The “Big Data” Ecosystem at LinkedIn

Roshan Sumbaly, Jay Kreps, and Sam Shah

Presenter: Camilo Muñoz
Lack of literature on productionizing machine learning workflows

Data analytics and warehousing → Hive

Ingress → Scribe (Twitter), Chukwa (Yahoo)

Egress → Hbase, PNUTS (Yahoo)
Introduction

- LinkedIn +200 million users
- Data-driven features → collaborative filtering (*wisdom of the crowd*)
- Hadoop provides a rich ecosystem → horizontal scalability, fault tolerance, and multitenancy
- How to make life easier for machine learning researchers and data scientists
Let’s talk Apache

The “Big Data” Ecosystem at LinkedIn
The Data Ecosystem

- Online – Offline – Online. HDFS acts as the sink for all the data
- 2 types of incoming data → event data and core database snapshots
- From ETL Hadoop instance to dev and prod
- Researchers and DS define workflows to play with the data
- Data delivery → Key-value, Data Streams, and Analytics/OLAP
- Avro as the standard serialization format
Ingress

- Challenge to make data available without manual intervention → large datasets, diverse data, evolving functionalities, and data quality
- Kafka allows data publishers interact with consumers through topics
- Distribution of logical consumers for large data feeds → Zookeeper
Ingress: Data Evolution

- Unstructured vs structured data
- LinkedIn uses a schema registry to map topics to schemas
Ingress: Hadoop Load

- Data pulled from Kafka brokers into Hadoop every 10 minutes
- Replication → from ETL cluster to prod and dev cluster
- LinkedIn maintains topics historic data
- 2 Kafka clusters for event data → primary (online services) and secondary (offline prototyping and data loading into Hadoop). Use of mirroring process for sync
- 100 TB = 300 topics
- 15 billion messages writes, 55 billion messages reads
Ingress: Monitoring

- Audit trail → assessment of correctness and latency
- Audit data: topic, machine name, time window, number of events
- Continuous audit → Programmed to alert if completeness is not reached in a fixed time
Workflows

- Workflow → Chain of MapReduce jobs. DAG
- Primary interfaces → Hive, Pig, and native MapReduce
- Common functionalities between workflows → creation of wrappers to read and write time-partitioned data
Workflows: Azkaban

- Configuration and dependencies for jobs are maintained as files of simple key-value pairs
- Researchers can edit, deploy, monitor, restart, setup notifications, and even capture logs and statistics
- Example: ML application → Each feature becomes an individual Azkaban job followed by a join of the output of these jobs into a feature vector.
- $A_{DEV} \rightarrow$ test period $\rightarrow$ production review $\rightarrow A_{PROD}$
Egress: Key-value access (70%)

- Voldemort → distributed key-value store with a simple get(key) and put(key, value) interface.

- Tuples are grouped into logical stores (tables). Keys are replicated. Nodes are split into logical partitions → A key is mapped to multiple partitions (hashing).
Egress: Stream-oriented access (20%)

- Useful for applications that need a change log of the underlying data.
- Hadoop OutputFormat
Egress: OLAP access (10%)

- Avatara separates cube generation (high throughput) and query serving (low latency)
- Large cubes are split into ‘small cubes’ using shard key
Applications: Key-value

- People You May Know → Link prediction problem. Key = member ID, Value = list of member ID, score
- Collaborative Filtering → Association rule mining, member-to-member, member-to-company. Key = <entity ID, entity type>, Value = top related entity pairs.
- Skill Endorsements → Definition of a Taxonomy (e.g., “Rails” is the same as “Ruby on Rails”). Key = member ID, Value = <member ID, skill ID, score>.
- Related Searches → Member search activity. Key = <term ID, local>, Value = search term
Applications: Key-value
Applications: Streams

- News Feed Updates → A connection updates her profile, company that most of a member’s former coworkers now work for
- Email → Online or offline. Examples: password recovery, joining a group, weekly digest.
- Relationship Strength → LinkedIn’s social graph edge scoring. Examples: best path in the graph, search typeahead, search suggestions
Applications: OLAP

- Who’s Viewed My Profile?
- Who’s Viewed This Job?
Key Take away

A rich developer ecosystem empowers machine learning researchers and data scientists to productionize their work →
Their focus is to build data products
Big Data Ecosystem at LinkedIn

- Data Ingress
  - Moving data from online to offline system

- Data Processing
  - Batch processing using Hadoop, Azkaban, Cubert
  - Stream processing using Samza
  - Iterative processing using Spark

Big 2015 at WWW
The “Big Data” Ecosystem at LinkedIn

Simplifying Data Integration

Gobblin’ Big Data with Ease

Lin Qiao
Data Analytics Infra @ LinkedIn

Open source @ github.com/linkedin/gobblin
Adopted by LinkedIn, Intel, Swisscom, Prezi, PayPal, NerdWallet and many more...
Apache incubation under way

Hundreds of TB per day
Thousands of datasets
~30 different source systems
80%+ of data ingest
Discussion

- How about empowering prototyping and feature testing?
- Trade-off between in-house infrastructure and on the cloud infrastructure
- Is there a better replication schema than ETL+Dev+Prod?