Hopsworks Feature Store: Fast & Fresh Data for AI at Scale

Moritz Meister
Software Engineer
Logical Clocks

Fabio Buso
Head of Engineering
Logical Clocks
HOPSWORKS
Feature Store
Fast & Fresh Data for AI at Scale
Road To AI Value

- **Traditional Analytics**
  - Reporting, Dashboards
  - Offline predictions
  - Ad-hoc analysis, unsupervised learning

- **Analytical ML**
  - Offline predictions

- **Operational ML**
  - Online predictions

- **Real-Time ML**
  - Online predictions

**Enterpise Machine Learning Capabilities**
Road To AI Value

- **Traditional Analytics**
  - Operational Sources (DWH)
  - Reporting, Dashboards

- **Ad-hoc ML Research**
  - Batch Sources
  - Ad-hoc analysis, unsupervised learning

- **Analytical ML**
  - Batch Sources

- **Operational ML**
  - Batch Sources, Application Data

- **Real-Time ML**
  - Streaming Sources, Application Data

- **Business Value**
Road To AI Value

Enterprise Machine Learning Capabilities

- **Traditional Analytics**: Reporting, Dashboards
- **Ad-hoc ML Research**: Batch Sources
- **Analytical ML**: Batch Sources, Application Data
- **Operational ML**: Batch Sources, Application Data
- **Real-Time ML**: Streaming Sources, Application Data

Data Complexity

- Offline predictions: Batch Sources
- Online predictions: Batch Sources, Application Data
- Ad-hoc analysis, unsupervised learning: Batch Sources
- Reporting, Dashboards: Operational Sources (DWH)
Mapping the Requirements

Enterprise Machine Learning Capabilities

- **Traditional Analytics**
- **Ad-hoc ML Research**
- **Analytical ML**
- **Operational ML**
- **Real-Time ML**

**SLAs for**
- lookup throughput and latency
- Feature freshness

**Volume**
- Data Versioning
  - (point-in-time correctness)
- Access
- Reusability

**Business Value**

Feature Stores for ML
Adding Infrastructure to the Mix

**LATS datastore**
(low Latency, high Availability, high Throughput, Scalable storage)

**Data Lake/Warehouse**
(Lakehouse)

**Consistency**

SLAs for
lookup throughput and latency
Feature freshness

Volume
Data Versioning
(point-in-time correctness)
Access
Reusability

Enterprise Machine Learning Capabilities
Is this a Feature Store?

Enterprise Machine Learning Capabilities

Online Storage

Consistency

Offline Storage

SLAs for
- lookup throughput and latency
- Feature freshness

Volume
- Data Versioning
- (point-in-time correctness)
- Access
- Reusability
Hopworks Feature Store

**Data Sources**

**Online**
- Real-time Features
- Low Latency Feature Update
- Transformations

**Offline**
- Batch
- Activity
- Time Travel
- Data Validation
- Statistics

**Feature Engineering & preprocessing**
- Feature Management
- Custom Metadata

**Lineage**

**Metadata**
Solving the Hardest Problems at Scale: Real-Time!
Hopworks Feature Store

**Scale-Out Metadata:**
- Governance
- Unifying Offline/Online
- Search
- Statistics

**HSFS API:**
- Consistency
- Accessibility

**Low Latency, High Availability, High Throughput and Scalable Storage (LATS)**

- RonDB
Solving the Hardest Problems at Scale: Feature Freshness!
How It Works

User/Application

fg.insert(df)

Prometheus
Grafana

Monitoring entire Platform
Can also be utilized for Drift/Feature Monitoring
HSFS: How To Use It - Batch Ingest

```python
df = spark.read.parquet(...)  # do feature engineering with your favourite library

# get expectations
customer_exp = fs.get_expectation("customer_id")

fg = fs.create_feature_group(
    "my_feature_group",
    version=1,
    description="Grouping features that were engineered together",
    primary_key=["location_id", "customer_id"],
    partition_key=["year"],
    online_enabled=True,
    event_time="event_ts",
    statistics_config=True,
    validation_type=STRICT,
    expectations=[customer_exp],
)

fg.save(df)
```
HSFS: How To Use It - External Feature Groups

```python
snowflake_conn = fs.get_storage_connector("telco_snowflake_cluster")

telco_on_dmd = fs.create_on_demand_feature_group(name="telco_snowflake",
        version=1,
        query="select * from telco",
        description="On-demand FG",
        storage_connector=snowflake_conn,
        statistics_config=True)

telco_on_dmd.save()
```
HSFS: How To Use It - Stream Ingest

```python
df_read = spark.readStream.format("kafka") \  
   .option("subscribe", KAFKA_TOPIC_NAME).load()

# Deserialise data from Kafka and create streaming query
df_deser = df_read.selectExpr(...).select(...)

# 10 minute window
windowed_df = df_deser \  
   .selectExpr(...) \  
   .withWatermark(...) \  
   .groupBy(window("datetime", "10 minutes"), "cc_num").agg(avg("amount")) \  
   .select(...)

card_transactions = fs.get_feature_group("card_transactions", version = 1)
query_10m = card_transactions.insert_stream(windowed_df)
```
```java
Properties kafkaProperties = utils.getKafkaProperties(featureGroup);
aggregationStream
  .rescale()
  .rebalance()
  .addSink(
    new FlinkKafkaProducer<byte[]>(
      featureGroup.getOnlineTopicName(),
      new AvroKafkaSink(keyName, featureGroup.getOnlineTopicName()),
      kafkaProperties,
      FlinkKafkaProducer.Semantic.AT_LEAST_ONCE)
  );
```
Throughput of writing 20M feature vectors from Kafka to RonDB using two instances of the OnlineFS materialization service.
Two-node RonDB cluster (r5.2xlarge VMs) scales linearly to >250k ops/sec with feature vector lookups of 11 features of ~1KB in size and 7.5 ms p99 latency.
Demo

Online Feature Store

Streaming Data

Fresh Features <200 ms

Serving
Thank you!

Do you have any questions?

Moritz Meister & Fabio Buso
Hopsworks
info@logicalclocks.com

Logical Clocks
@hopsworks