Radish: a distributed relevant image retrieval system

Giovanni Berti

20-04-2020

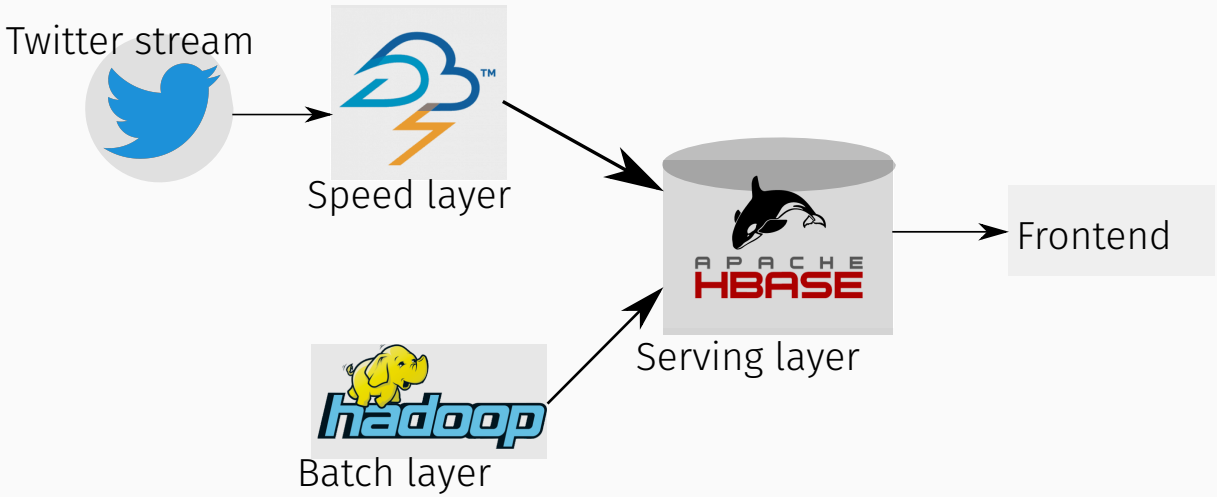
Università degli Studi di Firenze

- Lambda Architecture
- Image retrieval
- Implementation notes

Introduction

- New software architectures needed to handle web-scale data analysis
- *Lambda Architecture* approach for scalable realtime systems
  - *Batch layer*: batch-oriented computation, high latency
  - *Serving layer*: presents view on data emitted by the batch layer
  - *Speed layer*: low latency, incremental results, periodic synchronization with batch layer
- Image retrieval systems as an example of data-intensive application

Architecture



- Implementation of a distributed architecture
- Speed layer started with a list of keywords to use as a filter on tweets' text
- Three layer collaborate to provide a unified view to the frontend

Image relevance

- Image relevance for a keyword is mapped to a clustering problem
- Compute clusters to group similar images
- Image nearest to cluster center is most representative
- Clusters computed on CEDD feature vectors

Serving layer

Three tables in HBase:

- *images* table, keeps track of downloaded images and their relative feature vectors
- *clusters* table, stores results of batch clustering
- *clusters_speed*, updates clustering results in real-time based on batch data

Synchronization between tables based on timestamps of last update provided by HBase API

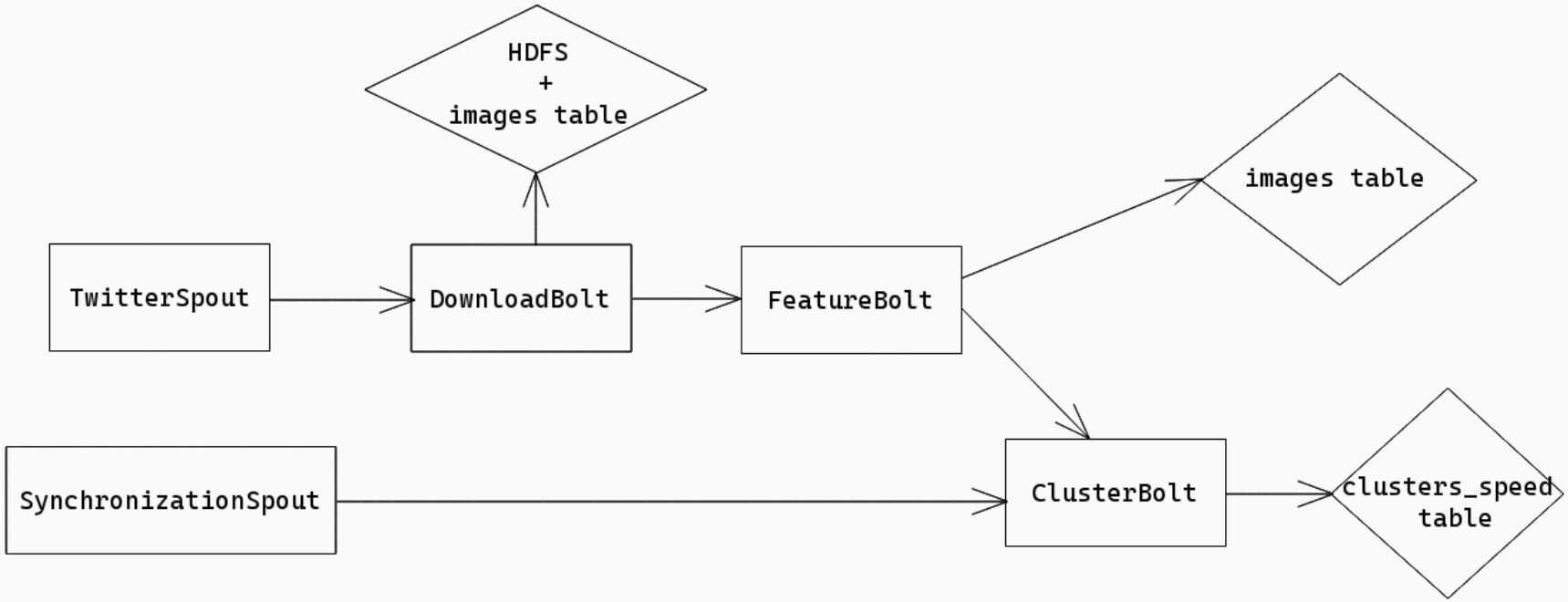
Batch layer

- Implemented with Apache Hadoop with MapReduce framework
- Reads feature vectors from *images* table, computes cluster centroids
- MapReduce job that reads and writes data internally invokes another MapReduce job to cluster data
- Clustering implementation: k-means++
- Final output: pairs $\langle centroid, nearest\_member \rangle$

Speed layer

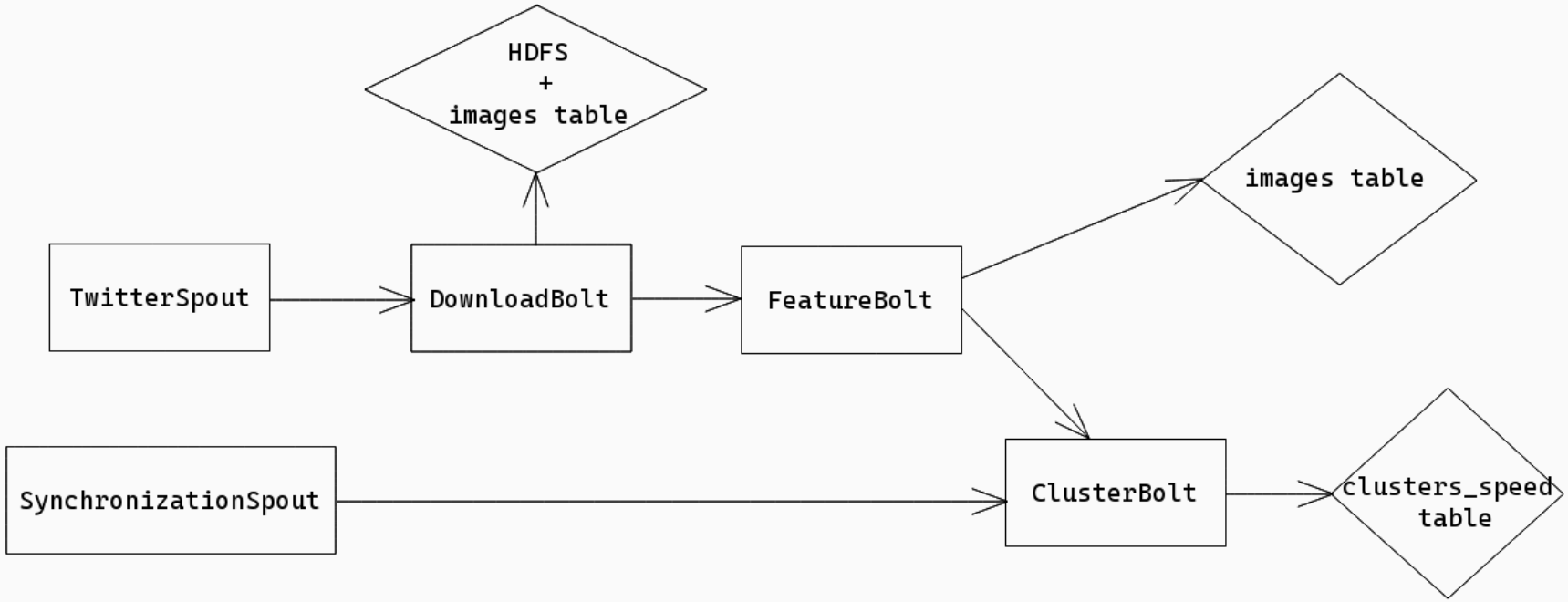
- Apache Storm topology, stream oriented computation
- Downloads images from Twitter stream
- Writes feature vectors to *images* table
- Persists images to HDFS
- Updates HBase real-time view
- Synchronizes with batch updates

Speed layer diagram



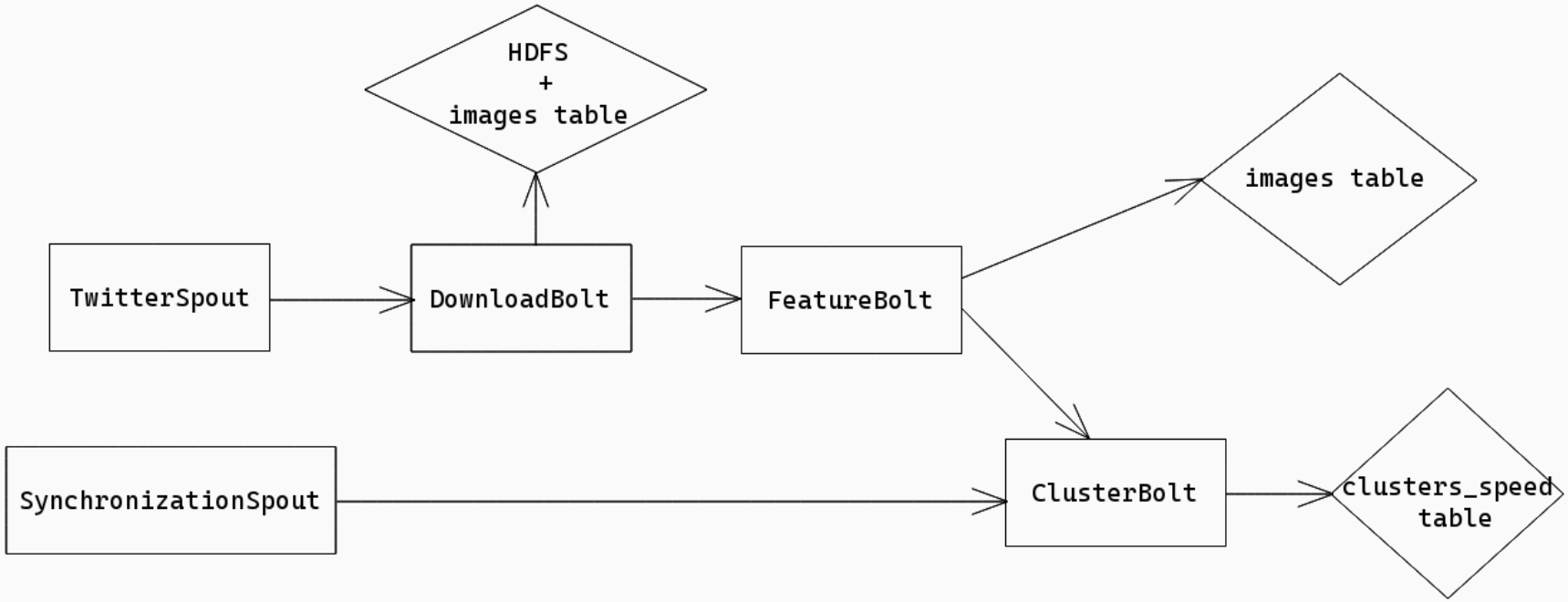
- Each diamond has a corresponding Storm bolt that maps data according to an application-defined schema

Speed layer diagram



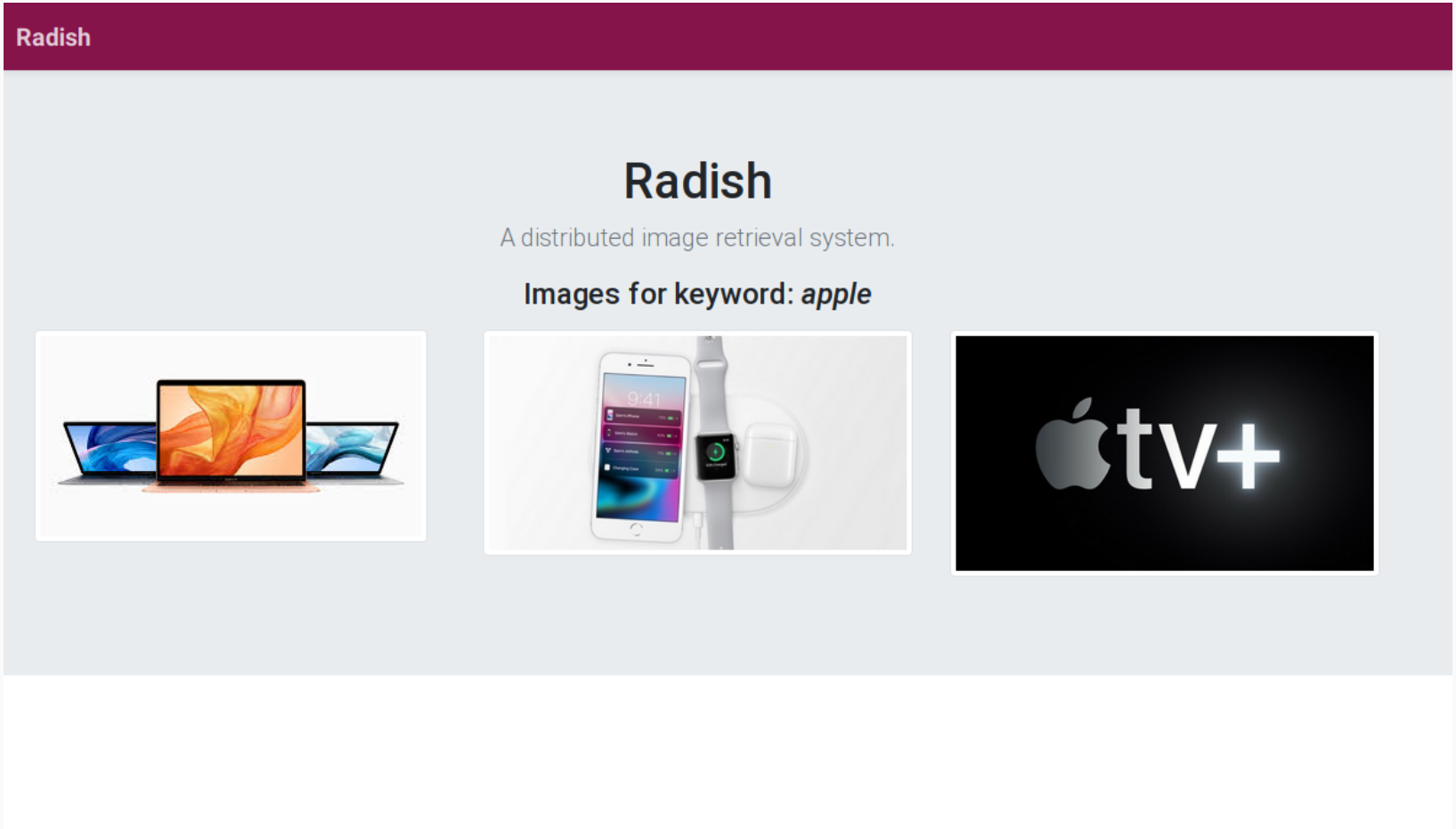
- TwitterSpout: reads twitter stream and filters tweets by keyword in tweet text and tweets that have at least an image
- DownloadBolt: persists images to HDFS
- FeatureBolt: computes CEDD descriptors for incoming images

Speed layer diagram



- SynchronizationSpout: emits centroid data from batch view every time it is updated
- ClusterBolt: updates real-time view

Frontend


Radish
Radish
A distributed image retrieval system.
Images for keyword: *apple*
9:41
tv+

Conclusions

- A distributed image retrieval system has been implemented with a Lambda Architecture
- A simple frontend was developed to query incoming data
- The usage of more sophisticated underlying algorithm can pave a path to future developments