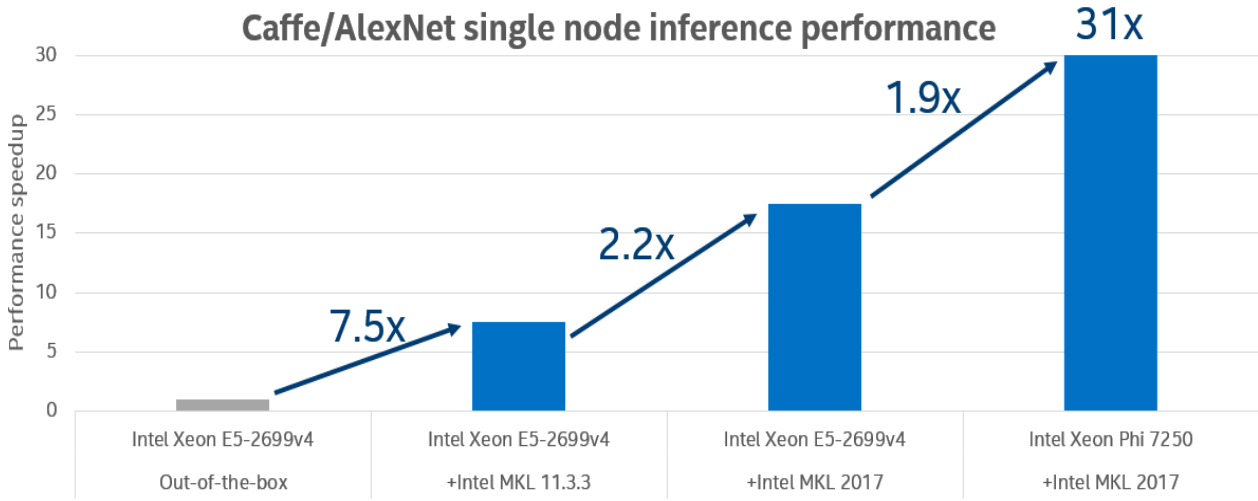
DNN Primitives in Intel® MKL Highlights

A plain C API to be used in the existing DNN frameworks

Brings IA-optimized performance to popular image recognition topologies:

- AlexNet, Visual Geometry Group (VGG), GoogleNet, and ResNet





Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to http://www.intel.com/performance . *Other names and brands may be property of others

Configurations:

- 2 socket system with Intel® Xeon® Processor E5-2699 v4 (22 Cores, 2.2 GHz,), 128 GB memory, Red Hat* Enterprise Linux 6.7, BVLC Caffe, Intel Optimized Caffe framework, Intel® MKL 11.3.3, Intel® MKL 2017
- Intel® Xeon Phi™ Processor 7250 (68 Cores, 1.4 GHz, 16GB MCDRAM), 128 GB memory, Red Hat* Enterprise Linux 6.7, Intel® Optimized Caffe framework, Intel® MKL 2017

All numbers measured without taking data manipulation into account.

What's New: Intel® MKL 2017

- Optimized math functions to enable neural networks (CNN and DNN) for deep learning
- Improved ScaLAPACK performance for symmetric eigensolvers on HPC clusters
- New data fitting functions based on B-splines and monotonic splines
- Improved optimizations for newer Intel processors, especially Knight's Landing Xeon Phi
- Extended TBB threading layer support for all BLAS level-1 functions

Intel® MKL Summary

Intel MKL boosts application performance with minimal effort

- feature set is robust and growing
- provides scaling from the core, to multicore, to manycore, and to clusters
- automatic dispatching matches the executed code to the underlying processor
- future processor optimizations included well before processors ship

Showcases the world's fastest supercomputers[1]

- Intel® Optimized MP LINPACK Benchmark
- Intel® Optimized Technology Preview for High Performance Conjugate Gradient Benchmark

[1]http://www.top500.org

Intel® Integrated Performance Primitives (Intel® IPP)

Intel® IPP Your Building Blocks for Image, Signal & Data Processing Applications

What is Intel® IPP?

Intel IPP provides developers with ready-to-use, processor optimized functions to accelerate ***Image, Signal, Data Processing & Cryptography computation tasks***

Why should you use Intel® IPP?

- High Performance
- Easy to use API's
- Faster Time To Market (TTM)
- Production Ready

How to get Intel® IPP?

Intel Parallel Studio XE
Intel System Studio
Free Tools Program

Optimized for

Supports

Addresses

Find out more at: ***http://software.intel.com/intel-ipp***
Contact us through our forum:

http://software.intel.com/en-us/forums/intel-integrated-performance-primitives

Image Processing

- Medical Imaging
- Computer Vision
- Digital Surveillance
- Biometric Identification
- Automated Sorting
- ADAS
- Visual Search

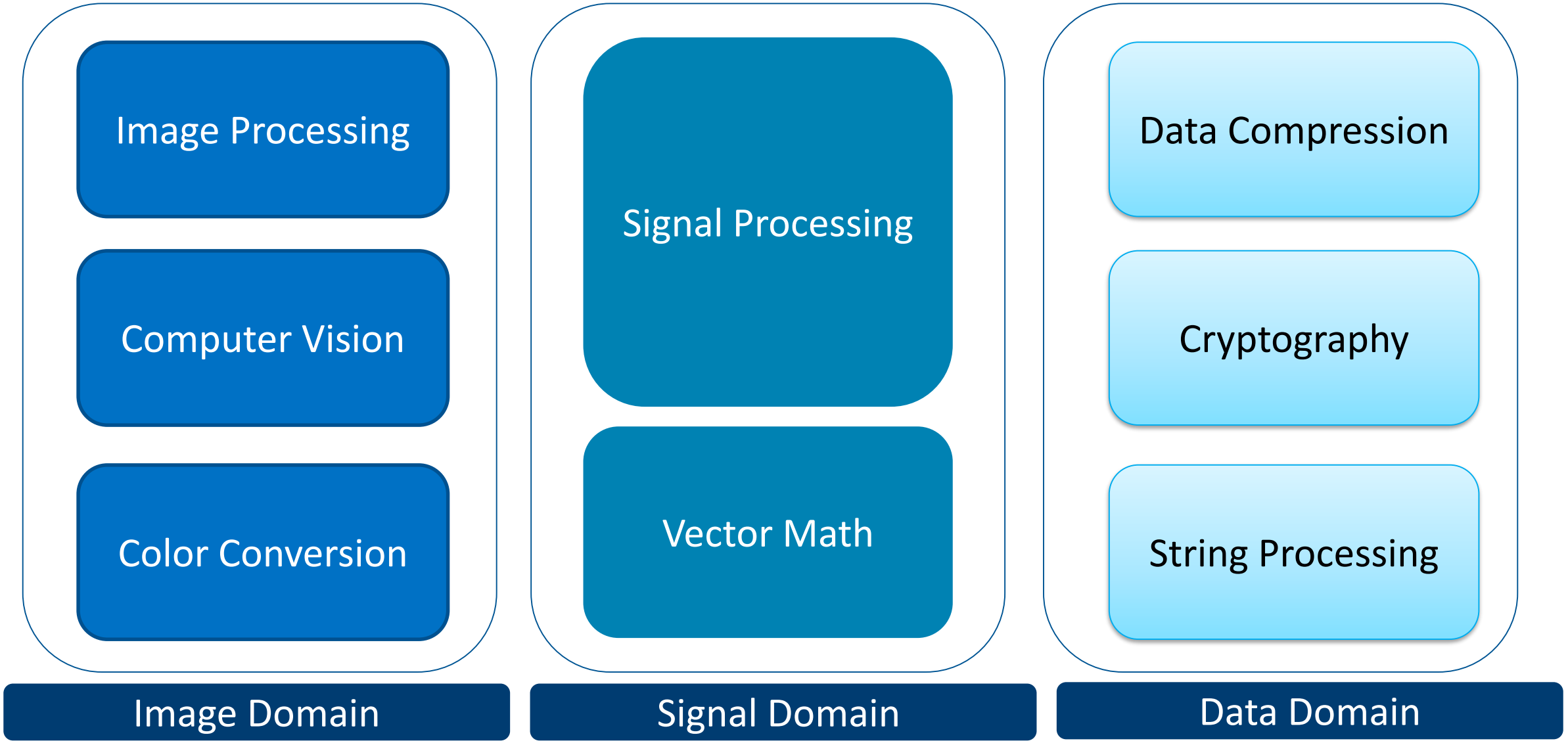
Signal Processing

- Games (sophisticated audio content or effects)
- Echo cancellation
- Telecommunications
- Energy

Data Compression & Cryptography

- Data centers
- Enterprise data Managements
- ID verification
- Smart Cards/wallets
- Electronic Signature
- Information security/cybersecurity

Different Domains in Intel IPP



Ready to use high performance Image processing functions (2D)

Filters/Transforms for Image processing

- Geometry transformations, such as resize/rotate.
- Linear and non-linear filtering operation on an image for edge detection, blurring, noise removal, etc.
- Linear transforms for 2D FFTs, DFTs, DCT.
- Image statistics and analysis.

Computer Vision

- Background differencing, Feature Detection (Corner Detection, Canny Edge detection), Distance Transforms.
- Image Gradients, Flood fill, Motion analysis and Object Tracking,
- Pyramids, Pattern recognition, Camera Calibration
- Canny, Optical Flow, Segmentation, Haar Classifiers

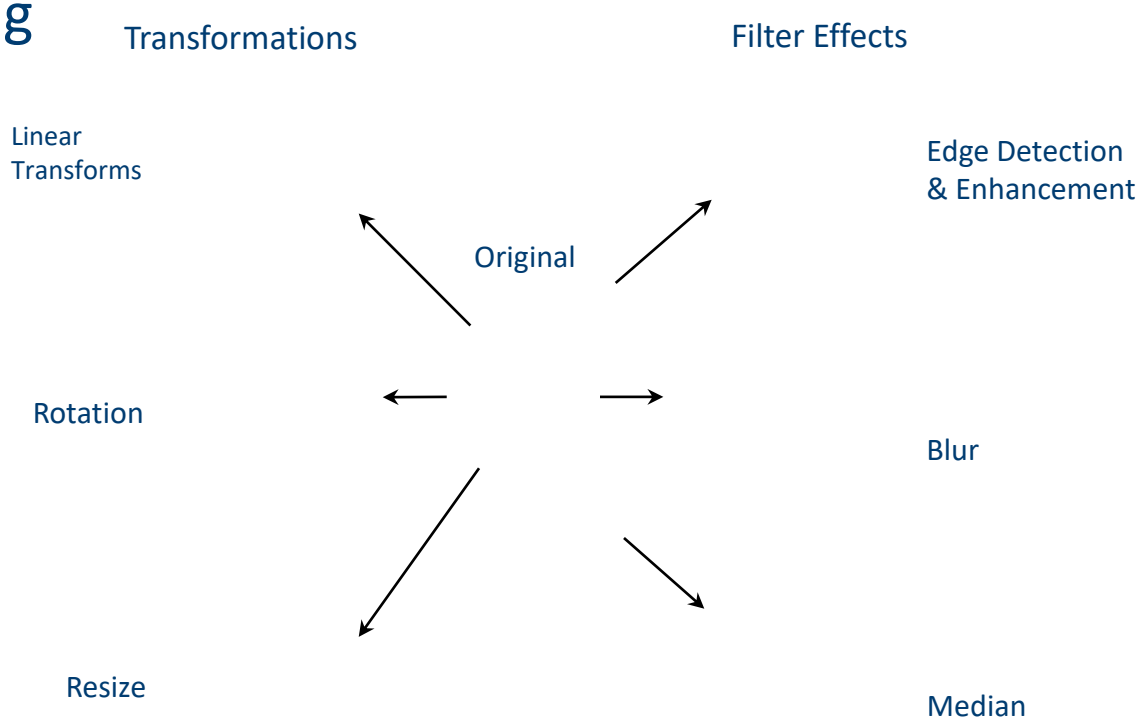
Color Conversion

- Convert image/video color space formats: RGB, HSV, YUV, YCbCr
- Up/Down sampling,
- Brightness and contrast adjustments

IPP Image Processing Function Sets

Ready to use high performance Image processing functions (2D):

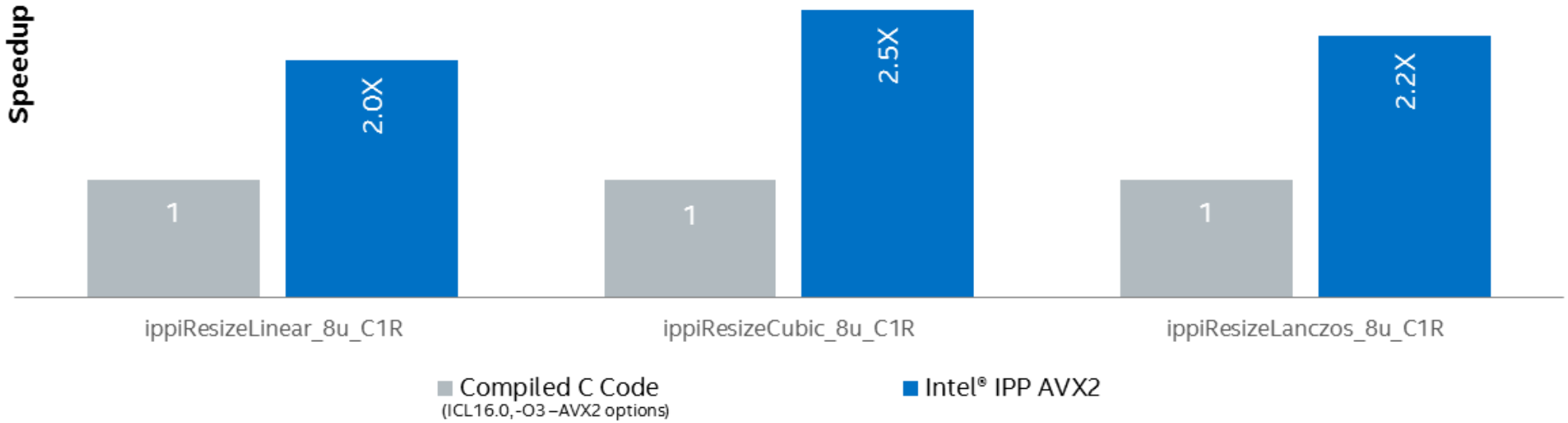
- Memory allocation and image initialization
- Arithmetic and logical operations
- Color conversion
- Compare and threshold
- Morphological operations
- Filtering
- Linear transforms
- Geometry transforms
- Image statistical functions



More than two thousands functions for processing different format images

Intel® IPP Image Resize Functions Performance Boost

Intel® IPP AVX2 Optimization Code vs. Compiled C Code

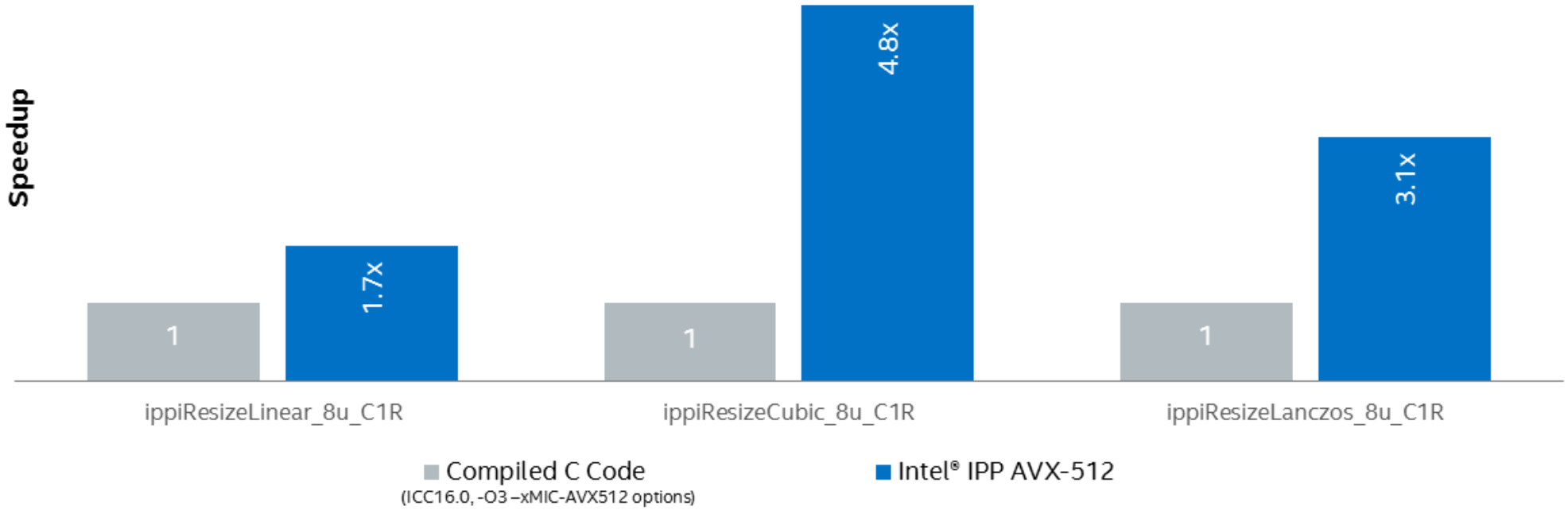


Configuration Info– SW Versions: Intel® Integrated Performance Primitives (Intel® IPP) 2017. Hardware: Intel® Core™ Processor i7-6700K, 8 MB cache, 4.2 GHz, 16 GB RAM, Windows Server® 2012 R2, single-threaded. Compiler: Intel C++ Compiler 16.0, with -O3 –AXV2 options
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Intel® IPP Image Resize Functions Performance Boost

Intel® IPP AVX-512 Optimization Code vs. Compiled C Code



Configuration Info– SW Versions: Intel® Integrated Performance Primitives (Intel® IPP) 2017, RH EL Server 7.0, 64-bit. Hardware: Intel® Xeon Phi™ Processor 7250, 32 MB L2 cache, 1.4 GHz. Single-threaded. Compiler: Intel C++ Compiler 16.0, with –xMIC-AVX512 options
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Ready to use high performance Signal processing functions (1D)

Signal Processing

- FFT, DFT, DCT, MDCT
- Wavelet, Hilbert, Hartley and Walsh-Hadamard Transforms
- Convolution, Cross-Correlation, Auto-Correlation, Conjugate
- Windowing, Jaehne/Tone/Triangle signal generation

Digital Filtering

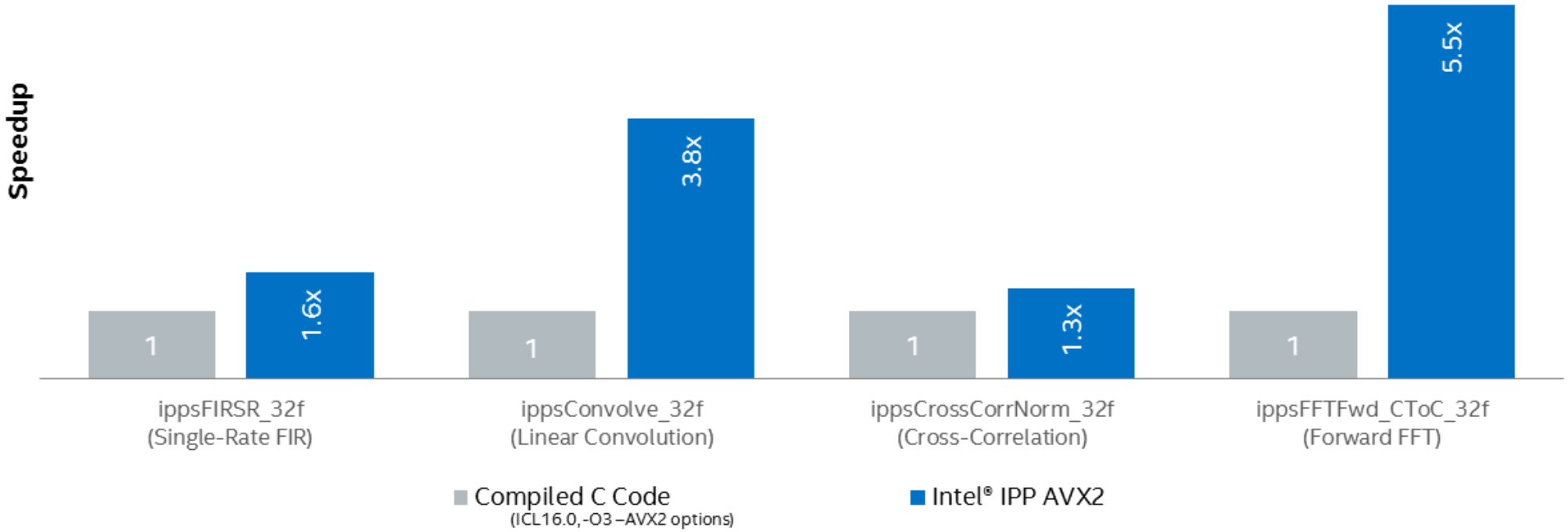
- Finite Impulse Response (FIR), Infinite Impulse Response (IIR), Single-Rate Adaptive FIR, Multi-Rate Adaptive FIR, Median Filter, Convolution and Correlation,
- Coordinate Conversions (polar↔cartesian), Numeric Conversion (real↔complex), Emphasize, Nearest Neighbor, Threshold

Statistics

- Mean
- StdDev
- NormDiff
- Sum
- MinMax

Intel® IPP Signal Processing Functions Performance Boost

Intel® IPP AVX2 Optimization Code vs. Compiled C Code

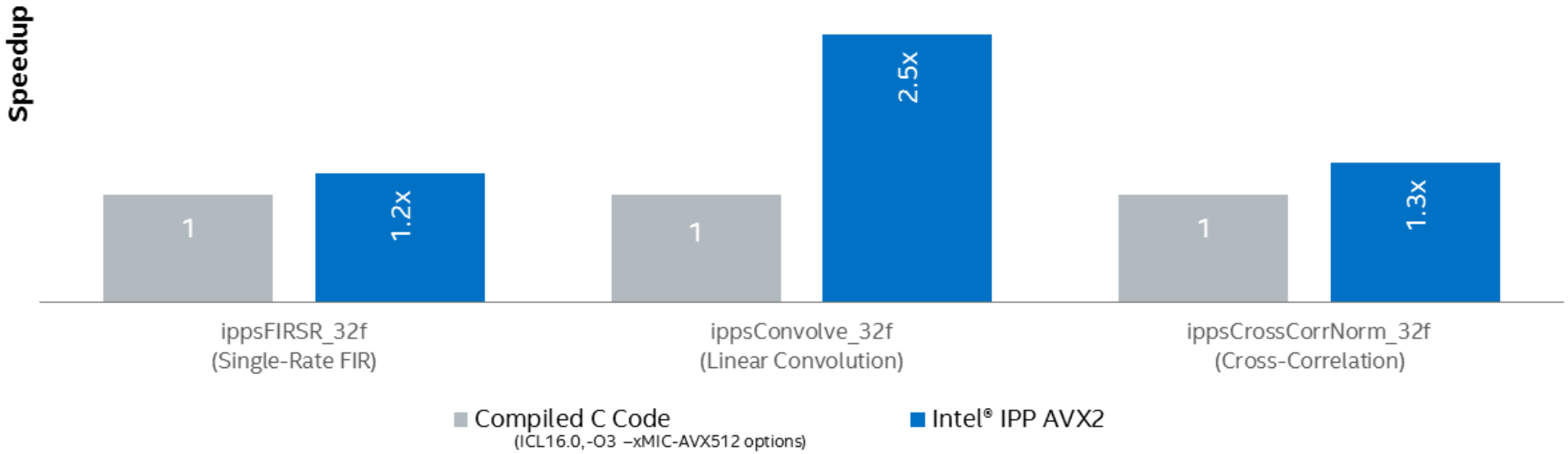


Configuration Info– SW Versions: Intel® Integrated Performance Primitives (Intel® IPP) 2017. Hardware: Intel® Core™ Processor i7-6700K, 8 MB cache, 4.2 GHz, 16 GB RAM, Windows Server® 2012 R2, single-threaded. Compiler: Intel C++ Compiler 16.0, with -O3 –AXV2 options
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Intel® IPP Signal Processing Functions Performance Boost

Intel® IPP AVX-512 Optimization Code vs. Compiled C Code



Configuration Info– SW Versions: Intel® Integrated Performance Primitives (Intel® IPP) 2017, RH EL Server 7.0, 64-bit. Hardware: Intel® Xeon Phi™ Processor 7250, 32 MB L2 cache, 1.4 GHz. Single-threaded. Compiler: Intel C++ Compiler 16.0, with –xMIC-AVX512 options
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Intel® IPP Data Processing

Data Compression

- Huffman/Variable Length Coding (VLC)
- Lempel-Ziv-Storer-Syzmanski (LZSS) PKzip
- Lempel-Ziv (lz77) Zlib,gzip
- Lempel-Ziv-Oberhumer (LZO) lzop
- Burrows-Wheeler Transform (BWT) bzip2

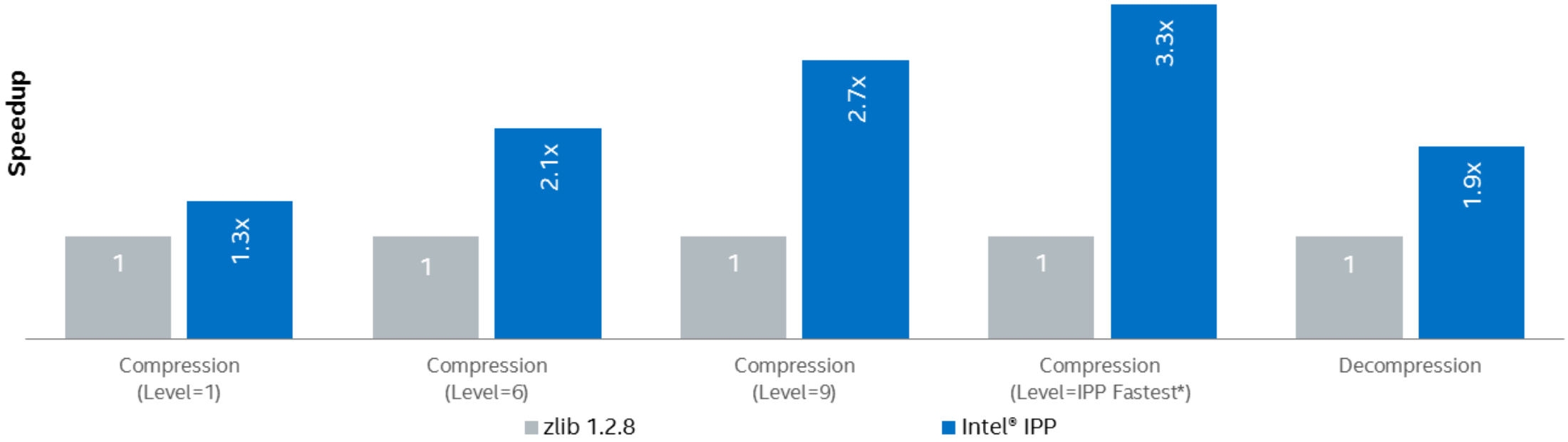
Cryptography

- Symmetric cryptography (AES, TDES)
- One-way hash (SHA1, MD5)
- Public key cryptography (RSA)

String Processing

- Find
- Insert
- Remove
- Compar
- Trim
- Replace
- Upper, Lower
- Hash
- Concatenate, Split
- Regular Expression Find/Replace

Intel® IPP Data Compression and Decompression Performance Boost vs. ZLIB Library

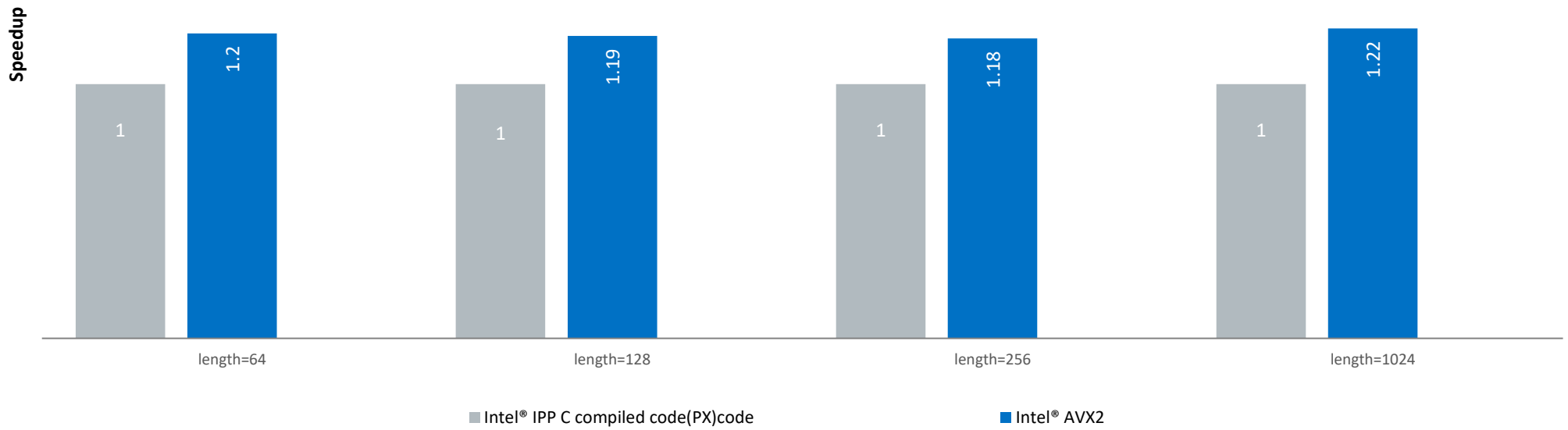


Configuration Info– SW Versions: Intel® Integrated Performance Primitives (Intel® IPP) 2017, Intel C++ Compiler 16.0. Hardware: Intel® Core™ Processor i7-6700K, 8 MB cache, 4.2 GHz, 16 GB RAM, Windows Server® 2012 R2. Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. *Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Intel® IPP MD5 Performance Boost

Intel® AVX2 Optimization Code vs. Intel® IPP C optimized code



Configuration Info – SW Versions: Intel® Integrated Performance Primitives (Intel® IPP) 9.0. Hardware: Intel® Core™ Processor i5-4300U, 3 MB Intel® Smart Cache, 8 GB RAM. Operating system: Windows* 8 64-bit, single-threaded. Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Code Dispatching

- Intel IPP functions are optimized for specific processors.
- A single function has many versions, each one optimized to run on a specific processor.
- New CPU dispatching code K0 for processors with Intel® Advanced Vector Extensions 512 (Intel® AVX-512).

ippsCopy_8u(…)

Some examples:

mx_ippsCopy_8u(…)
l9_ippsCopy_8u(…)
k0_ippsCopy_8u(…)

Platform	Identifier	Optimization
IA-32 Intel® Architecture	**px**	C-optimized for all IA-32 processors
	w7	Optimized for processors with Intel SSE2
	p8	Optimized for processors with Intel® Streaming SIMD Extensions 4.1 (Intel SSE4.1)
	h9	Optimized for processors with Intel® Advanced Vector Extensions 2 (Intel® AVX2)
Intel® 64 architecture	**mx**	C-optimized for processors with Intel® 64 instructions set architecture
	n8	Optimized for the Intel® Atom™ processor
	y8	Optimized for 64-bit applications on processors with Intel® Streaming SIMD Extensions 4.1 (Intel SSE4.1)
	e9	Optimized for processors that support Intel® Advanced Vector Extensions instruction set
	l9	Optimized for processors with Intel® Advanced Vector Extensions 2 (Intel® AVX2)
	k0	Optimized for processors with Intel® Advanced Vector Extensions 512 (Intel® AVX-512)

Intel® IPP Linkage

Linkage Models Features	1. Dynamic Linkage	2. Static Linkage with Dispatching	3. Custom Dynamic Linkage	4. Static Linkage without Dispatching
Optimization	All processors	All processors	All processors	One processor
Build	Link to stub libraries	Link to static libraries and stubs	Build separate DLL	Link to processor-specific merged libraries
Calling	Regular names	Regular names	Modified names	Processor-specific names
Total Binary Size	Large	Small	Small	Smallest
Executable Size	Smallest	Small	Smallest	Small
Kernel Mode	No	Yes	No	Yes

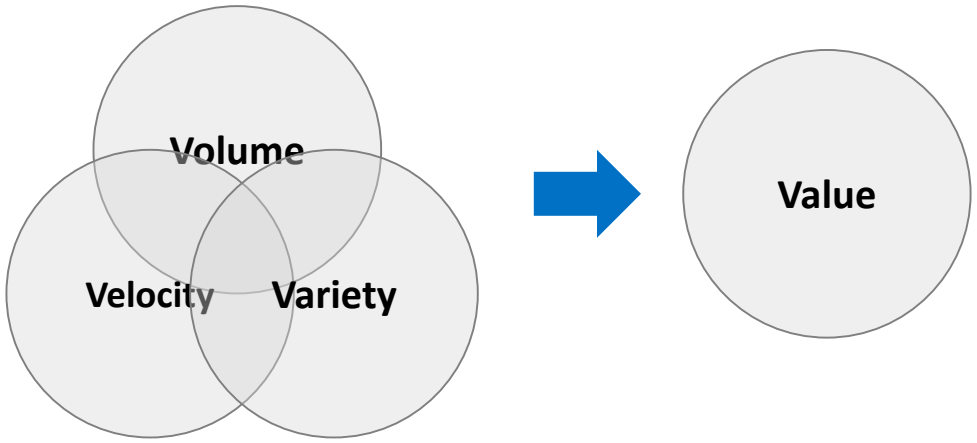
Intel® IPP provides a lot of flexibility

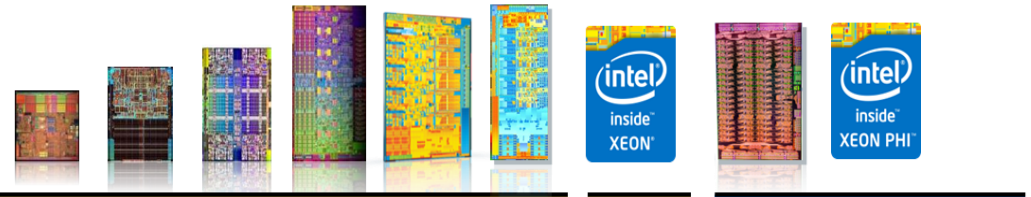
New Enhancements in Intel® IPP 2017

- New zlib example to provide drop-in optimization of zlib library with Intel® IPP functions.
  - Provided additional optimization on pattern matching algorithms.
  - Added a new fastest compression level with best compression performance.
- Added Intel® IPP Platform-Aware APIs to support 64-bit parameters:
  - New 64-bit parameters can support large image and vector data processing.
  - Support external tiling and threading by processing tiled images, and enables application level threading.
- New C and C++ integration wrappers for the image processing and computer vision functions.
- New performance improvement and optimization:
  - Support for recently launched versions of Intel® processors, including Intel® Xeon Phi™ processor x200 (formerly code-named Knights Landing).
  - Extended Intel® SSE4.2, Intel® AVX-512 and Intel® AVX2 optimization.

Intel® Data Analytics Acceleration Library (Intel® DAAL)

Data Analytics in the Age of Big Data





	Intel® Xeon® Processor 64-bit	Intel® Xeon® Processor 5100 series	Intel® Xeon® Processor 5500 series	Intel® Xeon® Processor 5600 series	Intel® Xeon® Processor E5-2600 v2 series	Intel® Xeon® Processor E5-2600 v3 series	~	Future Intel® Xeon® Processor[1]	Intel® Xeon Phi™ x100 Coprocessor	Intel® Xeon Phi™ x200 Processor & Coprocessor
Up to Core(s)	1	2	4	6	12	18	~	*Tbd*	57-61	TBD
Up to Threads	2	2	8	12	24	36		*tbd*	228-244	TBD
SIMD Width	128	128	128	128	256	256	~	512	512	512
Vector ISA	Intel® SSE3	Intel® SSE3	Intel® SSE4.2	Intel® AVX	Intel® AVX	Intel® AVX2		Intel® AVX-512	IMCI 512	Intel® AVX-512

Problem:

- Big data needs high performance computing.
- Many big data applications leave performance at the table – Not optimized for underlying hardware.

Solution:

- A performance library provides building blocks to be easily integrated into big data analytics workflow.

Intel® Data Analytics Acceleration Library (Intel® DAAL)

An Intel-optimized library that provides building blocks for all data analytics stages, from data preparation to data mining & machine learning

- Python, Java & C++ APIs
- Can be used with many platforms (Hadoop*, Spark*, R*, …) but not tied to any of them
- Flexible interface to connect to different data sources (CSV, SQL, HDFS, …)
- Windows*, Linux*, and OS X*

- Developed by same team as the industry-leading Intel® Math Kernel Library
- Open source, Free community-supported and commercial premium-supported options
- Also included in Parallel Studio XE suites

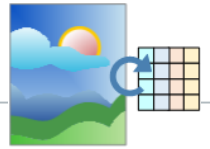
Intel® Data Analytics Acceleration Library
(Intel® DAAL)

An industry leading Intel® Architecture based data analytics acceleration library of fundamental algorithms covering all machine learning stages.

Pre-processing

(De-)Compression
Outlier Detection
Normalization

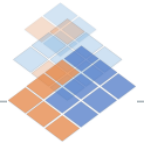
Transformation



PCA
Statistical moments
Variance matrix
Pp-QR, SVD, Cholesky
Apriori
Sorting

Analysis

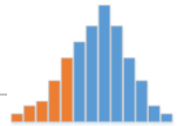
Modeling



Ridge linear regression
Naïve Bayes
SVM
Classifier boosting

Kmeans
EM GMM

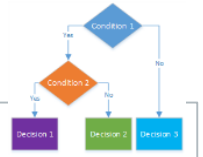
Validation



Collaborative filtering

Neural Networks

Decision Making



Intel DAAL Performance Scaling

Within a CPU core:

- SIMD vectorization: optimized for the latest instruction sets, AVX2, AVX512...
- Internally relies on sequential MKL

Scale to multi cores or many cores:

- Intel TBB threading

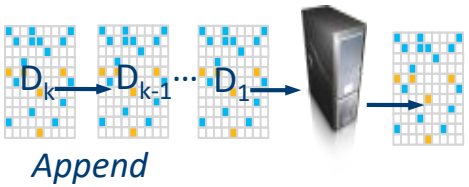
Scale to cluster:

- Distributed processing done by user application (MPI, MapReduce, etc.)
- DAAL provides
  - Data structures for partial and intermediate results
  - Functions to combine partial or intermediate results into global result

	Algorithms	Batch	Distributed	Online
Descriptive statistics	Low order moments	√	√	√
	Quantiles/sorting	√		
Statistical relationships	Correlation / Variance-Covariance	√	√	√
	(Cosine, Correlation) distance matrices	√		
Matrix decomposition	SVD	√	√	√
	Cholesky	√		
	QR	√	√	√
Regression	Linear/ridge regression	√	√	√
Classification	Multinomial Naïve Bayes	√	√	√
	SVM (two-class and multi-class)	√		
	Boosting (Ada, Brown, Logit)	√		
Unsupervised learning	Association rules mining (Apriori)	√		
	Anomaly detection (uni-/multi-variate)	√		
	PCA	√	√	√
	KMeans	√	√	
	EM for GMM	√		
Recommender systems	ALS	√	√	
Deep learning	Fully connected, convolution, normalization, activation layers, model, NN, optimization solvers,	√		

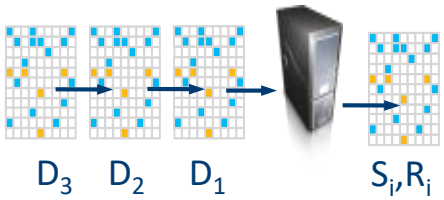
Processing modes

Batch Processing



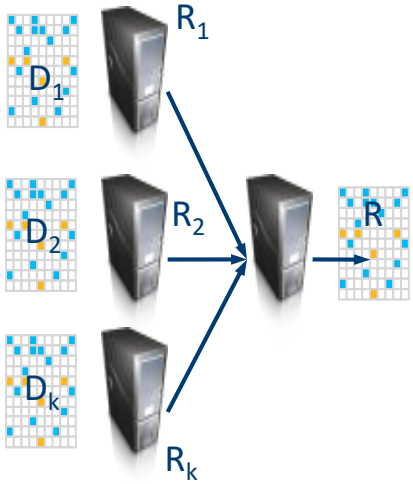
$R = F(D_1,\ldots,D_k)$

Online Processing



$S_{i+1} = T(S_i,D_i)$
$R_{i+1} = F(S_{i+1})$

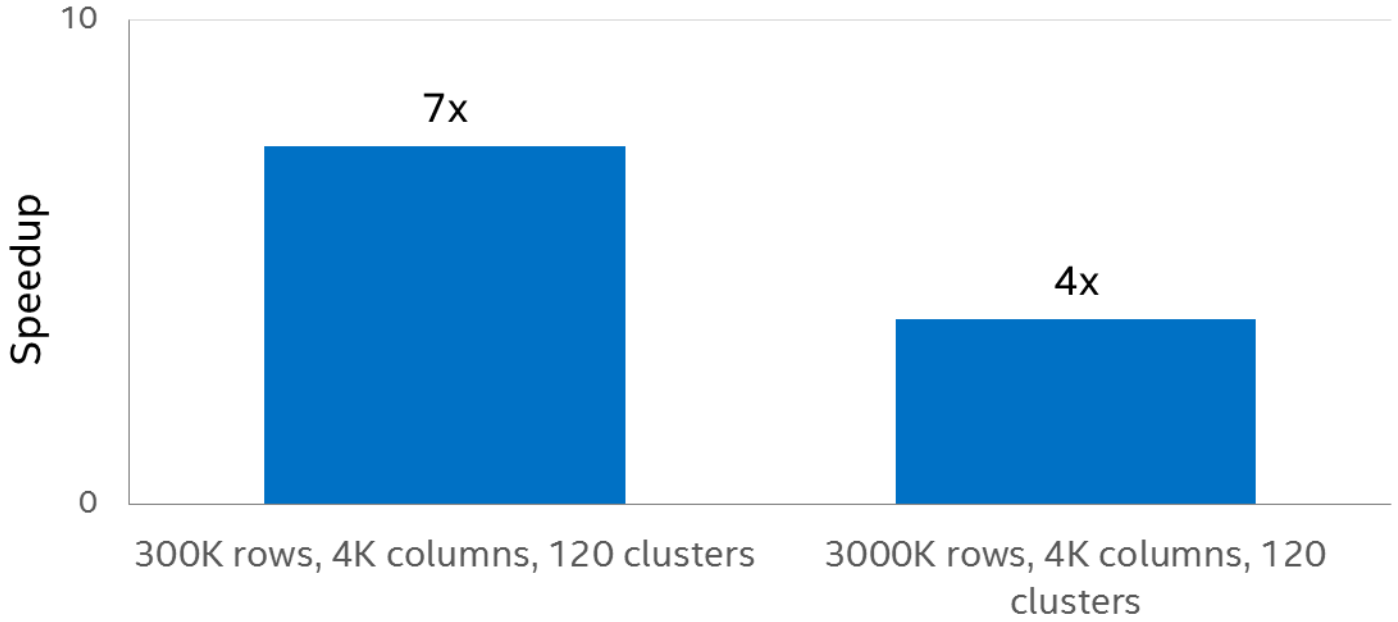
Distributed Processing



$R = F(R_1,\ldots,R_k)$

Intel® DAAL vs. Spark* Mllib

K-means Performance Comparison on Eight-node Cluster



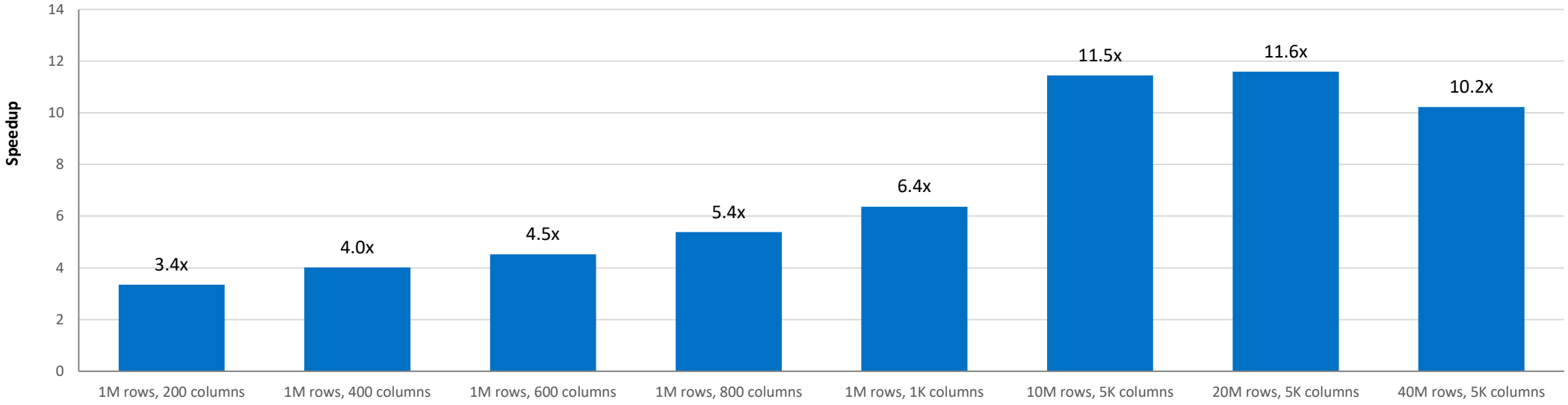
Configuration Info - Versions: Intel® Data Analytics Acceleration Library 2017, Spark 1.2; Hardware: Intel® Xeon® Processor E5-2699 v3, 2 Eighteen-core CPUs (45MB LLC, 2.3GHz), 128GB of RAM per node; Operating System: CentOS 6.6 x86_64.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Intel® DAAL Performance

PCA Performance Boosts
Using Intel® DAAL vs. Spark* MLlib on an Eight-node Cluster

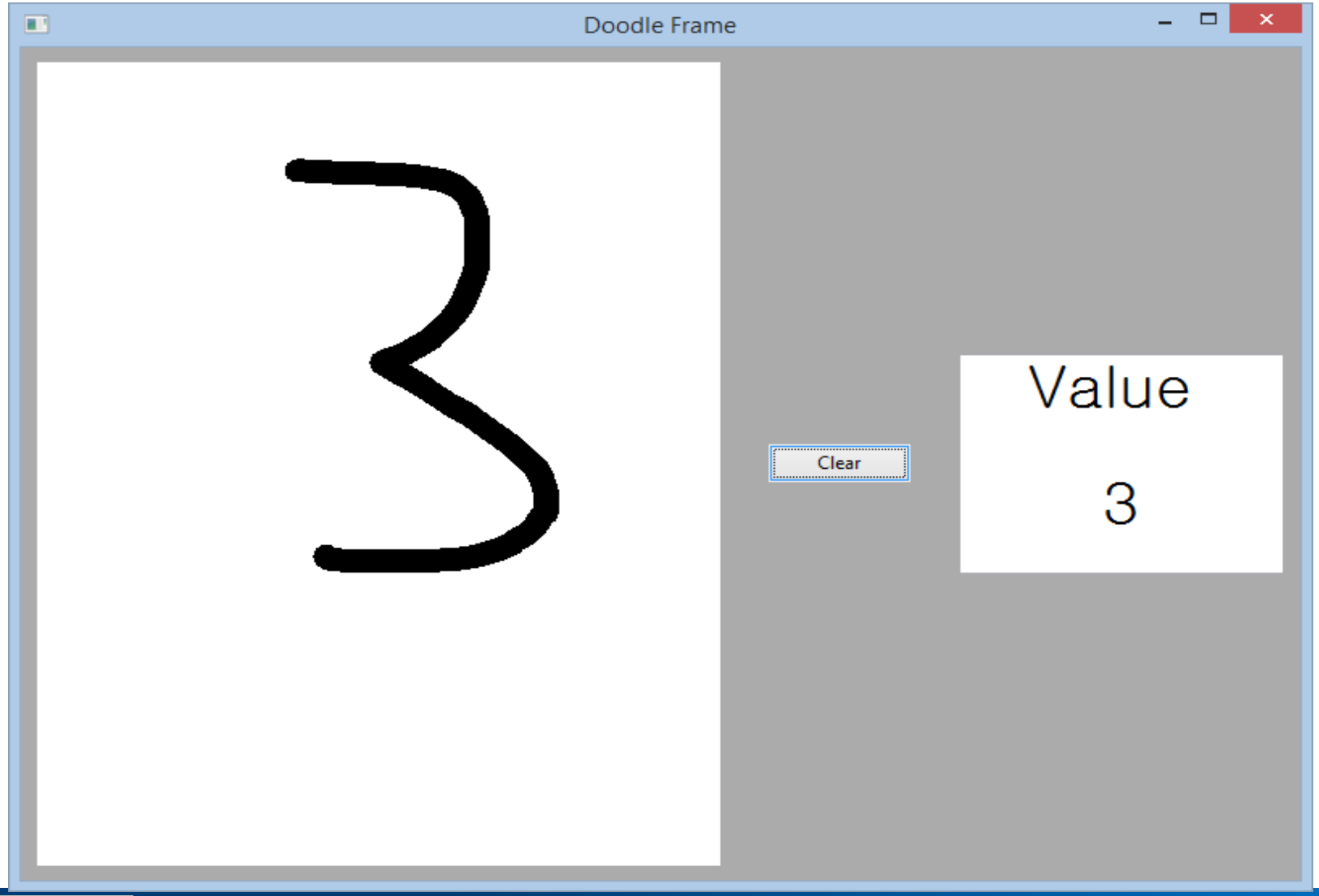


Configuration Info - Versions: Intel® Data Analytics Acceleration Library 2017, Spark 1.2; Hardware: Intel® Xeon® Processor E5-2699 v3, 2 Eighteen-core CPUs (45MB LLC, 2.3GHz), 128GB of RAM per node; Operating System: CentOS 6.6 x86_64.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Demo: Handwritten Digit Recognition



Handwritten Digit Recognition

Training multi-class SVM for 10 digits recognition.

3,823 pre-processed training data.

- available at http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits

99.6% accuracy with 1,797 test data from the same data provider.

Confusion matrix:

177.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000
0.000	181.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000
0.000	2.000	173.000	0.000	0.000	0.000	0.000	1.000	1.000	0.000
0.000	0.000	0.000	176.000	0.000	1.000	0.000	0.000	3.000	3.000
0.000	1.000	0.000	0.000	179.000	0.000	0.000	0.000	1.000	0.000
0.000	0.000	0.000	0.000	0.000	180.000	0.000	0.000	0.000	2.000
0.000	0.000	0.000	0.000	0.000	0.000	180.000	0.000	1.000	0.000
0.000	0.000	0.000	0.000	0.000	0.000	0.000	170.000	1.000	8.000
0.000	3.000	0.000	0.000	0.000	0.000	0.000	0.000	166.000	5.000
0.000	0.000	0.000	2.000	0.000	1.000	0.000	0.000	2.000	175.000

Average accuracy: 0.996
Error rate: 0.004
Micro precision: 0.978
Micro recall: 0.978
Micro F-score: 0.978
Macro precision: 0.978
Macro recall: 0.978
Macro F-score: 0.978

Training Handwritten Digits

```
void trainModel()
{
        /* Initialize FileDataSource<CSVFeatureManager> to retrieve input data from .csv file */
        FileDataSource<CSVFeatureManager> trainDataSource(trainDatasetFileName,
                DataSource::doAllocateNumericTable, DataSource::doDictionaryFromContext);

        /* Load data from the data files */
        trainDataSource.loadDataBlock(nTrainObservations);

        /* Create algorithm object for multi-class SVM training */
        multi_class_classifier::training::Batch<> algorithm;

        algorithm.parameter.nClasses = nClasses;
        algorithm.parameter.training = training;

        /* Pass training dataset and dependent values to the algorithm */
        algorithm.input.set(classifier::training::data,trainDataSource.getNumericTable());

        /* Build multi-class SVM model */
        algorithm.compute();

        /* Retrieve algorithm results */
        trainingResult = algorithm.getResult();

        /* Serialize the learned model into a disk file */
        ModelFileWriter writer("./model");
        writer.serializeToFile(trainingResult->get(classifier::training::model));
}
```

Create a numeric table

Create an alg. Obj.

Set input and parameters

Compute

Get result

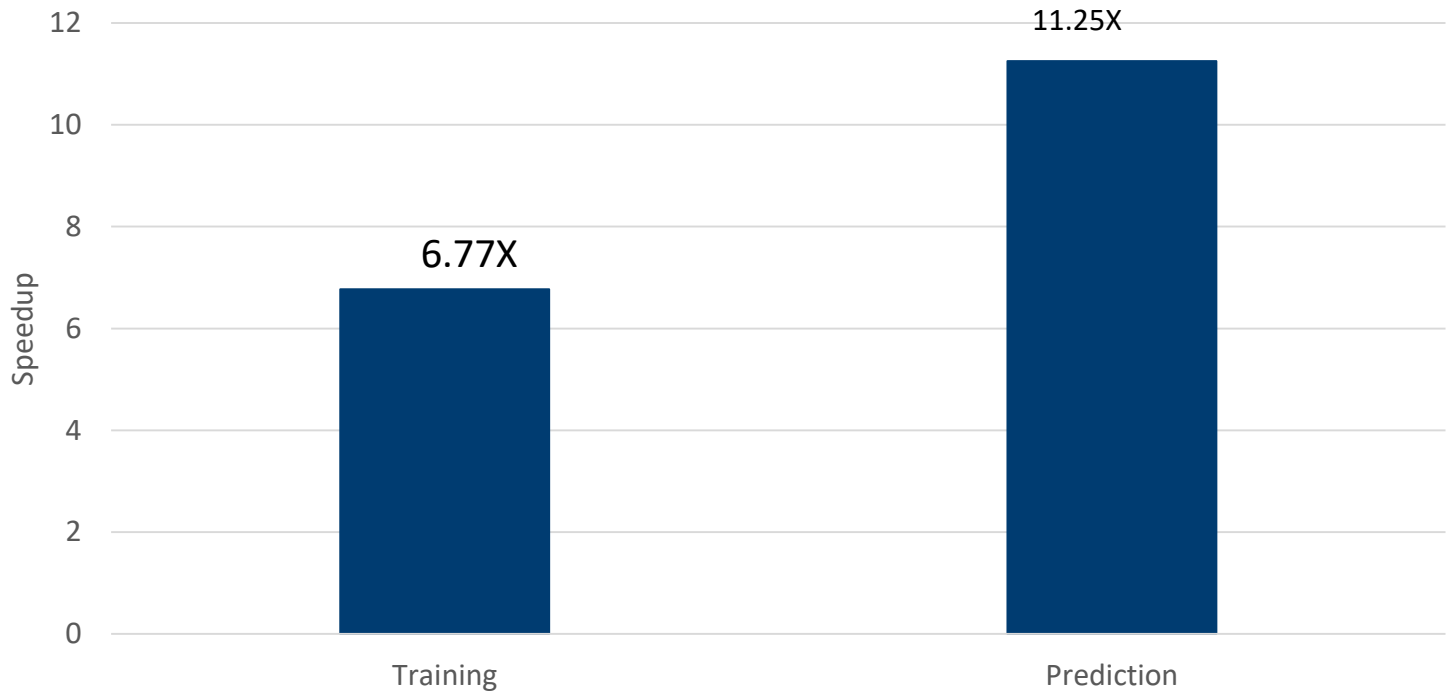
Serialize the learned model

Handwritten Digit Prediction

```
void testDigit()
{
      /* Initialize FileDataSource<CSVFeatureManager> to retrieve the test data
from .csv file */
      FileDataSource<CSVFeatureManager> testDataSource(testDatasetFileName,
            DataSource::doAllocateNumericTable, DataSource::doDictionaryFromContext);
      testDataSource.loadDataBlock(1);

      /* Create algorithm object for prediction of multi-class SVM values */
      multi_class_classifier::prediction::Batch<> algorithm;

      algorithm.parameter.prediction = prediction;

      /* Deserialize a model from a disk file */
      ModelFileReader reader("./model");
      services::SharedPtr<multi_class_classifier::Model> pModel(new
multi_class_classifier::Model());
      reader.deserializeFromFile(pModel);

      /* Pass testing dataset and trained model to the algorithm */
      algorithm.input.set(classifier::prediction::data,
testDataSource.getNumericTable());
      algorithm.input.set(classifier::prediction::model, pModel);

      /* Predict multi-class SVM values */
      algorithm.compute();

      /* Retrieve algorithm results */
      predictionResult = algorithm.getResult();

      /* Retrieve predicted labels */
      predictedLabels = predictionResult->get(classifier::prediction::prediction);
}
```

Deserialize learned model

SVM Performance Boosts Using Intel® DAAL vs. scikit-learn on Intel® CPU



Configuration Info - Versions: Intel® Data Analytics Acceleration Library 2016 U2, scikit-learn 0.16.1; Hardware: Intel Xeon E5-2680 v3 @ 2.50GHz, 24 cores, 30 MB L3 cache per CPU, 256 GB RAM; Operating System: Red Hat Enterprise Linux Server release 6.6, 64-bit.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. * Other brands and names are the property of their respective owners. Benchmark Source: Intel Corporation

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice revision #20110804 .

Resources

Intel® MKL Product Information

- https://software.intel.com/en-us/intel-mkl

Intel® MKL benchmarks

- https://software.intel.com/en-us/intel-mkl/benchmarks#

Intel® IPP Product Information

- https://software.intel.com/software/products/ipp

Intel® DAAL Product Information

- http://software.intel.com/en-us/intel-daal

Intel® DAAL Getting Started Guides

- https://software.intel.com/en-us/intel-daal-support/training

Legal Disclaimer & Optimization Notice

INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS”. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

Optimization Notice

Intel’s compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804