Software

BIGDL: A DISTRIBUTED DEEP LEARNING LIBRARY ON SPARK

Zhichao Li

Big Data Technology Team, Software and Service Group, Intel

Outline

- What's BigDL
- Why BigDL
- Inside BigDL
- What can BigDL do

WHAT IS BIGDL?

BigDL: Deep learning on Apache Spark*

BigDL open sourced on Dec 30, 2016

https://github.com/intel-analytics/BigDL

- Apache Spark*, MKL Acceleration, High perform

Rich function

- Scala/Java + Python
- AlexNet, GoogleNet, VGG, Faster R-CNN, SSD, Deep Speech, Recommendation…
- TensorBoard, Notebook, caffe/torch/tensorflow load/export…

Popularity

- Support from Cloud: Microsoft, Amazon, Cloudera, Databricks…

Basic Component

Tensor:

- ND-array data structure
- Generic data type
- Rich and fast math operations (powered by Intel MKL)

Layers

- 113+ layers (Conv, 3D Conv, Pooling, 3D Pooling, FC …)

Criterion

- 23+ criterions (DiceCoefficient, ClassNLL, CrossEntropy …)

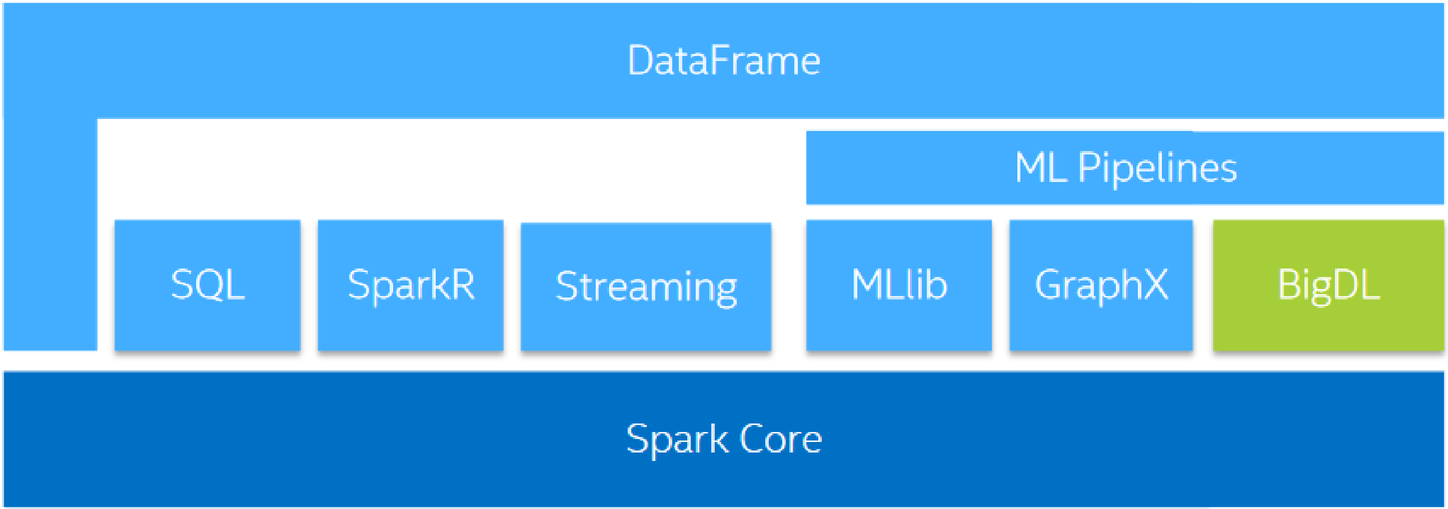
Optimization

- SGD, Adagrad
- **Community contribution**: Adam, Adadelta, RMSprop, Adamx

WHAT IS BIGDL?

BigDL is a distributed deep learning library for Apache Spark*

BigDL: implemented as a standalone library on Spark (Spark package)



WHY BIGDL?

Why BigDL

There're a lot of deep learning frameworks. Only list a part of them

neon

Caffe

theano

mxnet

CNTK

WHY BIGDL?

Production ML/DL system is **Complex and Distributed.**
Spark-based Deep Learning library is a natural fit

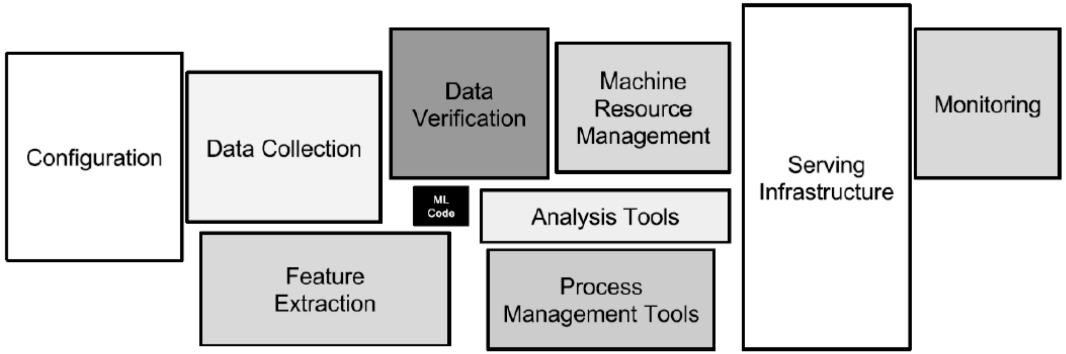
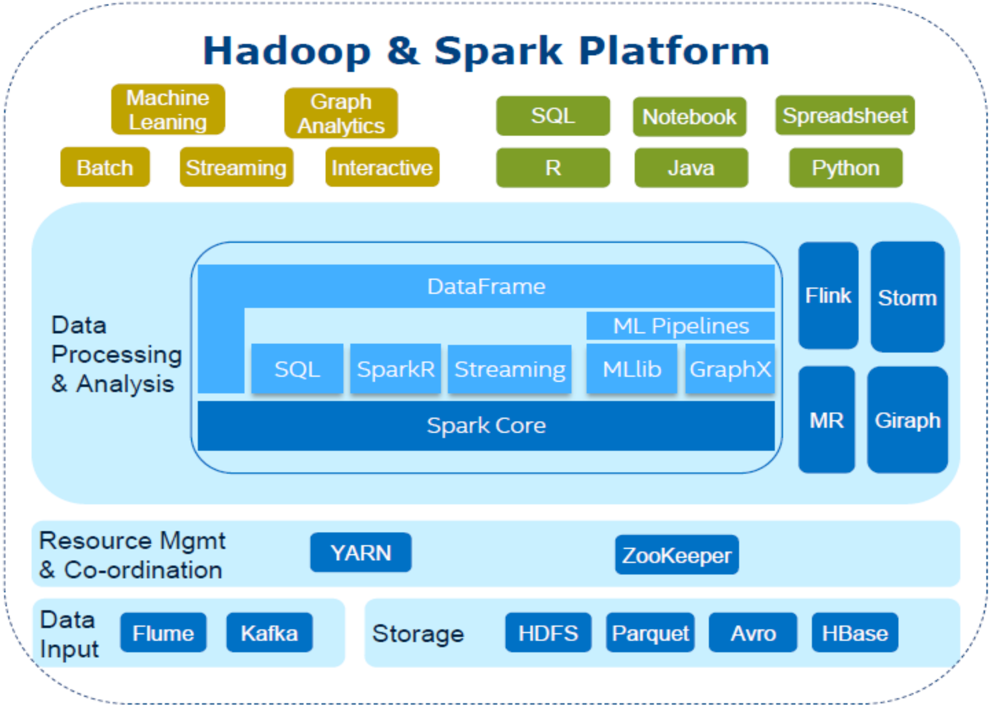


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

"Hidden Technical Debt in Machine Learning Systems",
Google, NIPS 2015 Paper

Why BigDL

BigDL: Run deep learning on Big Data platform

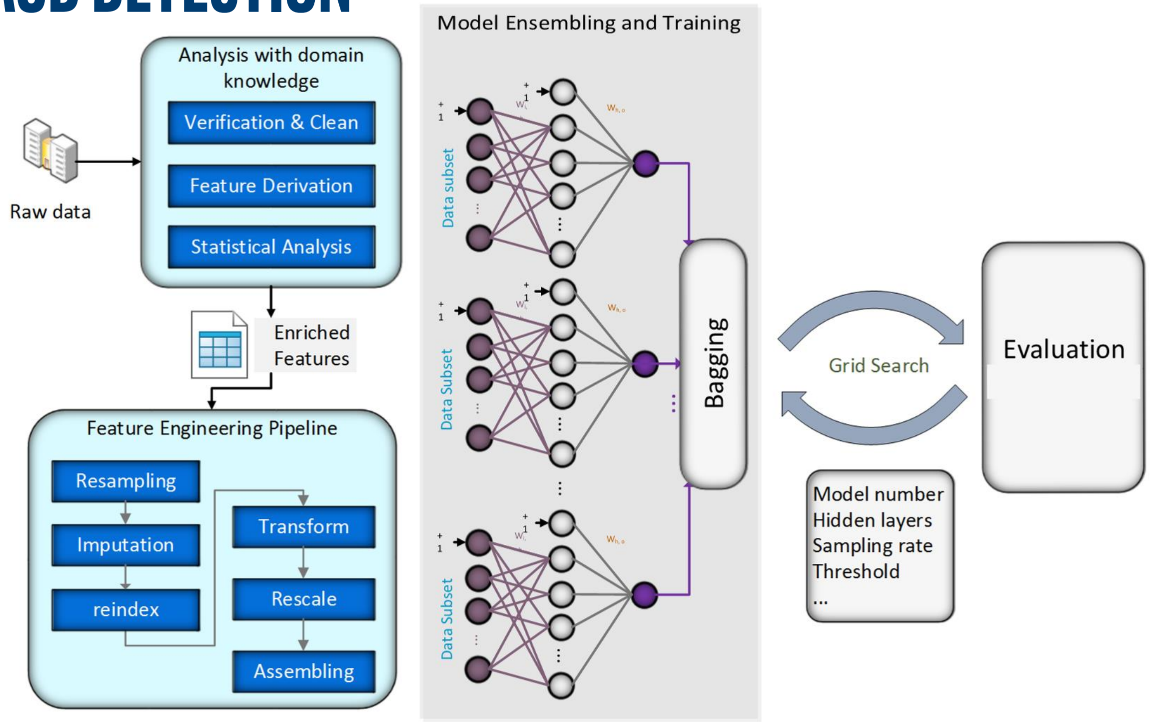


Outstanding features

- Massively distributed
- Fault tolerance
- Elasticity
- Dynamic resource sharing
- …

FINTECH: TRANSACTION FRAUD DETECTION

- Historical data is stored on Hive
- Data preprocessing with SparkSQL
- Spark ML pipeline for complex feature engineering
- Use multiple BigDL CNN models
- Use Sample+Bagging to solve unbalance problem
- Grid search for hyper parameter tuning



Powered by BigDL

BIGDL FEATURES

- Single node Xeon performance
  - Benchmarked to be best on Xeon E5-26XX v3 or E5-26XX v4
  - Orders of magnitude speedup vs. out-of-box open source Caffe, Torch
- Scaling-out
  - Efficiently scales out to 10s~100s of Xeon servers on Spark

Why BigDL

People use BigDL to build applications

- Large internet company
- Financial company
- Manufactory company
- Medical school

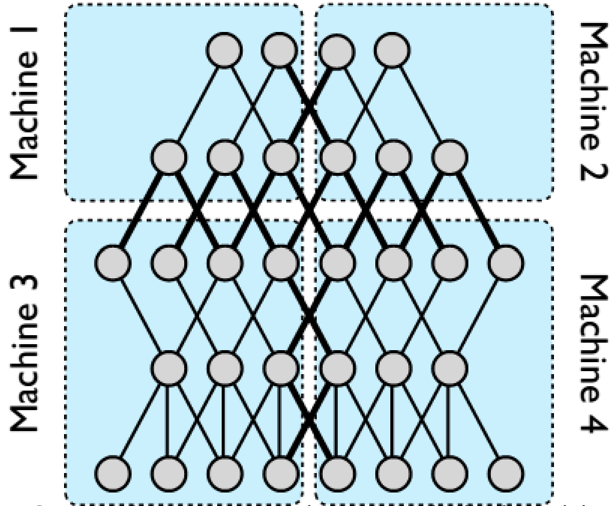
Image, Recommendation, Fraud detection, Audio, NLP

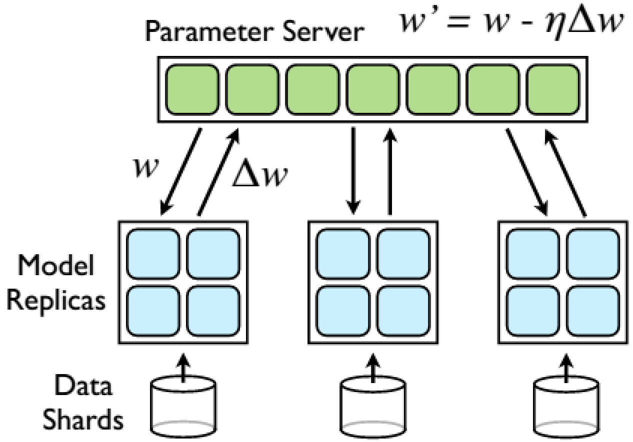
INSIDE BIGDL

Pattern

Model Parallelism

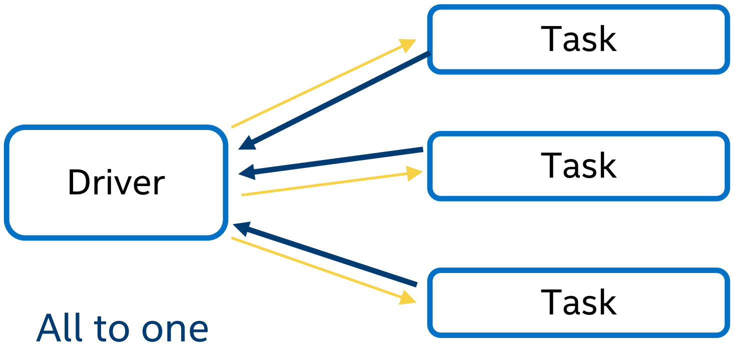
Data Parallelism



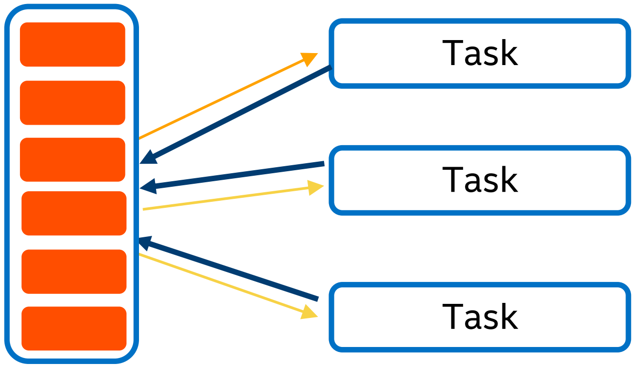


Source: Dean J, Corrado G, Monga R, et al. Large scale distributed deep networks[C]//Advances in neural information processing systems. 2012: 1223-1231.

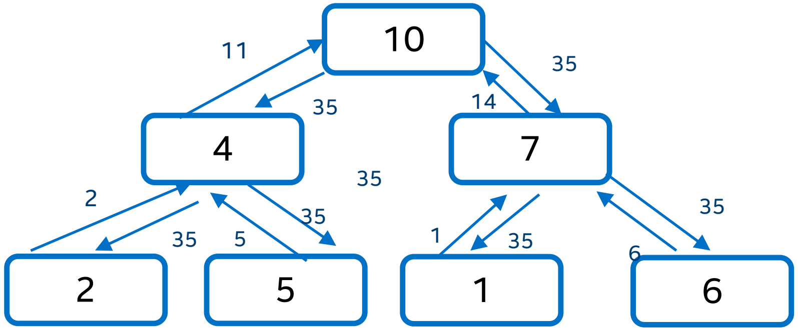
Communication Model



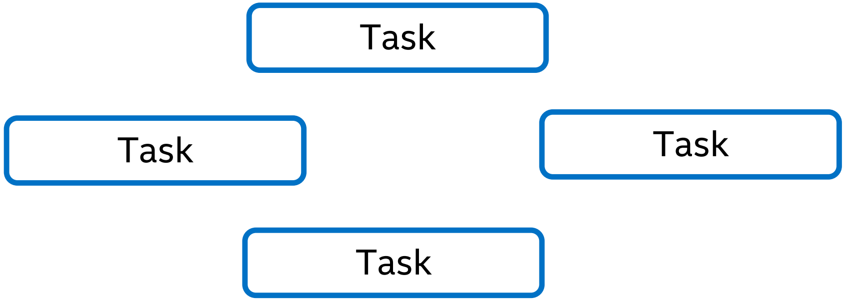
All to one



Parameter Server

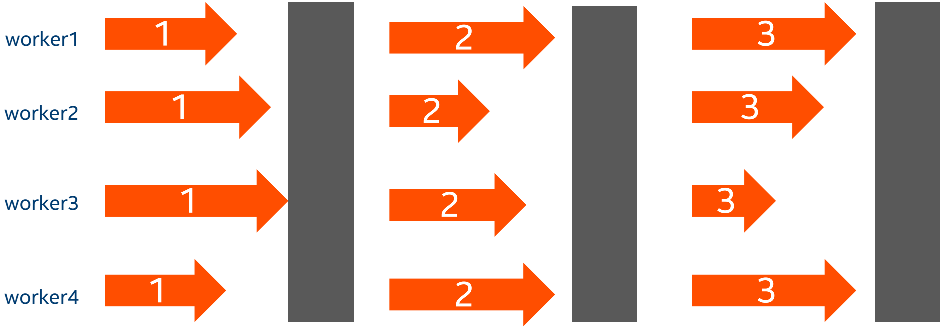


All reduce (tree aggregation)

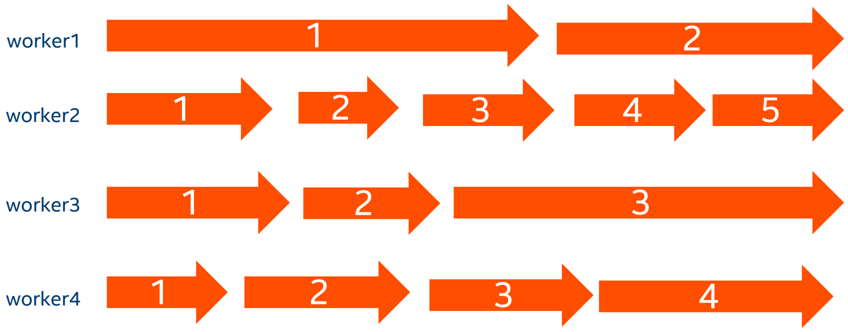


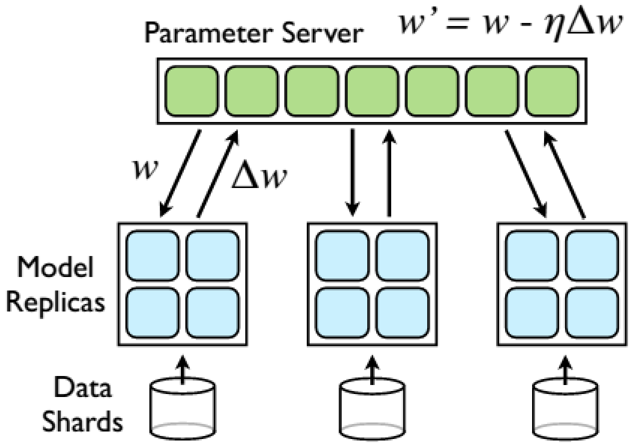
All reduce

Bulk Synchronous Parallel (BSP)



Asynchronous Synchronous Parallel (ASP)



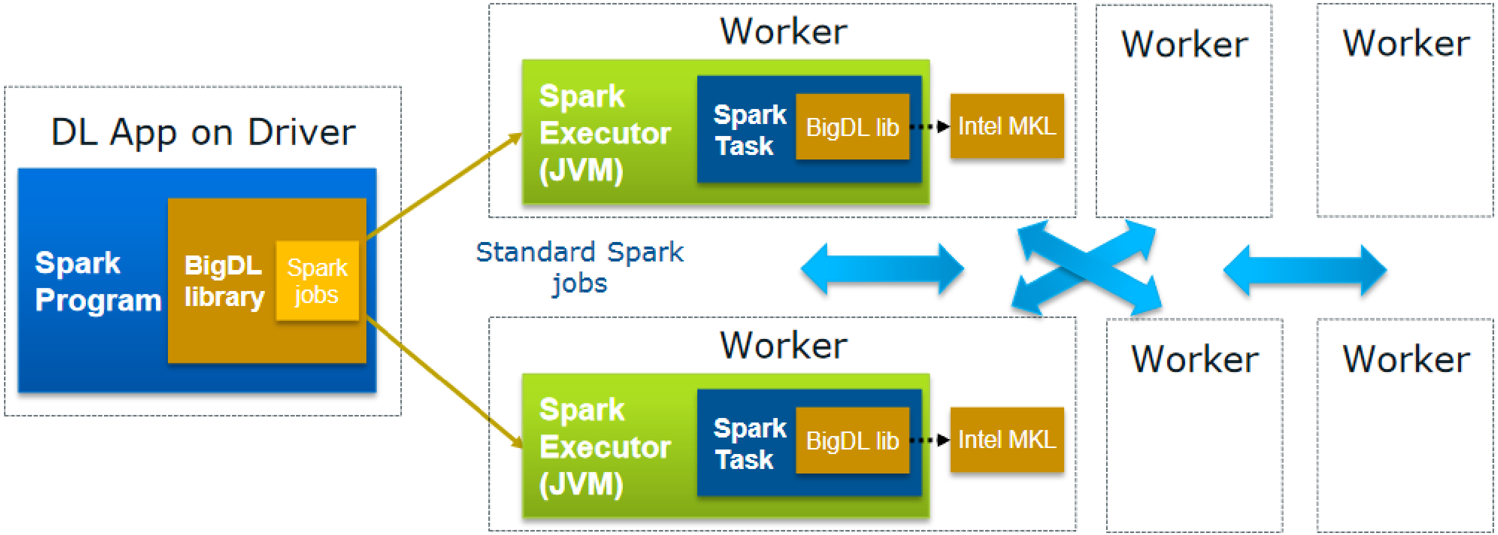


Source: Dean J, Corrado G, Monga R, et al. Large scale distributed deep networks[C]//Advances in neural information processing systems. 2012: 1223-1231.

BIGDL FEATURES

Distributed Deep learning applications on Apache Spark*

- No changes to the existing Hadoop/Spark clusters needed



PYTHON API SUPPORT

Based on PySpark, ***Python API*** in BigDL allows use of existing Python libs:

- Numpy
- Scipy
- Pandas
- Scikit-learn
- Matplotlib
- …

```
train_data = get_minst("train").map(
    normalizer(mnist.TRAIN_MEAN, mnist.TRAIN_STD))
test_data = get_minst("test").map(
    normalizer(mnist.TEST_MEAN, mnist.TEST_STD))
state = {"batchSize": int(options.batchSize),
         "learningRate": 0.01,
         "learningRateDecay": 0.0002}
optimizer = Optimizer(
    model=build_model(10),
    training_rdd=train_data,
    criterion=ClassNLLCriterion(),
    optim_method="SGD",
    state=state,
    end_trigger=MaxEpoch(100))
optimizer.setvalidation(
    batch_size=32,
    val_rdd=test_data,
    trigger=EveryEpoch(),
    val_method=["top1"]
)
optimizer.setcheckpoint(EveryEpoch(), "/tmp/lenet5/")
trained_model = optimizer.optimize()
```

JUPYTER NOTEBOOK SUPPORT

Running BigDL applications directly in Jupyter notebooks

✓ Share and Reproduce

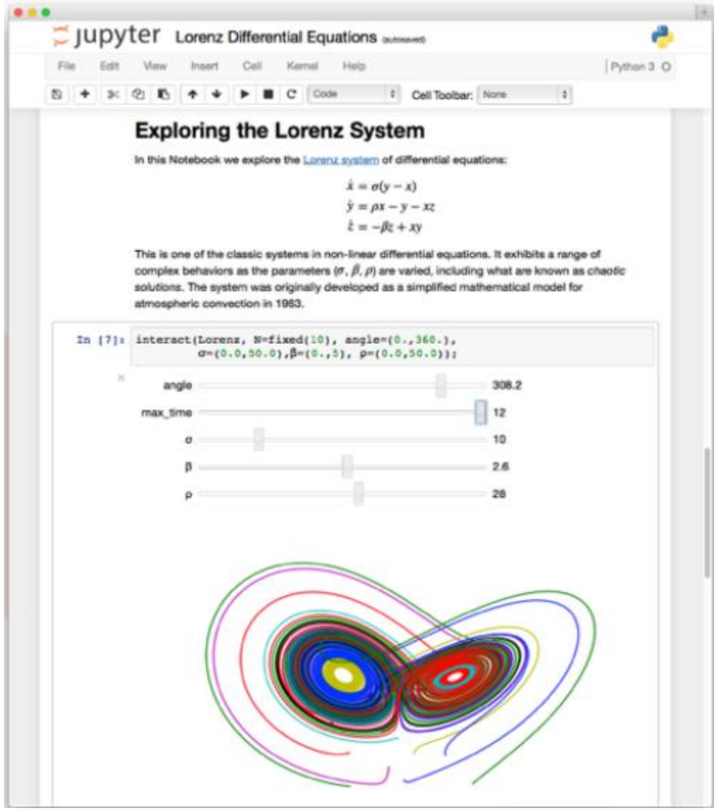
- Notebooks can be shared with others
- Easy to reproduce and track

✓ Rich Content

- Texts, images, videos, LaTeX and JavaScript
- Code can also produce rich contents

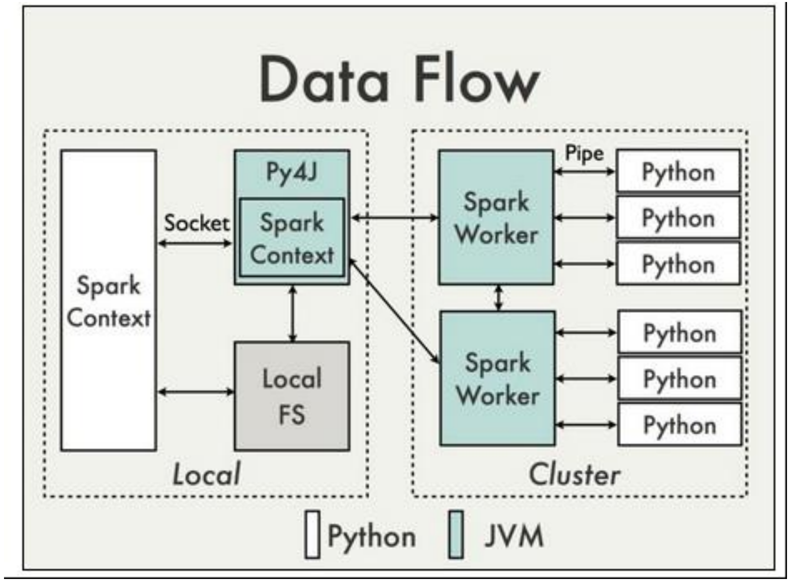
✓ Rich toolbox

- Apache Spark, from Python, R and Scala
- Pandas, scikit-learn, ggplot2, dplyr, etc



Python API

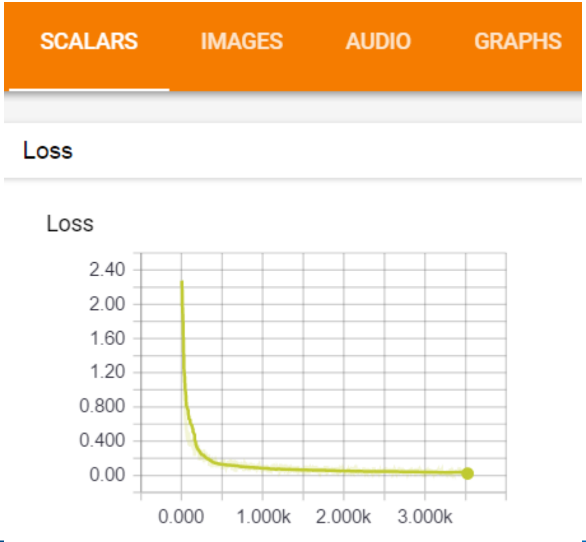
RDD[raw data] → Transform (python) → RDD[Samle(ndarray,ndarray)] → Train(python model)

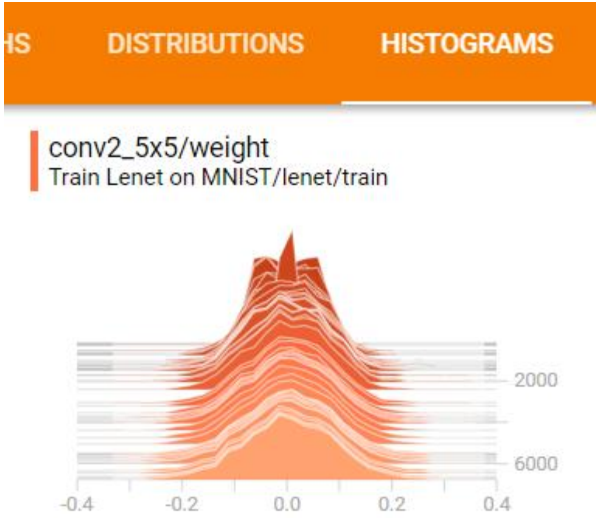


VISUALIZATION OF OPTIMIZATION PROCESS - TENSORBOARD

BigDL integration with TensorBoard

- TensorBoard is a suite of web applications from Google for visualizing and understanding deep learning applications

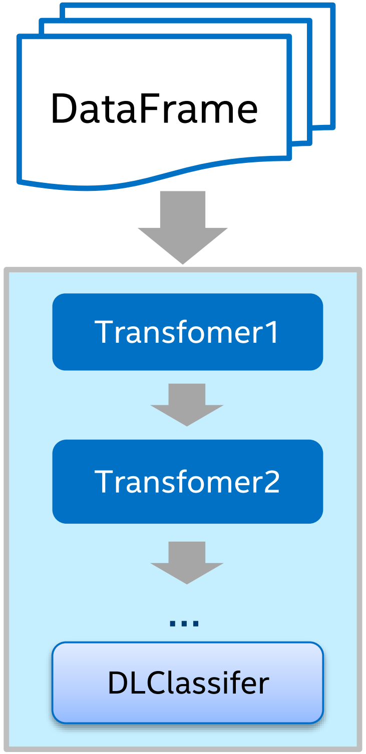




BIGDL INTEGRATION WITH SPARK ML

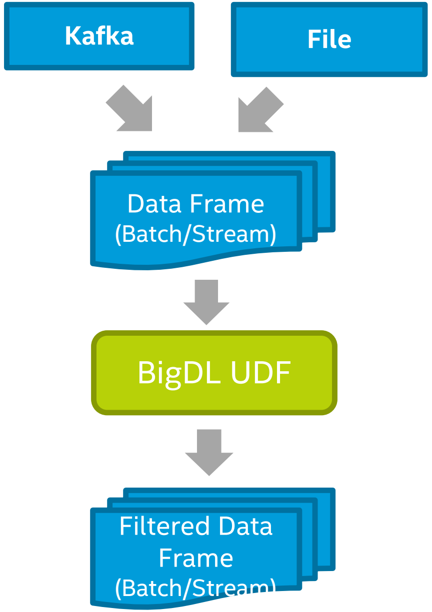
Integrates with Spark-ML Pipeline:

- Wrapper with Spark ML Transformer
- BigDL Plugs into Spark ML pipeline
- Support Spark v1.5/1.6/2.0/2.1



BIGDL FEATURES

Tight Integrations with Spark SQL, DataFrame and Structured Streaming



```
df.select($'image')
  .withColumn(
     "image_type",
ImgClassifier("image"))
  .filter($'image_type'
== 'dog')
```

*Image classification on ImageNet(http://www.image-net.org)

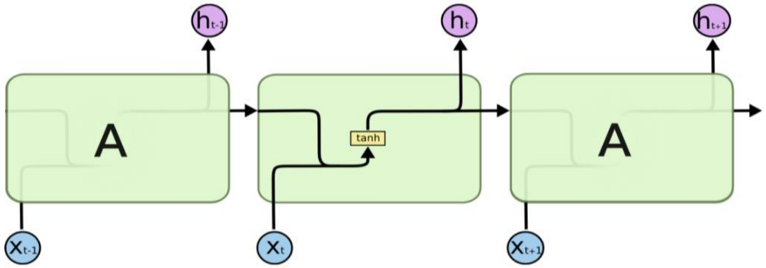
NATURAL LANGUAGE MODEL - RNN

RNN:

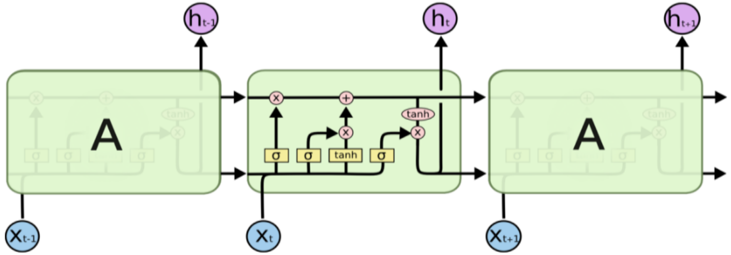
- Recurrent
- BiRecurrent

Cell:

- SimpleRNN
- LSTM
- GRU
- LSTM with peepholes



The repeating module in a standard RNN contains a single layer.



The repeating module in an LSTM contains four interacting layers.

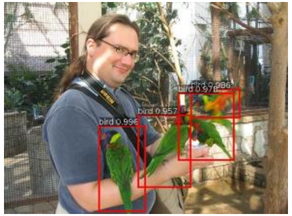
Source: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

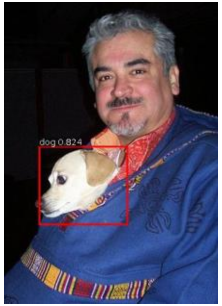
BigDL: design for big data

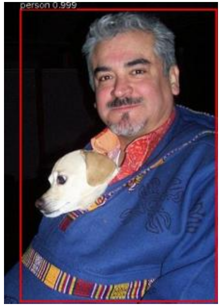
- **Standard Spark Programs (Python and Scala)**
- **Easy to deploy** on top of **Existing** Spark or Hadoop clusters.
- **Rich** deep learning support, **close integrate** with other big data work load
- Interact with other deep learning framework.
- **High performance** powered by Intel MKL and multi-threaded programming
- **Efficient scale-out** with an all-reduce communications on Spark

WHAT CAN BIGDL DO

OBJECT DETECTION ON PASCAL



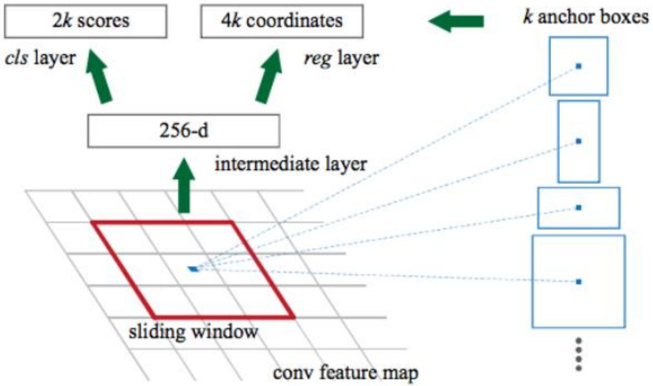




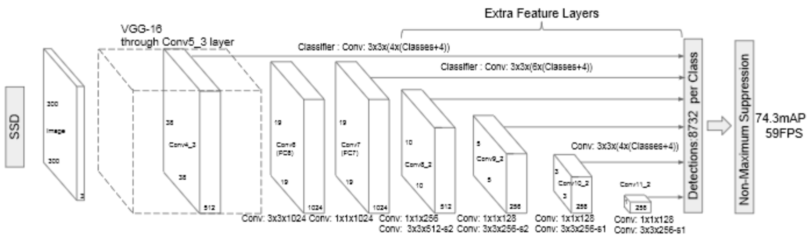
*(http://host.robots.ox.ac.uk/pascal/VOC/)

VISUAL RECOGNITION AND OBJECT DETECTION

Faster-RCNN



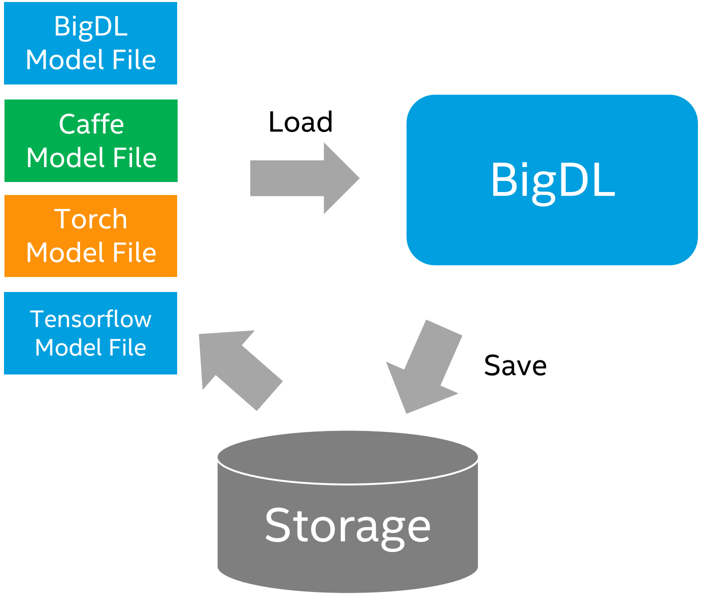
SSD: Single Shot MultiBox Detector



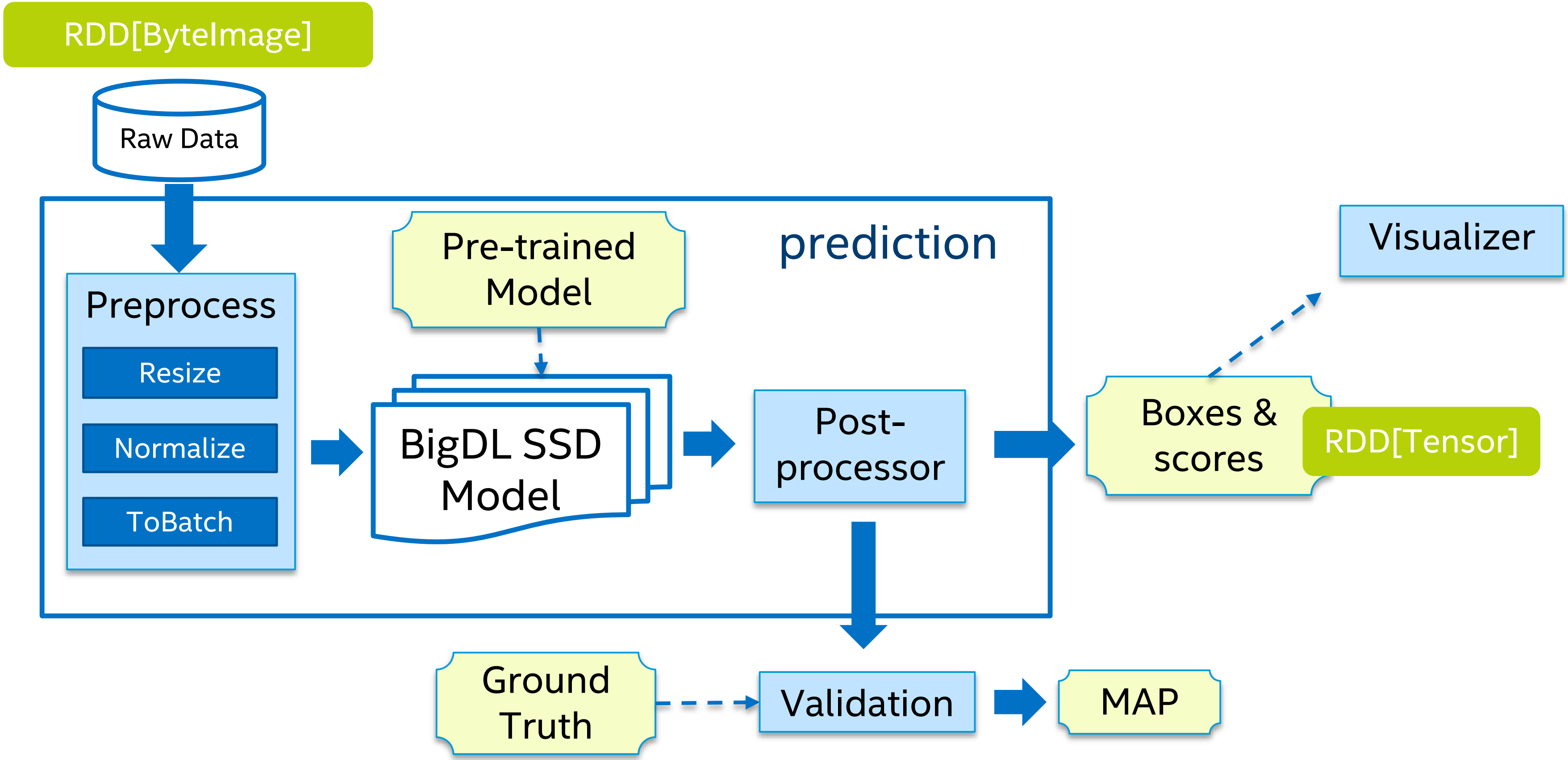
Model Persistent

- Model Snapshot
  - Long training work checkpoint
  - Model deployment and sharing
  - Fine-tune

- Caffe/Torch/Tensorflow Model Support
  - Model file load
  - Easy to migrate your caffe/torch/tensorflow work to Spark

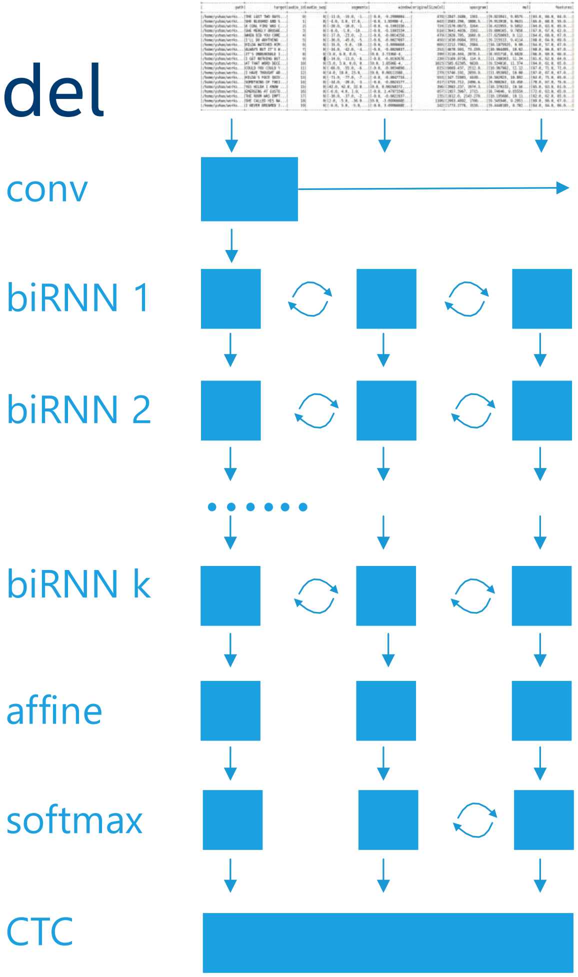


SSD Pipeline



Deep Speech 2 on BigDL: Model

```scala
val model = Sequential[T]()
  .add(conv)
  .add(ReLU[T]())
  .add(Squeeze(4))
  .add(brnn)
  .add(linear1)
  .add(HardTanhDS[T](0, 20, true))
  .add(linear2)
```



9 layers biRNN: >50 Million parameters

BigDL is an open source project

- Positive feedback from community
  - 1.7k+ stars,
  - Feature request from community(3D Conv, visualization ...)
  - PRs from community
  - Already see some adoptions

Documents

- Start with tutorials

  https://github.com/intel-analytics/BigDL-Tutorials/

- BigDL provide examples to help developer play with bigdl and start with popular models.
  - Vgg, Inception, AlexNet, ResNet, RNN
  - Text Classification, Image Classification, Load Torch/Caffe model

  https://github.com/intel-analytics/BigDL/wiki/Examples

- BigDL Out-of-box run scripts on AWS

  https://github.com/intel-analytics/BigDL/wiki/Running-on-EC2

BIGDL INSTALLATION ON MAJOR CLOUD FRAMEWORKS.

- "Apache Spark BigDL on Databricks"
  https://databricks.com/blog/2017/02/09/intels-bigdl-databricks.html
- "BigDL on Cloudera's CDH Data Science Virtual Machine"
  http://blog.cloudera.com/blog/2017/04/bigdl-on-cdh-and-cloudera-data-science-workbench/
- "How to use BigDL on Apache Spark for Azure HDInsight"
  https://blogs.msdn.microsoft.com/azuredatalake/2017/03/17/how-to-use-bigdl-on-apache-spark-for-azure-hdinsight/
- "BigDL on Microsoft's Data Science Virtual Machine"
  Coming soon

BIGDL INSTALLATION ON MAJOR CLOUD FRAMEWORKS - 2.

- “Apache Spark BigDL on AWS”
  https://github.com/intel-analytics/BigDL/wiki/Running-on-EC2
- “Apache Spark BigDL for E-MapReduce on Ali Cloud ”
  https://yq.aliyun.com/articles/73347

BIGDL ON GITHUB

HTTPS://GITHUB.COM/INTEL-ANALYTICS/BIGDL

BIGDL COMMUNITY

Join Our Mail List

bigdl-user-group+subscribe@googlegroups.com

Report Bugs And Create Feature Request

https://github.com/intel-analytics/BigDL/issues

Legal Disclaimer

Risk Factors

The above statements and any others in this document that refer to plans and expectations for the first quarter, the year and the future are forward-looking statements that involve a number of risks and uncertainties. Words such as “anticipates,” “expects,” “intends,” “plans,” “believes,” “seeks,” “estimates,” “may,” “will,” “should” and their variations identify forward-looking statements. Statements that refer to or are based on projections, uncertain events or assumptions also identify forward-looking statements. Many factors could affect Intel’s actual results, and variances from Intel’s current expectations regarding such factors could cause actual results to differ materially from those expressed in these forward-looking statements. Intel presently considers the following to be the important factors that could cause actual results to differ materially from the company’s expectations. Demand could be different from Intel's expectations due to factors including changes in business and economic conditions; customer acceptance of Intel’s and competitors’ products; supply constraints and other disruptions affecting customers; changes in customer order patterns including order cancellations; and changes in the level of inventory at customers. Uncertainty in global economic and financial conditions poses a risk that consumers and businesses may defer purchases in response to negative financial events, which could negatively affect product demand and other related matters. Intel operates in intensely competitive industries that are characterized by a high percentage of costs that are fixed or difficult to reduce in the short term and product demand that is highly variable and difficult to forecast. Revenue and the gross margin percentage are affected by the timing of Intel product introductions and the demand for and market acceptance of Intel's products; actions taken by Intel's competitors, including product offerings and introductions, marketing programs and pricing pressures and Intel’s response to such actions; and Intel’s ability to respond quickly to technological developments and to incorporate new features into its products. The gross margin percentage could vary significantly from expectations based on capacity utilization; variations in inventory valuation, including variations related to the timing of qualifying products for sale; changes in revenue levels; segment product mix; the timing and execution of the manufacturing ramp and associated costs; start-up costs; excess or obsolete inventory; changes in unit costs; defects or disruptions in the supply of materials or resources; product manufacturing quality/yields; and impairments of long-lived assets, including manufacturing, assembly/test and intangible assets. Intel's results could be affected by adverse economic, social, political and physical/infrastructure conditions in countries where Intel, its customers or its suppliers operate, including military conflict and other security risks, natural disasters, infrastructure disruptions, health concerns and fluctuations in currency exchange rates. Expenses, particularly certain marketing and compensation expenses, as well as restructuring and asset impairment charges, vary depending on the level of demand for Intel's products and the level of revenue and profits. Intel’s results could be affected by the timing of closing of acquisitions and divestitures. Intel's results could be affected by adverse effects associated with product defects and errata (deviations from published specifications), and by litigation or regulatory matters involving intellectual property, stockholder, consumer, antitrust, disclosure and other issues, such as the litigation and regulatory matters described in Intel's SEC reports. An unfavorable ruling could include monetary damages or an injunction prohibiting Intel from manufacturing or selling one or more products, precluding particular business practices, impacting Intel’s ability to design its products, or requiring other remedies such as compulsory licensing of intellectual property. A detailed discussion of these and other factors that could affect Intel’s results is included in Intel’s SEC filings, including the company’s most recent reports on Form 10-Q, Form 10-K and earnings release.

Software