Lesson 7

Apache® Spark™ Streaming

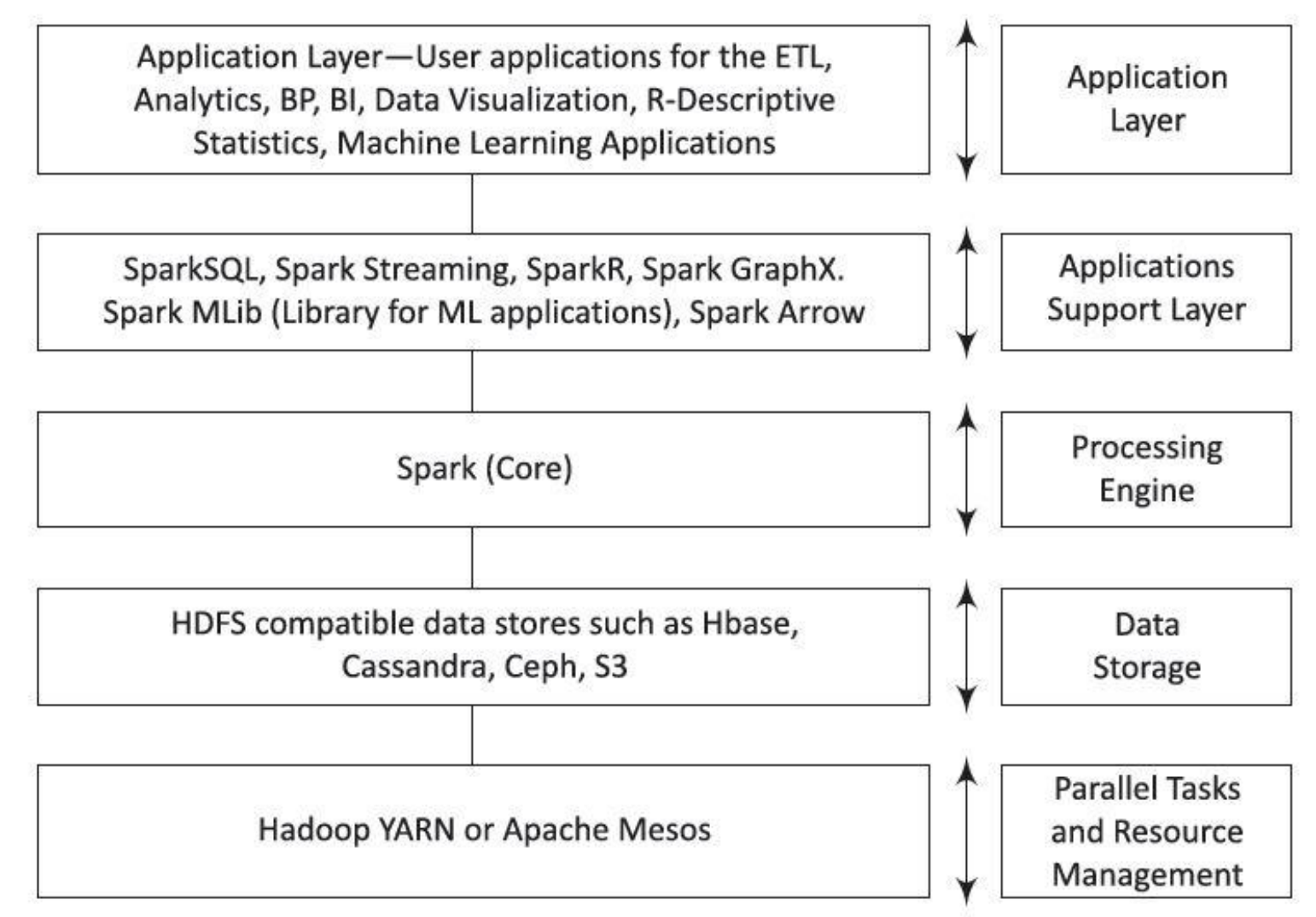
SparkStreaming

- An extension of core Apache Spark API
- Applications are in a variety of use cases and business applications
- SparkStreaming use cases include Uber (the ride sharing service), Pinterest, (the content sharing service), Netflix (a subscription service that provides access to movies and TV shows).

Spark Software Stack

- Spark stack main components, namely Core, SQL, Streaming, R, GraphX, MLib and Arrow in a five-layered architecture.
- All software components are also available when using SparkStreaming.

Figure 5.3 Spark Streaming as a part of Apache Spark Stack



Features of data processing using SparkStreaming

1. Combines batch processing and streaming processing in the same system
2. Applies Spark's machine learning and graph processing algorithms on data stream

Features of data processing using SparkStreaming

3. Divides the stream of data into micro-batches of a pre-defined interval (N seconds)

   A micro-batch is used in operations similar to Resilient Distributed Datasets (RDDs).

Features of data processing using SparkStreaming

4. Support Scala, Java, R and Python language in the form of suitable APIs
5. Results save at a data store for performing analytics later
6. Results also generate reports, display visuals on live dashboard and generate alerts on the events.

SparkStreaming Library

- DStream (Discretized Stream)—an abstraction of a continuous data stream in SparkStreaming

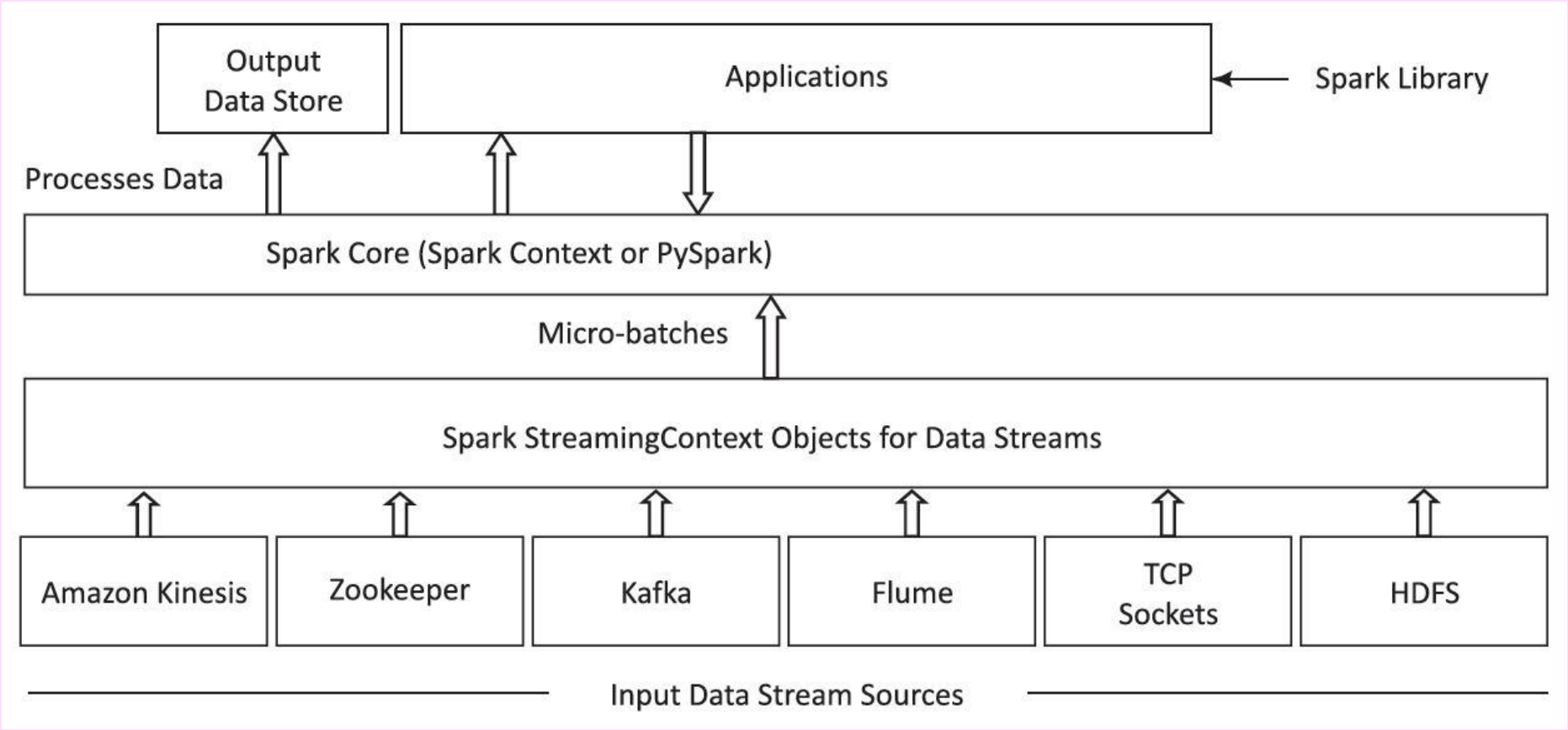
DStream

- Creates either from basic sources or input data stream from sources, such as Kafka, Flume and Kinesis
- Stream operators apply functions on Dstreams
- DStream represents streaming data from a TCP source

Apache Kafka and ZooKeeper

- Kafka A real-time, fault tolerant, scalable messaging system for moving data in real time
- ZooKeeper—a centralized service providing reliable distributed coordination for distributed applications.

Figure 7.7 Various architecture components of SparkStreaming application



SparkStreaming functionalities and operators

- Creating StreamingContext Object using SparkConf
- Operations on DStreams from input sources (join, cogroup, union)
- Streaming transformation operations

Stream APIs Processing RDD like Operations

- map(func); flatMap(func); filter(func)
- reduce(func); repartition(numPartitions)
- count(); countByValue()
- reduceByKey(func, [numTasks])
- updateStateByKey(func)

Streaming output operations

- foreach RDD(func)
- saveAsTextFiles(prefix, [suffix]);
- saveAsObjectFiles(prefix, [suffix])
- saveAsHadoopFiles(prefix, [suffix])
- print() or
- pprint() in Python

Stream APIs (time) Window Related Transformations

- Window(windowLength, slideInterval)
- countByWindow(windowLength, slideInterval); reduceByWindow(func, win-dowLength, slideInterval)

Stream APIs (time) Window Related Transformations

- reduceByKeyAndWindow(func, win-dowLength, slideInterval, [numTasks])
- reduceByKeyAndWindow(func, invFunc, windowLength, slideInterval, [numTasks])

Spark Streaming Start and Stop

- streamingContext.start() for starting data receiving and processing
- streamingContext.awaitTermination() for Termination
- streamingContext.stop() for
- stop operations
-

Create a program in Scala for counting the number of words

- Example 7.9 Uses stream from a socket in every 1 second, using window of length 60 seconds and sliding interval of 10 seconds, and save the result in windowed WordCounts.
- Uses SparkStreaming context and API.

Summary

We learnt:

- Spark Streaming
- Combines batch processing and streaming processing in the same system
- Various architecture components of SparkStreaming application
- SparkStreaming functionalities and operators

End of Lesson 7 on

Apache® Spark™ Streaming