Building Unified Big Data Analytics and AI Pipelines

Jason Dai
Senior Principal Engineer

2019/07/22

Overview

AI on Apache Spark™

BigDL

Distributed, High-Performance
Deep Learning Framework
for Apache Spark*

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

Analytics Zoo

Analytics + AI Platform

Distributed TensorFlow*, Keras*, PyTorch* and BigDL on Apache Spark*

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

Accelerating Data Analytics + AI Solutions At Scale

Real-World ML/DL Applications Are Complex Data Analytics Pipelines

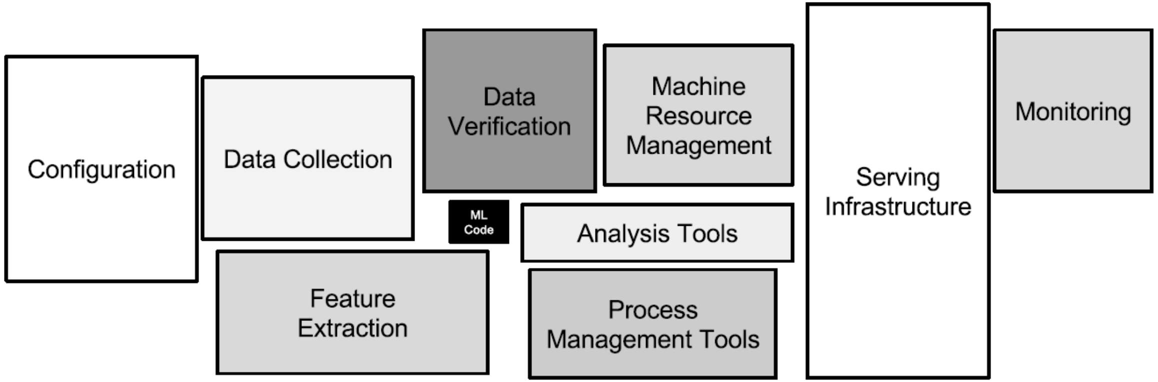


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

"Hidden Technical Debt in Machine Learning Systems", Sculley et al., Google, NIPS 2015 Paper

End-to-End Big Data Analytics and AI Pipeline

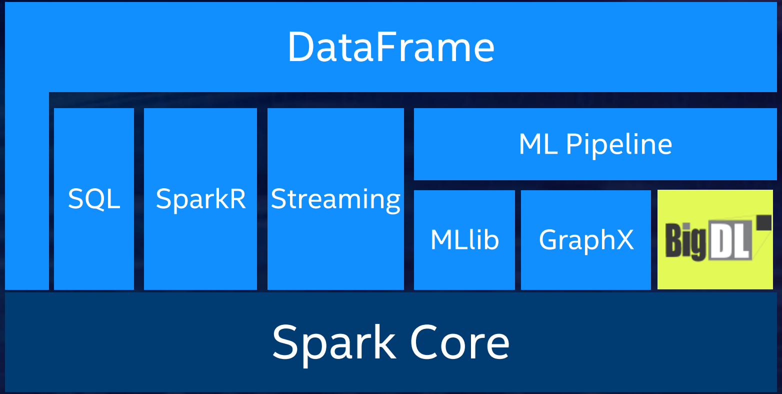
Seamless Scaling from Laptop to Production with Analytics Zoo

- **"Zero" code change from laptop to distributed cluster**
- **Directly access production data (Hadoop/Hive/HBase) without data copy**
- **Easily prototype the end-to-end pipeline**
- **Seamlessly deployed on production big data clusters**

BigDL

Bringing Deep Learning To Big Data Platform

- Distributed deep learning framework for Apache Spark
- Make deep learning more accessible to big data users and data scientists
  - Write deep learning applications as ***standard Spark programs***
  - Run on existing Spark/Hadoop clusters (***no changes needed***)
- Feature parity with popular deep learning frameworks
  - E.g., Caffe, Torch, Tensorflow, etc.
- High performance (on CPU)
  - Powered by Intel MKL and multi-threaded programming
- Efficient scale-out
  - Leveraging Spark for distributed training & inference



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

https://bigdl-project.github.io/

Analytics Zoo

End-to-End, Unified Analytics + AI Platform for Big Data

Use case	Recommendation	Anomaly Detection	Text Classification	Text Matching	
Model	Image Classification	Object Detection	Seq2Seq	Transformer	BERT
Feature Engineering	image	3D image	text	Time series	

High Level Pipelines

- tfpark: Distributed TF on Spark
- Distributed Keras w/ autograd on Spark
- nnframes: Spark Dataframes & ML Pipelines for Deep Learning
- Distributed Model Serving (batch, streaming & online)

Backend/ Library

TensorFlow | Keras | BigDL | NLP Architect | Apache Spark | Apache Flink

MKLDNN | OpenVINO | Intel® Optane™ DCPMM | DL Boost (VNNI)

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

Analytics Zoo

End-to-End, Unified Analytics + AI Platform for Big Data

Build end-to-end deep learning applications for big data

- Distributed *TensorFlow* on Spark
- *Keras* API (with autograd & transfer learning support) on Spark
- *nnframes*: native DL support for Spark DataFrames and ML Pipelines

Productionize deep learning applications for big data at scale

- Plain Java/Python *model serving* APIs (w/ OpenVINO support)
- Support Web Services, Spark, Flink, Storm, Kafka, etc.

Out-of-the-box solutions

- Built-in deep learning *models*, *feature engineering* operations, and reference *use cases*

Distributed TF & Keras on Spark

- **Data wrangling and analysis using PySpark**
- **Deep learning model development using TensorFlow or Keras**
- **Distributed training / inference on Spark**

Write TensorFlow code inline in PySpark program

```
#pyspark code
train_rdd = spark.hadoopFile(…).map(…)
dataset = TFDataset.from_rdd(train_rdd,…)

#tensorflow code
import tensorflow as tf
slim = tf.contrib.slim
images, labels = dataset.tensors
with slim.arg_scope(lenet.lenet_arg_scope()):
   logits, end_points = lenet.lenet(images, …)
loss = tf.reduce_mean( \
   tf.losses.sparse_softmax_cross_entropy( \
   logits=logits, labels=labels))

#distributed training on Spark
optimizer = TFOptimizer.from_loss(loss, Adam(…))
optimizer.optimize(end_trigger=MaxEpoch(5))
```

Spark Dataframe & ML Pipeline for DL

```
#Spark dataframe transformations
parquetfile = spark.read.parquet(…)
train_df = parquetfile.withColumn(…)

#Keras API
model = Sequential()
          .add(Convolution2D(32, 3, 3, activation='relu', input_shape=…)) \
          .add(MaxPooling2D(pool_size=(2, 2))) \
          .add(Flatten()).add(Dense(10, activation='softmax')))

#Spark ML pipeline
Estimater = NNEstimater(model, CrossEntropyCriterion()) \
                .setLearningRate(0.003).setBatchSize(40).setMaxEpoch(5) \
                .setFeaturesCol("image")
nnModel = estimater.fit(train_df)
```

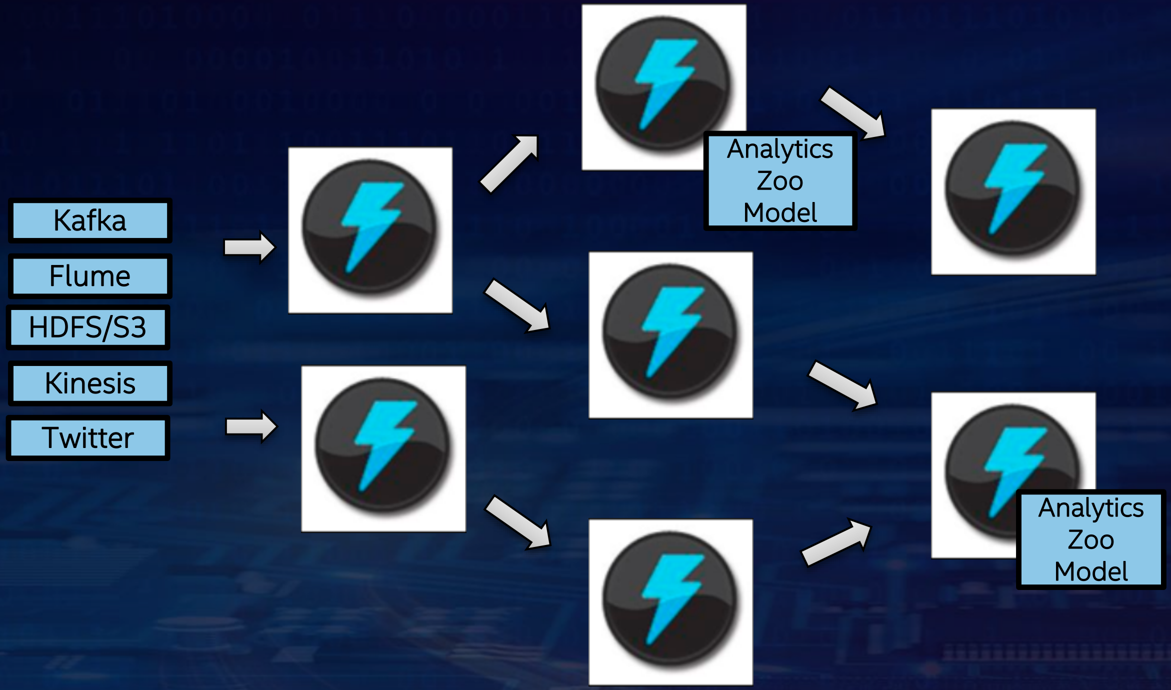
Spark Dataframe & ML Pipeline for DL

```
#Spark dataframe transformations
parquetfile = spark.read.parquet(…)
train_df = parquetfile.withColumn(…)

#Keras API
model = Sequential()
          .add(Convolution2D(32, 3, 3, activation='relu', input_shape=…)) \
          .add(MaxPooling2D(pool_size=(2, 2))) \
          .add(Flatten()).add(Dense(10, activation='softmax')))

#Spark ML pipeline
Estimater = NNEstimater(model, CrossEntropyCriterion()) \
                .setLearningRate(0.003).setBatchSize(40).setMaxEpoch(5) \
                .setFeaturesCol("image")
nnModel = estimater.fit(train_df)
```

Distributed Model Serving



Distributed model serving in Web Service, Flink, Kafka, Storm, etc.

- **Plain Java or Python API, with OpenVINO and DL Boost (VNNI) support**

OpenVINO Support for Model Serving

```python
from zoo.common.nncontext import init_nncontext
from zoo.feature.image import ImageSet
from zoo.pipeline.inference import InferenceModel

sc = init_nncontext("OpenVINO Object Detection Inference Example")
images = ImageSet.read(options.img_path, sc,
                       resize_height=600, resize_width=600).get_image().collect()
input_data = np.concatenate([image.reshape((1, 1) + image.shape) for image in images], axis=0)

model = InferenceModel()
model.load_tf(options.model_path, backend="openvino", model_type=options.model_type)
predictions = model.predict(input_data)

# Print the detection result of the first image.
print(predictions[0])
```

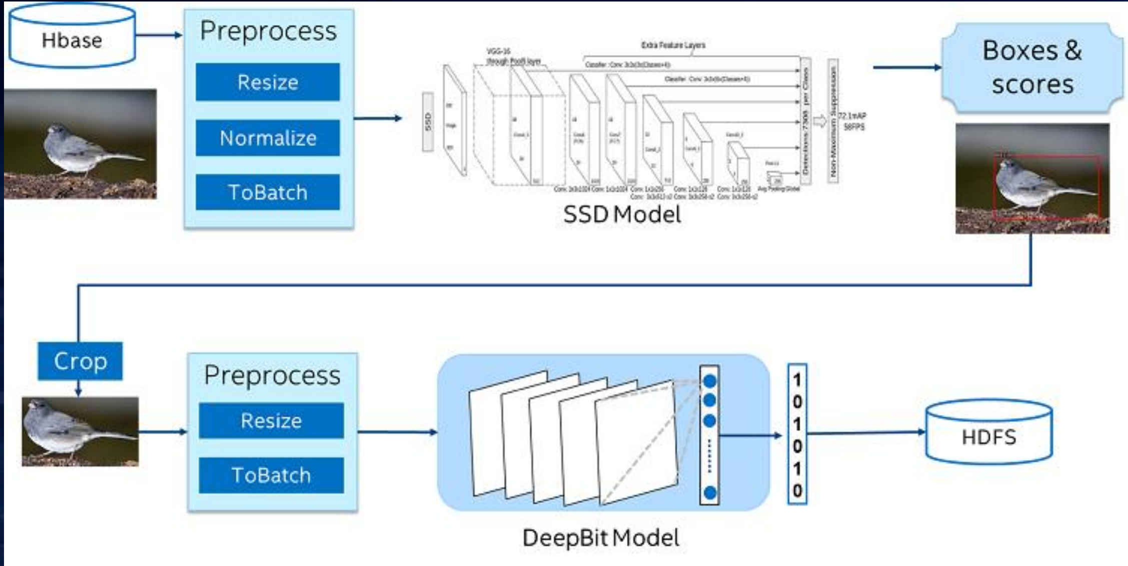
Transparently support OpenVINO in model serving,
which deliver a significant boost for inference speed

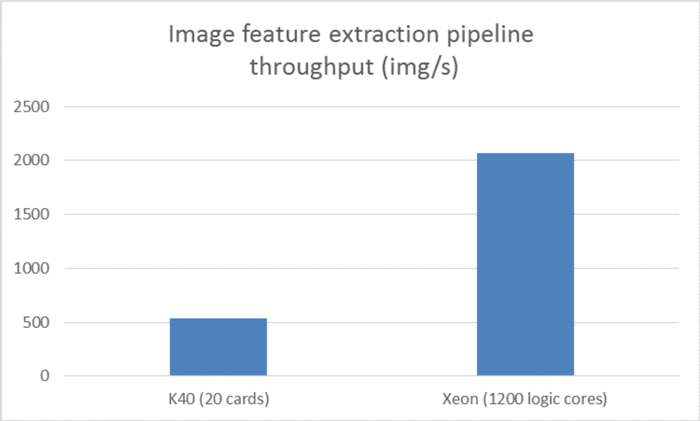
Upcoming Analytics Zoo 0.6 Release

- Distributed PyTorch on Spark
- Ray on Spark
  - Run Ray programs directly on standard Hadoop/YARN clusters
- AutoML support
  - Automatic feature generation, model selection and hyper-parameter tuning for *time series prediction*
- Cluster serving
  - Distributed, real-time (streaming) model serving with simple pub-sub interface

Use Cases

Object Detection and Image Feature Extraction at JD.com

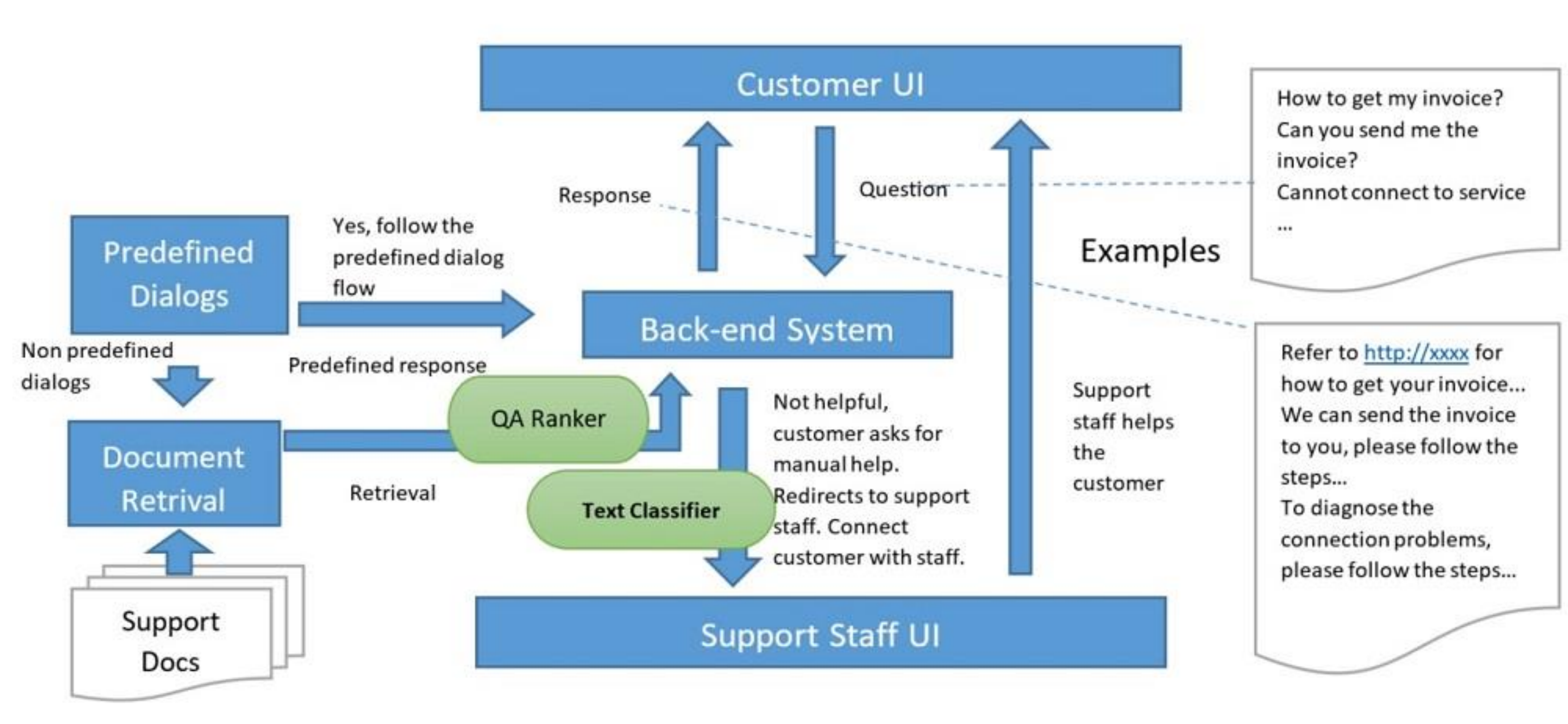




- Reuse existing Hadoop/Spark clusters for deep learning with no changes (image search, IP protection, etc.)
- Efficiently scale out on Spark with superior performance (***3.83x*** speed-up vs. GPU severs) as benchmarked by JD

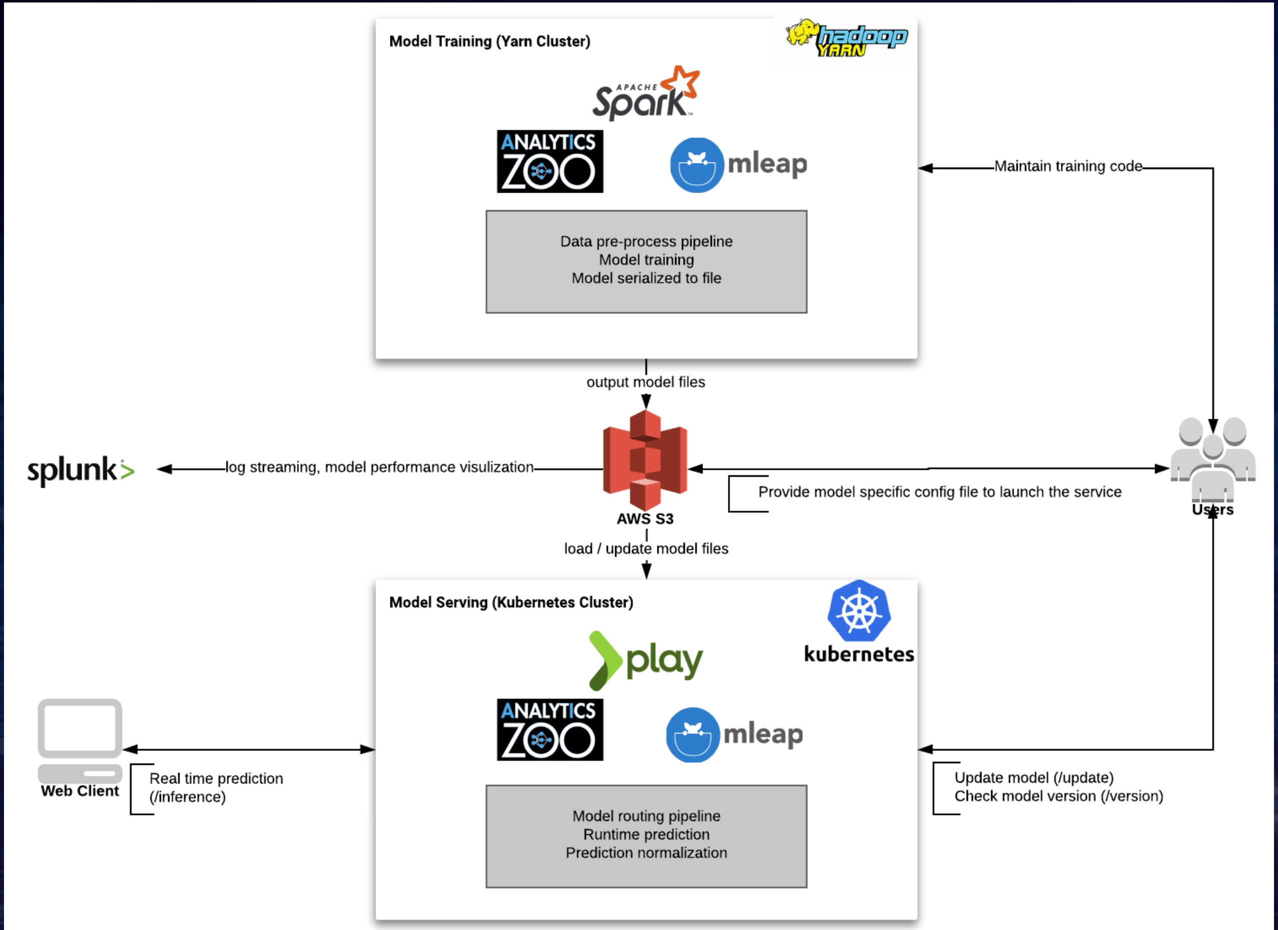
  http://mp.weixin.qq.com/s/xUCkzbHK4K06-v5qUsaNQQ
  https://software.intel.com/en-us/articles/building-large-scale-image-feature-extraction-with-bigdl-at-jdcom

NLP Based Customer Service Chatbot for Microsoft Azure



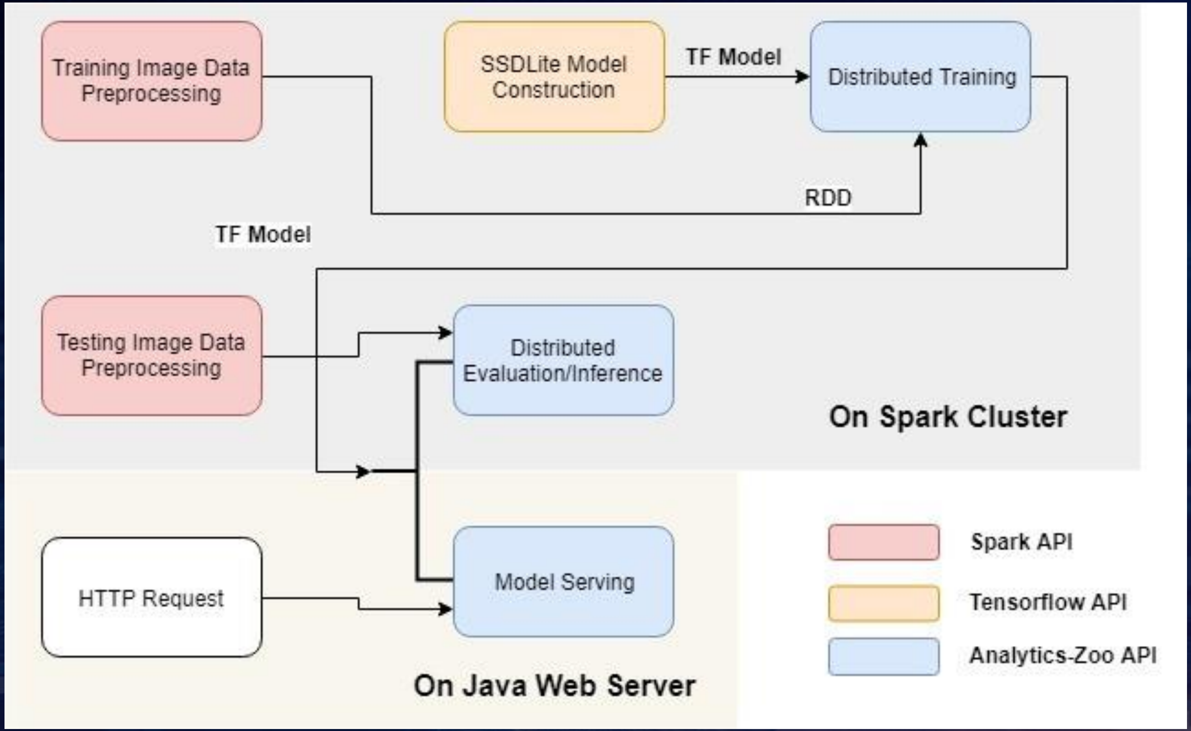
https://software.intel.com/en-us/articles/use-analytics-zoo-to-inject-ai-into-customer-service-platforms-on-microsoft-azure-part-1
https://www.infoq.com/articles/analytics-zoo-qa-module/

Product Recommendations in Office Depot



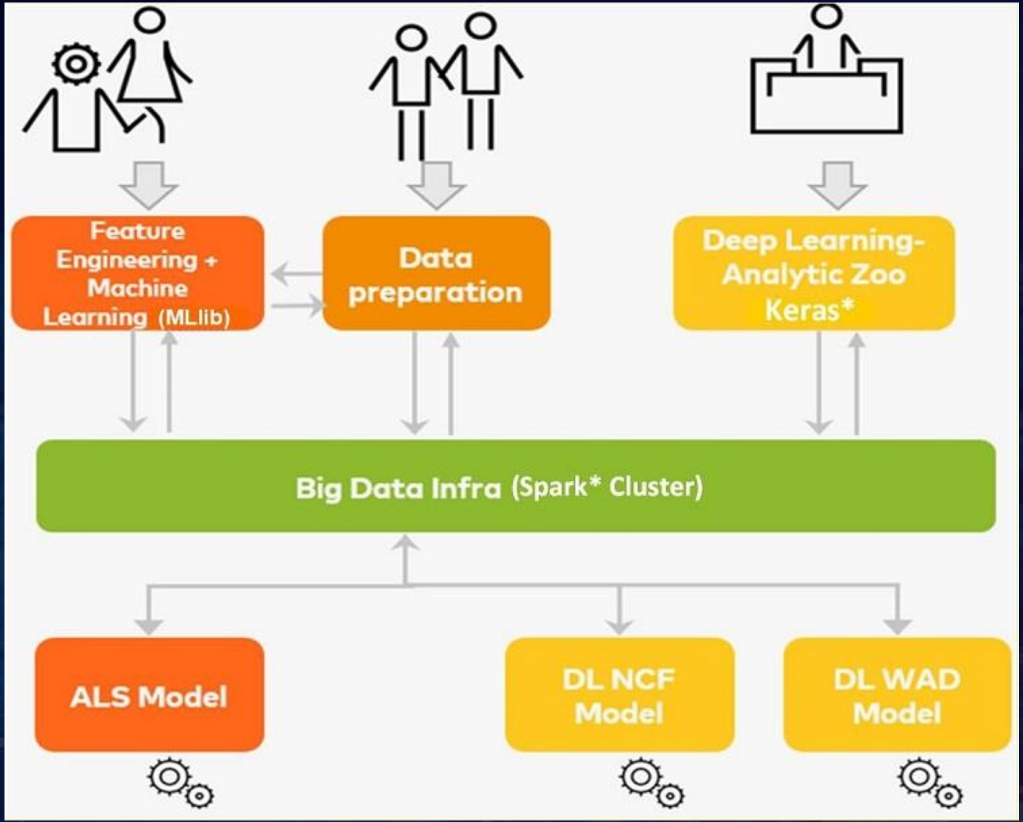
https://conferences.oreilly.com/strata/strata-ca-2019/public/schedule/detail/73079

Computer Vision Based Product Defect Detection in Midea



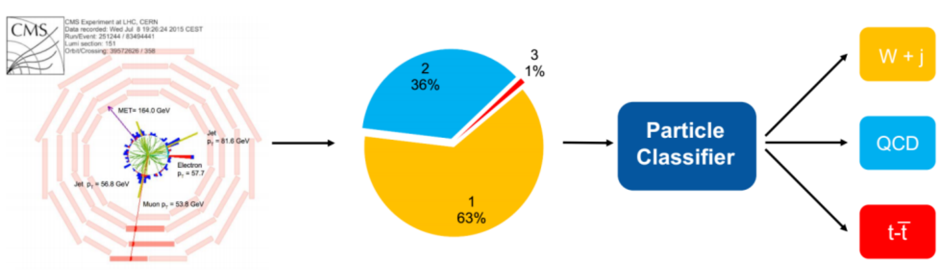
https://software.intel.com/en-us/articles/industrial-inspection-platform-in-midea-and-kuka-using-distributed-tensorflow-on-analytics

Recommender AI Service in MasterCard

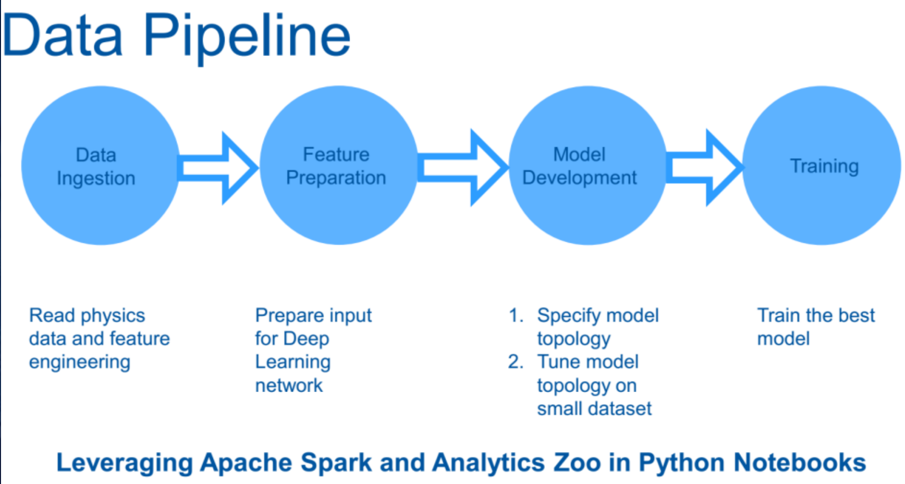


https://software.intel.com/en-us/articles/deep-learning-with-analytic-zoo-optimizes-mastercard-recommender-ai-service

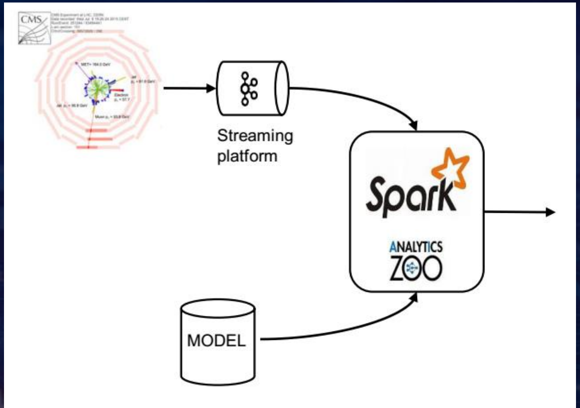
Particle Classifier for High Energy Physics in CERN



Deep learning pipeline for physics data



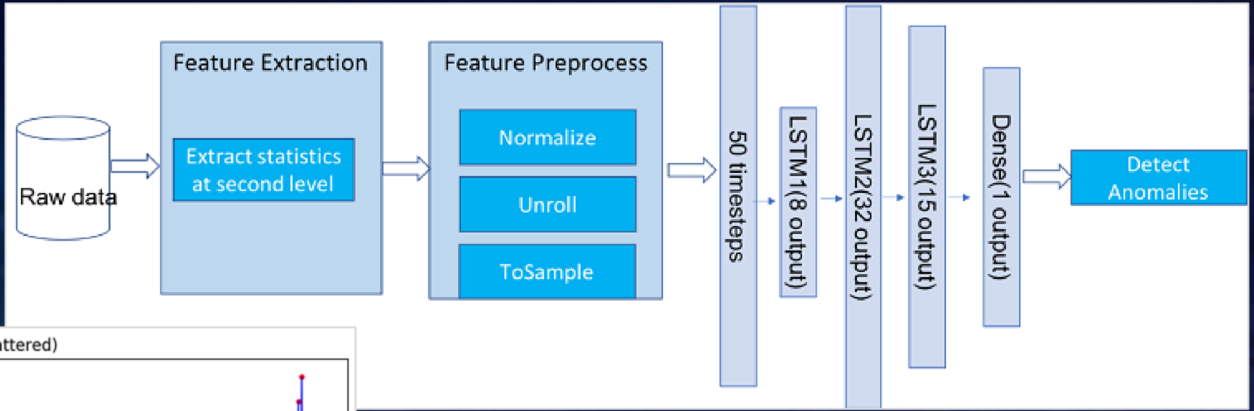
Model serving using Apache Kafka and Spark

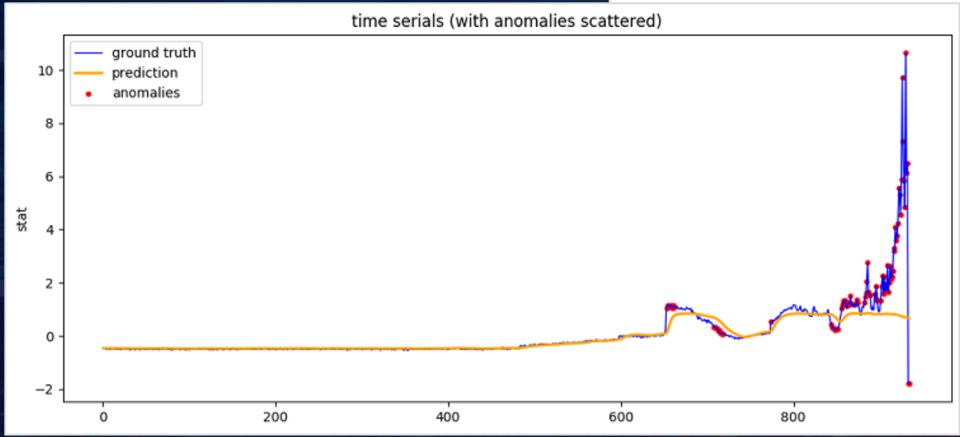


https://db-blog.web.cern.ch/blog/luca-canali/machine-learning-pipelines-high-energy-physics-using-apache-spark-bigdl

https://databricks.com/session/deep-learning-on-apache-spark-at-cerns-large-hadron-collider-with-intel-technologies

Unsupervised Time Series Anomaly Detection for Baosight





https://software.intel.com/en-us/articles/lstm-based-time-series-anomaly-detection-using-analytics-zoo-for-apache-spark-and-bigdl

And Many More

TECHNOLOGY

CLOUD SERVICE PROVIDERS

END USERS

software.intel.com/AlonBigData

Not a full list

More Information

- Analytics Zoo repo: https://github.com/intel-analytics/analytics-zoo/
- Tech Report: https://arxiv.org/abs/1804.05839
- AAAI 2019 Tutorial: https://jason-dai.github.io/aaai2019/
- CVPR 2018 Tutorial: https://jason-dai.github.io/cvpr2018/
- More presentations: https://analytics-zoo.github.io/master/#presentations/

End-to-End Big Data and AI Pipelines

Seamless Scaling from Laptop to Production

ANALYTICS ZOO

Unified Analytics + AI Platform

Distributed TensorFlow*, Keras*, PyTorch* & BigDL on Apache Spark*

https://github.com/intel-analytics/analytics-zoo intel®

LEGAL DISCLAIMERS

- Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer.
- No computer system can be absolutely secure.
- Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/performance.