About US

• Software Engineers at.databricks
• Apache Spark Committers and PMC Members

Xiao Li (Github: gatorsmile)  Wenchen Fan (Github: cloud-fan)
Databricks Unified Analytics Platform

**Databricks Workspace**
- Jobs
- APIs
- Notebooks
- Models
- Dashboards

**Databricks Runtime**
- Databricks Delta
  - Reliable & Scalable
- Apache Spark
  - Simple & Integrated
- ML Frameworks

**Databricks Cloud Service**
- Azure

End to end ML lifecycle
The Growth of the Spark Community

- **2015**: 46,700
- **2016**: 295,675
- **2017**: 375,800
- **2018**: 465,440
- **2019**: 522,018

Spark Meetup Members
This year, over 100,000 developers told us how they learn, build their careers, which tools they’re using, and what they want in a job.

<table>
<thead>
<tr>
<th>Framework</th>
<th>Loved</th>
<th>Dreaded</th>
<th>Wanted</th>
</tr>
</thead>
<tbody>
<tr>
<td>TensorFlow</td>
<td>73.5%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>React</td>
<td>69.4%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Torch/PyTorch</td>
<td>68.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Node.js</td>
<td>66.4%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>.NET Core</td>
<td>66.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spark</td>
<td>66.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spring</td>
<td>60.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Django</td>
<td>58.3%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Angular</td>
<td>54.9%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hadoop</td>
<td>53.9%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Xamarin</td>
<td>49.0%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cordova</td>
<td>40.4%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
20000+ Stars in Github
Top Skill in 2018: Apache Spark


2. Machine Learning Engineer (12X growth)
   - **Top Skills:** Deep Learning, Machine Learning, Tensorflow, Apache Spark, Natural Language Processing
   - **Where They Work:** Apple, Intel, NVIDIA
   - **Top Industries:** Computer Software, Internet, Information Technology & Services
   - **Cities Where Demand is High:** San Francisco, Denver, Austin

7. Data Science Specialist (5X growth)
   - **Top Skills:** Machine Learning, Data Science, Python, R, Apache Spark
   - **Where They Work:** IBM, Facebook, McKinsey & Company
   - **Top Industries:** Higher Education, Information Technology & Services, Computer Software
   - **Cities Where Demand is High:** New York City, San Francisco, Chicago

15. Data Science Manager (4X growth)
   - **Top Skills:** Data Science, Machine Learning, Apache Spark, Python, R
   - **Companies Employing This Talent:** Facebook, Capital One, Microsoft
   - **Top Industries:** Internet, Computer Software, Financial Services
   - **Cities Where Demand is High:** Atlanta, New York City, Los Angeles
Stack Overflow Trends

See how technologies have trended over time based on use of their tags since 2008, when Stack Overflow was founded. Enter up to 15 tags to compare growth and decline.

Tags:
- apache-spark
- hadoop
- apache-kafka
- hive
- cassandra
- hbase

Don't know what tags to look at? Try one of our presets:
- Most Popular Languages (TIOBE Index for May 2017)
- Operating Systems
- Mobile Operating Systems
- Javascript Frameworks
- Smaller Javascript Frameworks
- Closed-source Browser Plugins
- Data Science and Big Data
- Apache Open-source Projects

For more on this tool and what you can learn from it, see our blog post.
Introducing Apache Spark 2.4
Now available on Databricks Runtime 5.0

by Wencheng Fan, Xiao Li and Reynold Xin
Posted in ENGINEERING BLOG | November 8, 2018

UPDATED: 11/19/2018

We are excited to announce the availability of Apache Spark 2.4 on Databricks as part of the Databricks Runtime 5.0. We want to thank the Apache Spark community for all their valuable contributions to the Spark 2.4 release.

Databricks Runtime Version

- 5.0 (includes Apache Spark 2.4.0, Scala 2.11)
- 5.0 ML Beta (Scala 2.11)
- 5.0 (includes Apache Spark 2.4.0, GPU, Scala 2.11)
- 4.3 (includes Apache Spark 2.3.1, Scala 2.11)
- 4.3 (includes Apache Spark 2.3.1, GPU, Scala 2.11)
- 4.2 (includes Apache Spark 2.3.0, Scala 2.11)
- 4.1 (includes Apache Spark 2.3.0, Scala 2.11)
- 4.1 ML Beta (includes Apache Spark 2.3.0, Scala 2.11)
- 4.1 ML Beta (includes Apache Spark 2.3.0, GPU, Scala 2.11)
- 3.5 (includes Apache Spark 2.2.1, Scala 2.11)
- 3.5 LTS (includes Apache Spark 2.2.1, Scala 2.11)

Release: Nov 8, 2018
Blog: https://t.co/k7kEHrNZXp

Above 1100 tickets.
Major Features on Spark 2.4

- Barrier Execution
- Spark on Kubernetes
- Beta support Scala 2.12
- PySpark Improvement
- Structured Streaming
- Image Source
- Native Avro Support
- Built-in source Improvement
- Higher-order Functions
- Various SQL Features
Survey Key Findings: The AI Dilemma

Investment in AI is Growing Quickly

- **80%** are considering new AI projects
- **Nearly 90%** are investing in AI related technology

However, very few are succeeding

- Only **1 in 3** AI projects are a success
- **AI Projects are taking more than 6 MONTHS** to go into production

Major contributing factors to this AI dilemma

- **96%** of enterprises cite data-related challenges with AI projects
- **80%** cite collaboration as a challenge
- **7 Different machine learning frameworks and tools**

CIO Survey Key Findings: Unified Analytics Enables AI Success

79% highly value the notion of a unified analytics platform

Benefits Expected from a Unified Approach to Data & AI

- Increased operational efficiency
- More effective decision making
- Accelerated time to market
- Improved security
- Increased innovation
- Improved customer experience
- Increased competitive advantage
- Increased employee satisfaction
- Increased customer engagement
- Product/service transformation
- Topline growth
- Other (specify)
Big data v.s. AI Technologies
Project Hydrogen: Spark + AI

A *gang scheduling* to Apache Spark that embeds a distributed DL job as a Spark stage to simplify the distributed training workflow. [SPARK-24374]
Major Features on Spark 2.4

- Barrier Execution
- Spark on Kubernetes
- Native Avro Support
- Image Source
- Built-in source Improvement
- PySpark Improvement
- Higher-order Functions
- Scala 2.12
- Structured Streaming
- Various SQL Features
Flexible Streaming Sink

[SPARK-24565] Exposing output rows of each microbatch as a DataFrame

```scala
foreachBatch(f: Dataset[T] => Unit)
```

- Scala/Java/Python APIs in DataStreamWriter.
- Reuse existing batch data sources
- Write to multiple locations
- Apply additional DataFrame operations
Reuse existing batch data sources

```scala
spark.readStream.format("rate").load()
  .selectExpr("value % 10 as key")
  .groupBy("key")
  .count()
  .toDF("key", "value")
  .writeStream
  .foreachBatch { (batchDF: DataFrame, batchId: Long) =>

    batchDF.write // Use Cassandra batch data source to write streaming out
      .cassandraFormat(tableName, keyspace)
      .option("cluster", clusterName)
      .mode("append")
      .save()

    }
  .outputMode("update")
  .start()
```
Write to multiple location

```scala
streamingDF.writeStream.foreachBatch {(batchDF: DataFrame, batchId: Long) =>
  batchDF.persist()
  batchDF.write.format(...).save(...)  // location 1
  batchDF.write.format(...).save(...)  // location 2
  batchDF.unpersist()
}
```
Major Features on Spark 2.4

- Barrier Execution
- Spark on Kubernetes
- Native Avro Support
- Image Source
- Built-in source Improvement
- PySpark Improvement
- Higher-order Functions
- Scala 2.12
- Structured Streaming
- Various SQL Features
Parquet

Update from 1.8.2 to 1.10.0 [SPARK-23972].

- **PARQUET-1025** - Support new min-max statistics in parquet-mr
- **PARQUET-225** - INT64 support for delta encoding
- **PARQUET-1142** Enable `parquet.filter.dictionary.enabled` by default.

Predicate pushdown

- STRING [SPARK-23972] [20x faster]
- Decimal [SPARK-24549]
- Timestamp [SPARK-24718]
- Date [SPARK-23727]
- Byte/Short [SPARK-24706]
- StringStartsWith [SPARK-24638]
- IN [SPARK-17091]
Native vectorized ORC reader is GAed!

- Native ORC reader is on by default [SPARK-23456]
- Update ORC from 1.4.1 to 1.5.2 [SPARK-24576]
- Turn on ORC filter push-down by default [SPARK-21783]
- Use native ORC reader to read Hive serde tables by default [SPARK-22279]
- Avoid creating reader for all ORC files [SPARK-25126]
Major Features on Upcoming Spark 2.4

- Barrier Execution
- Spark on Kubernetes
- Native Avro Support
- Image Source
- Built-in source Improvement
- PySpark Improvement
- Higher-order Functions
- Scala 2.12
- Structured Streaming
- Various SQL Features
Higher-order Functions

Transformation on complex objects like arrays, maps and structures inside of columns.

```
tbl_nested
|-- key: long (nullable = false)
|-- values: array (nullable = false)
|   |-- element: long (containsNull = false)
```

UDF? Expensive data serialization
Higher-order Functions

1) Check for element existence

```sql
SELECT EXISTS(values, e -> e > 30) AS v
FROM tbl_nested;
```

2) Transform an array

```sql
SELECT TRANSFORM(values, e -> e * e) AS v
FROM tbl_nested;
```
Higher-order Functions

3) Filter an array

SELECT FILTER(values, e -> e > 30) AS v
FROM tbl_nested;

4) Aggregate an array

SELECT REDUCE(values, 0, (value, acc) -> value + acc) AS sum
FROM tbl_nested;

Ref Databricks Blog: http://dbricks.co/2rUKQ1A
Built-in Functions

[SPARK-23899] New or extended built-in functions for ArrayTypes and MapTypes

- **26 functions for ArrayTypes**
  - transform, filter, reduce, array_distinct, array_intersect, array_union, array_except, array_join, array_max, array_min, ...

- **3 functions for MapTypes**
  - map_from_arrays, map_from_entries, map_concat

Blog: Introducing New Built-in and Higher-Order Functions for Complex Data Types in Apache Spark 2.4. [https://t.co/p1TRRtabJJ](https://t.co/p1TRRtabJJ)
Major Features on Spark 2.4

- Barrier Execution
- Spark on Kubernetes
- Native Avro Support
- Image Source
- Built-in source Improvement
- PySpark Improvement
- Higher-order Functions
- Scala 2.12
- Structured Streaming
- Various SQL Features
Native Spark App in K8S

New Spark scheduler backend

- PySpark support [SPARK-23984]
- SparkR support [SPARK-24433]
- Client-mode support [SPARK-23146]
- Support for mounting K8S volumes [SPARK-23529]

Blog: What’s New for Apache Spark on Kubernetes in the Upcoming Apache Spark 2.4 Release
https://t.co/uUpdUj2Z4B
Major Features on Spark 2.4

- Barrier Execution
- Spark on Kubernetes
- Beta support Scala 2.12
- PySpark Improvement
- Structured Streaming
- Image Source
- Native Avro Support
- Built-in source Improvement
- Higher-order Functions
- Various SQL Features
What’s Next?
Safe Harbor Statement

This presentation may contain projections or other forward-looking statements regarding the upcoming release (Apache Spark 3.0). The statements are intended to outline our general direction. They are intended for information purposes only. They are not a commitment to deliver code or functionality. The development, release and timing of any feature or functionality described for Apache Spark remains at the sole discretion of ASF and the Apache Spark PMC.
What’s Next?

- GPU-aware Scheduling
- Spark Graph
- Data Source APIs
- Adaptive Execution
- Spark on Kubernetes
- mlflow
- Hadoop 3.x
- Scala 2.12
- Various SQL Features
- PySpark Usability
What’s Next?

- GPU-aware Scheduling
- Spark Graph
- Data Source APIs
- Adaptive Execution
- Spark on Kubernetes
- mlflow
- Hadoop 3.x
- Scala 2.12
- Various SQL Features
- PySpark Usability
Project Hydrogen: Spark + AI

GPU Aware Scheduling

• widely used for accelerating special workloads, e.g., deep learning and signal processing
It’s Hard to Productionize ML
ML Lifecycle is Manual, Inconsistent and Disconnected

Data Prep
- Low level integrations for Data and ML
- Difficult to track data used for a model

Build Model
- Ad hoc approach to track experiments
- Very hard to reproduce experiments

Deploy Model
- Multiple tightly coupled deployment options
- Different monitoring approach for each framework
What is mlflow?

Open source platform to manage ML development

- Lightweight APIs & abstractions that work with any ML library
- Designed to be useful for 1 user or 1000+ person orgs
- Runs the same way anywhere (e.g. any cloud)

Key principle: “open interface” APIs that work with any existing ML library, app, deployment tool, etc
MLflow Components

**mlflow Tracking**
Record and query experiments: code, params, results, etc

**mlflow Projects**
Code packaging for reproducible runs on any platform

**mlflow Models**
Model packaging and deployment to diverse environments
What's Next?

MLflow Tracking
- SQL database backend for scaling the tracking server (0.9)
- UI scalability improvements (0.8, 0.9, etc.)
- X-coordinate logging for metrics & batched logging (1.0)
- Fluent API for Java and Scala (1.0)

MLflow Projects
- Docker-based project environment specification (0.9)
- X-coordinate logging for metrics & batched logging (1.0)
- Packaging projects with build steps (1.0+)

MLflow Models
- Custom model logging in Python, R and Java (0.8, 0.9, 1.0)
- Better environment isolation when loading models (1.0)
- Logging schema of models (1.0+)
MLflow: An open platform to simplify the machine learning lifecycle

4:20pm-5:00pm, Mar 27 / 2008

Session

Topics: Data Engineering & Architecture

Developing applications that leverage machine learning is difficult. Practitioners need to be able to reproduce their model development pipelines, as well as deploy models and monitor their health in production. Corey Zumar offers an overview of MLflow, which simplifies this process by managing, reproducing, and operationalizing machine learning through a suite of model tracking and deployment APIs.

Corey Zumar
Databricks
What’s Next?

- GPU-aware Scheduling
- Spark Graph
- Data Source APIs
- Adaptive Execution
- Spark on Kubernetes
- mlflow
- Hadoop 3.x
- Scala 2.12
- Various SQL Features
- PySpark Usability
Challenges in Existing Graph Library

GraphX
- Not DataFrame based
- Not actively maintained

GraphFrame
- Limited graph pattern matching
- Semantically weak graph data model
Spark Graph

Given a single Property Graph data model and a Cypher query, Spark returns a tabular result [DataFrame]
What’s Next?

- Data Source APIs
- Spark on Kubernetes
- Adaptive Execution
- PySpark Usability
- Various SQL Features
- Scala 2.12
- mlflow
- Hadoop 3.x
- Spark Graph
- GPU-aware Scheduling
Data Source API V2

- **Unified API** for batch and streaming
- Flexible API for **high performance** implementation
- Flexible API for **metadata management**
JDBC source with data source v1

1. Specify the info of remote catalog for each op

```java
   df.write.format("jdbc")
   .option("url", ...)  
   .option("dbtable", ...)  
   .option("driver", ...)  
   .save()
```
JDBC source with data source v1

2. Register each **table** before usage

CREATE TABLE tab1(...) USING jdbc OPTIONS("url" ..., "dbtable" ..., ...)
CREATE TABLE tab2(...) USING jdbc OPTIONS("url" ..., "dbtable" ..., ...)
SELECT * FROM tab1 join tab2
INSERT INTO tab1 SELECT ...
JDBC source with data source v2

New: Register the catalog before usage

spark-defaults.conf

spark.sql.catalog.jdbcCatalogName  my.jdbc.v2.impl
spark.sql.catalog.jdbcCatalogName.url  ...
Spark.sql.catalog.jdbcCatalogName.driver  ...
JDBC source with data source v2

- No need to register the tables.
- Access the tables using n-part name.
- DDL/DML support.

```
CREATE TABLE jdbcCatalogName.db1.t1(...)  
ALTER TABLE jdbcCatalogName.db1.t2 CHANGE COLUMN ...
SELECT * FROM jdbcCatalogName.db2.t3  
INSERT INTO jdbcCatalogName.db3.t4 SELECT ...
```
What’s Next?

- GPU-aware Scheduling
- Spark Graph
- Data Source APIs
- Adaptive Execution
- Spark on Kubernetes
- mlflow
- Hadoop 3.x
- Scala 2.12
- Various SQL Features
- PySpark Usability
Spark adaptive execution: Unleash the power of Spark SQL

1:50pm-2:30pm, Mar 28 / 2004

Session

Topics: Data Engineering & Architecture

Spark SQL is widely used, but it still suffers from stability and performance challenges in highly dynamic environments with large-scale data. Haifeng Chen shares a Spark adaptive execution engine built to address these challenges. It can handle task parallelism, join conversion, and data skew dynamically during runtime, guaranteeing the best plan is chosen using runtime statistics.

Haifeng Chen
Intel
Adaptive Query Processing

Based on statistics of the materialized plan nodes, re-optimize the execution plan of the remaining queries

• Self tuning the number of reducers
• Adaptive join strategy
• Automatic skew join handling

Intel Blog: https://tinyurl.com/y3rjwcos
Adaptive Query Processing
What’s Next?

- GPU-aware Scheduling
- Spark Graph
- Data Source APIs
- Adaptive Execution
- Spark on Kubernetes
- mlflow
- Hadoop 3.x
- Scala 2.12
- Various SQL Features
- PySpark Usability
Native Spark App in K8S

- Support for using a pod template to customize the driver and executor pods.
- Dynamic resource allocation and external shuffle service.
- Better support for local application dependencies on client machines.
- Driver resilience for Spark Streaming.
- Better scheduling support.
Scaling Apache Spark on Kubernetes at Lyft

3:50pm-4:30pm, Mar 28 / 2001

Session

Topics: Data Engineering & Architecture

Li Gao and Bill Graham discuss the challenges the Lyft team faced and solutions they developed to support Apache Spark on Kubernetes in production and at scale.

Li Gao
Lyft

Bill Graham
Lyft
The other targets in Apache Spark 3.0

- Hadoop 3.x support
- Hive execution from 1.2.1 to 2.3.4
- Scala 2.12 GA
- Better ANSI SQL compliance
- PySpark usability

Please follow the announcements in Spark + AI Summit @ SF
What’s Next?

- GPU-aware Scheduling
- Spark Graph
- Data Source APIs
- Adaptive Execution
- Spark on Kubernetes
- mlflow
- Hadoop 3.x
- Scala 2.12
- Various SQL Features
- PySpark Usability
Apache Spark 3.x

SQL

Spark ML

Spark Streaming

Spark Graph

3rd-party Libraries

SparkSession / DataFrame / DataSet APIs

Catalyst Optimization & Tungsten Execution

Data Source Connectors

Spark Core
SPARK+AI SUMMIT 2019
 APRIL 23 - 25 | SAN FRANCISCO
ORGANIZED BY databricks

TRACKS
Apache Spark™
• Use Cases
• Research
• Technical Deep Dives

AI
• Productionizing ML
• Deep Learning
• Cloud Hardware

Fields
• Data Science
• Data Engineering
• Enterprise

5000+ ATTENDEES
Practitioners:
Data Scientists, Data Engineers, Analysts, Architects

Leaders:
Engineering Management, VPs, Heads of Analytics & Data, CxOs

databricks.com/sparkaisummit
Nike: Enabling Data Scientists to bring their Models to Market
Facebook: Vectorized Query Execution in Apache Spark at Facebook
Tencent: Large-scale Malicious Domain Detection with Spark AI
IBM: In-memory storage Evolution in Apache Spark
Capital One: Apache Spark and Sights at Speed: Streaming, Feature management and Execution
Apple: Making Nested Columns as First Citizen in Apache Spark SQL
EBay: Managing Apache Spark workload and automatic optimizing.
Google: Validating Spark ML Jobs
HP: Apache Spark for Cyber Security in big company
Microsoft: Apache Spark Serving: Unifying Batch, Streaming and RESTful Serving
ABSA Group: A Mainframe Data Source for Spark SQL and Streaming
Facebook: an efficient Facebook-scale shuffle service
IBM: Make your PySpark Data Fly with Arrow!
Facebook: Distributed Scheduling Framework for Apache Spark
Zynga: Automating Predictive Modeling at Zynga with PySpark
World Bank: Using Crowdsourced Images to Create Image Recognition Models and NLP to Augment Global Trade indicator
Microsoft: Azure Databricks with R: Deep Dive

Airbnb: Apache Spark at Airbnb
Netflix: Migrating to Apache Spark at Netflix
Microsoft: Infrastructure for Deep Learning in Apache Spark
Intel: Game playing using AI on Apache Spark
Facebook: Scaling Apache Spark @ Facebook
Lyft: Scaling Apache Spark on K8S at Lyft
Uber: Using Spark Mllib Models in a Production Training and Serving Platform
Apple: Bridging the gap between Datasets and DataFrames
Salesforce: The Rule of 10,000 Spark Jobs
Target: Lessons in Linear Algebra at Scale with Apache Spark
Nationwide: Deploying Enterprise Scale Deep Learning in Actuarial Modeling at Nationwide
Workday: Lesson Learned Using Apache Spark
Thank you

Xiao Li (lixiao@databricks.com)
Wenchen Fan (wenchen@databricks.com)