EXPERIENCES WITH STREAMING & MICRO-BATCH FOR ONLINE LEARNING

Swaminathan Sundararaman

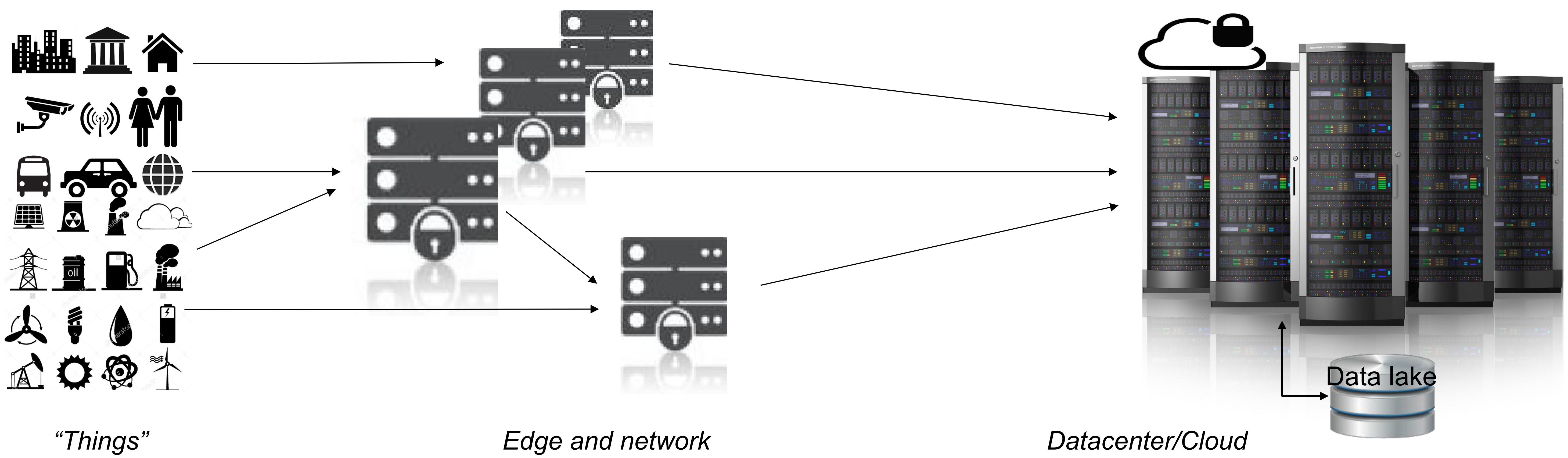
FlinkForward 2017

PARALLEL MACHINES

Powering Machine Learning

The Challenge of Today's Analytics Trajectory

IoT is Driving Explosive Growth in Data Volume



Edges benefit from real-time online learning and/or inference

Real-Time Intelligence: Online Algorithm Advantages

- **Real-world data is unpredictable and bursty**
  - Data behavior changes (different time of day, special events, flash crowds, etc.)
- **Data behavior changes require retraining & model updates**
  - Updating models offline can be expensive (compute, retraining)
- **Online algorithms retrain on the fly with real-time data**
  - Lightweight, low compute and memory requirements
  - Better accuracy through continuous learning
- **Online algorithms are more accurate, especially with data behavior changes**

Experience Building ML Algorithms on Flink 1.0

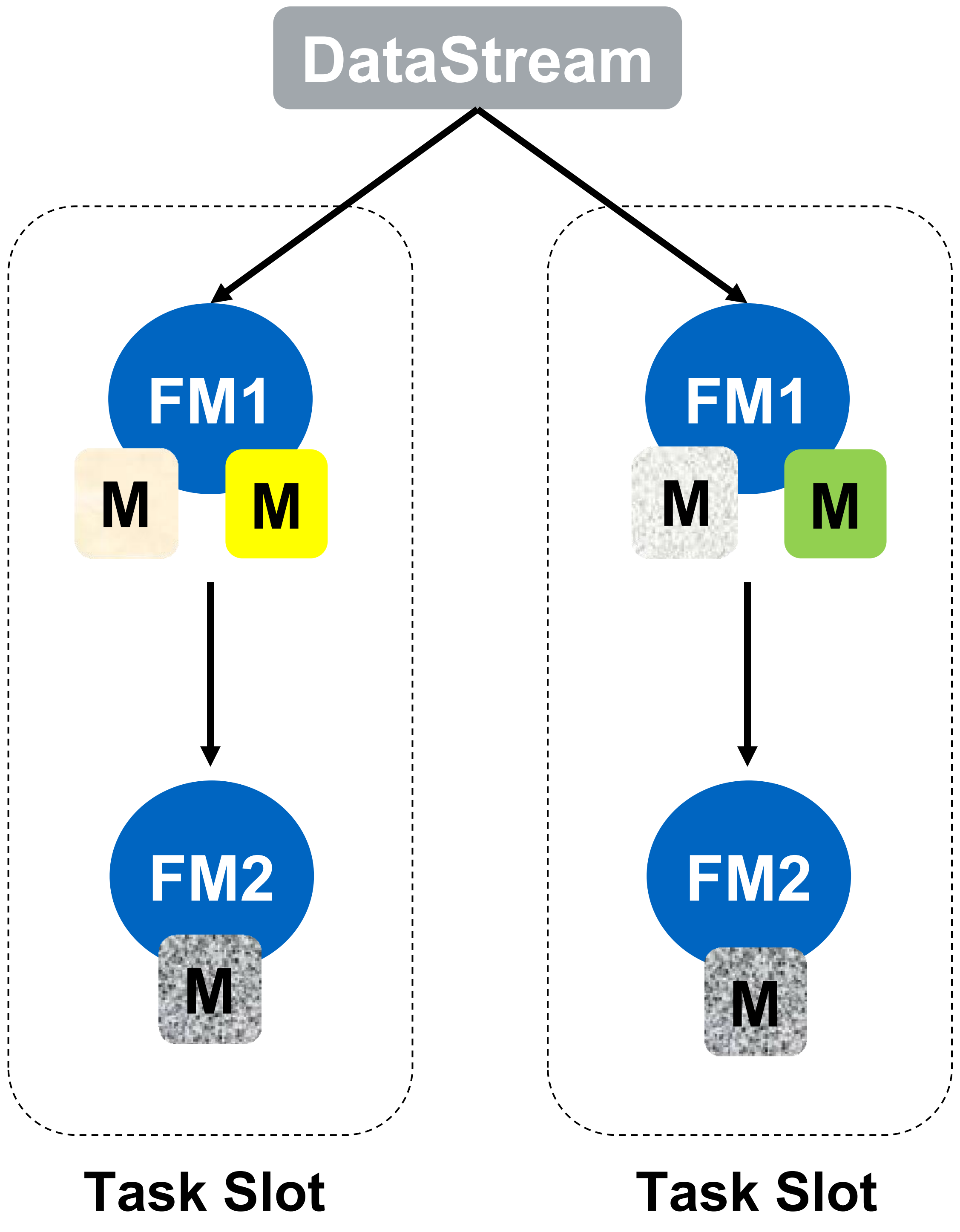
- Built both Offline(Batch) and Online algorithms
  - Batch Algorithms (Examples: KMeans, PCA, and Random Forest)
  - Online Algorithms (Examples: Online KMeans, Online SVM)
- Uses many of the Flink DataStream primitives:
  - DataStream APIs are sufficient and primitives are generic for ML algorithms.
  - CoFlatMaps, Windows, Collect, Iterations, etc.
- We have also added Python Streaming API support in Flink and are working with dataArtisans to contribute it to upstream Flink.

Example: Online SVM Algorithm

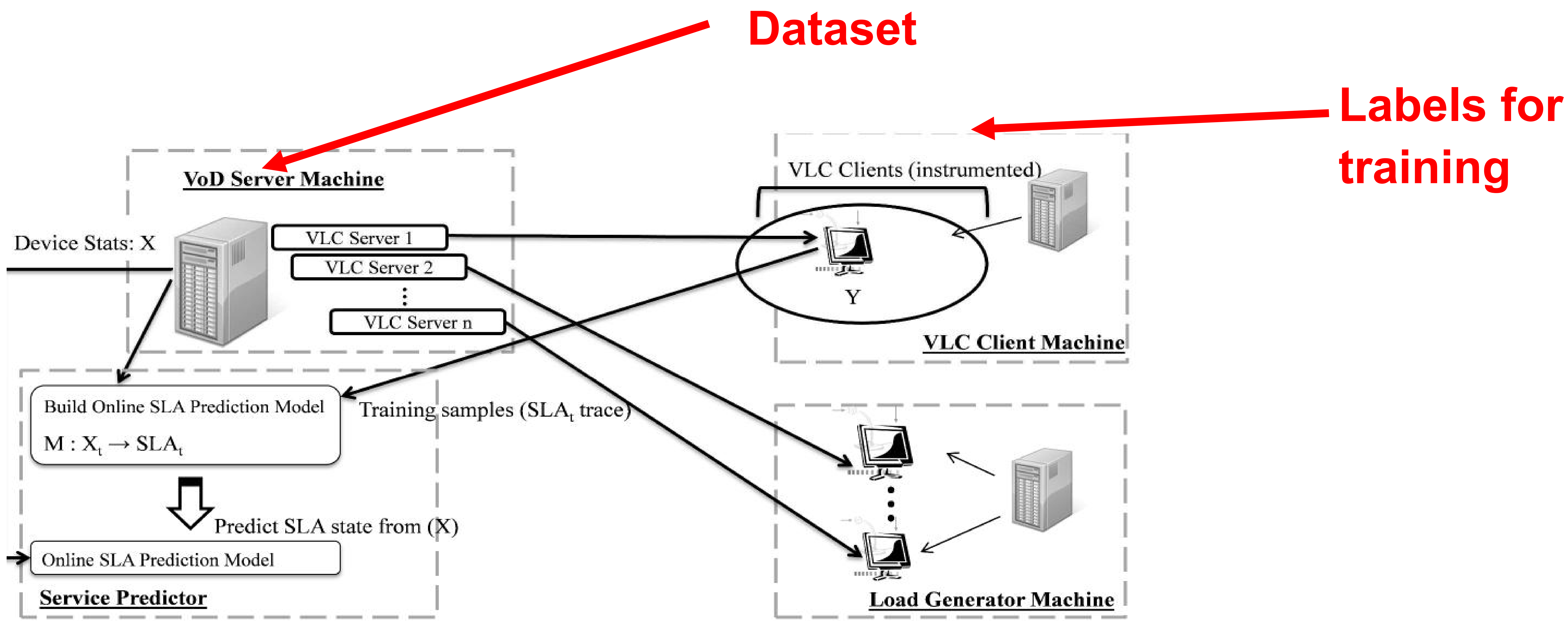
```
/* Co-map to update local model(s) when new data arrives and also
create the shared model when a pre-defined threshold is met */
private case class SVMModelCoMap(...) {
    /* flatMap1 processes new elements and updates local model*/
    def flatMap1(data: LabeledVector[Double],
                 out: Collector[Model]) {
        . . .
    }
    /* flatMap2 accumulates local models and creates a new model
(with decay) once all local models are received */
    def flatMap2(currentModel: Model, out: Collector[Model]) {
        . . .
    }
}

object OnlineSVM {
    . . .
    def main(args: Array[String]): Unit = {
        // initialize  input arguments and connectors
        . . .
    }
}
```

Aggregated and local models combined with decay factor



Telco Example: Measuring SLA Violations



- A server providing VoD services to VLC (i.e., media player) clients
  - Clients request videos of different sizes at different times
  - Server statistics used to predict violations
- **SLA violation:** service level drops below predetermined threshold

Dataset

(https://arxiv.org/pdf/1509.01386.pdf)

CPU Utilization	**Memory/Swap**	**I/O Transactions**	**Block I/O operations**	**Process Statistics**	**Network Statistics**
CPU Idle	Mem Used	Read transactions/s	Block Reads/s	New Processes/s	Received packets/s
CPU User	Mem Committed	Write transactions/s	Block Writes/s	Context Switches/s	Transmitted Packets/s
CPU System	Swap Used	Bytes Read/s			Received Data (KB)/s
CPU IO_Wait	Swap Cached	Bytes Written/s			Transmitted Data (KB)/s
					Interface Utilization %

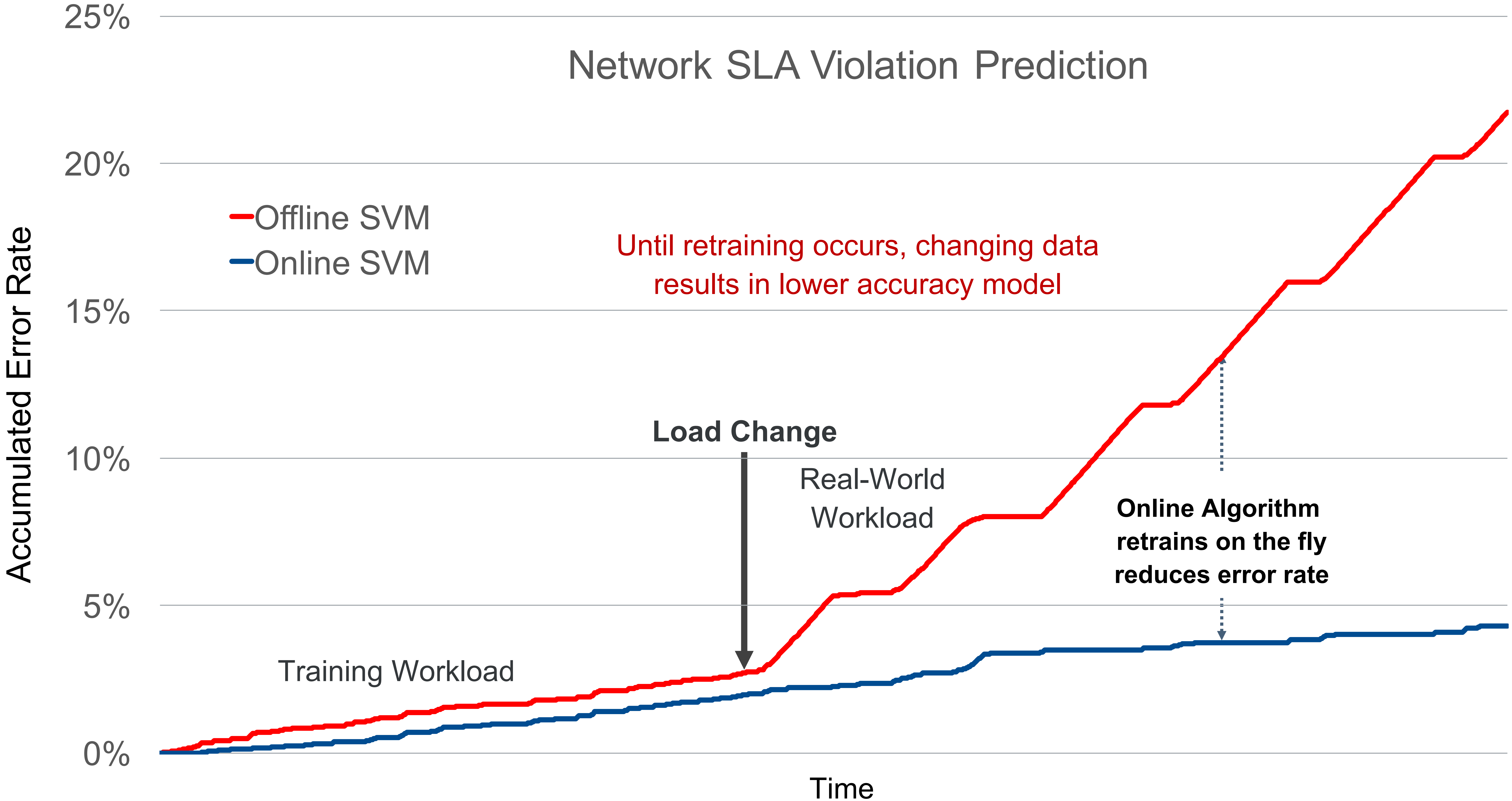
- Load patterns – Flashcrowd, Periodic
- Delivered to Flink and Spark as live stream in experiments

Fixed workloads – Online vs Offline (Batch)

Load Scenario	Offline (LibSVM) Accuracy	Offline (Pegasos) Accuracy	Online SVM Accuracy
flashcrowd_load	0.843	0.915	**0.943**
periodic_load	0.788	0.867	**0.927**
constant_load	0.999	0.999	**0.999**
poisson_load	0.963	0.963	**0.971**

When load pattern remains static (unchanged),
Online algorithms can be as accurate as Offline algorithms

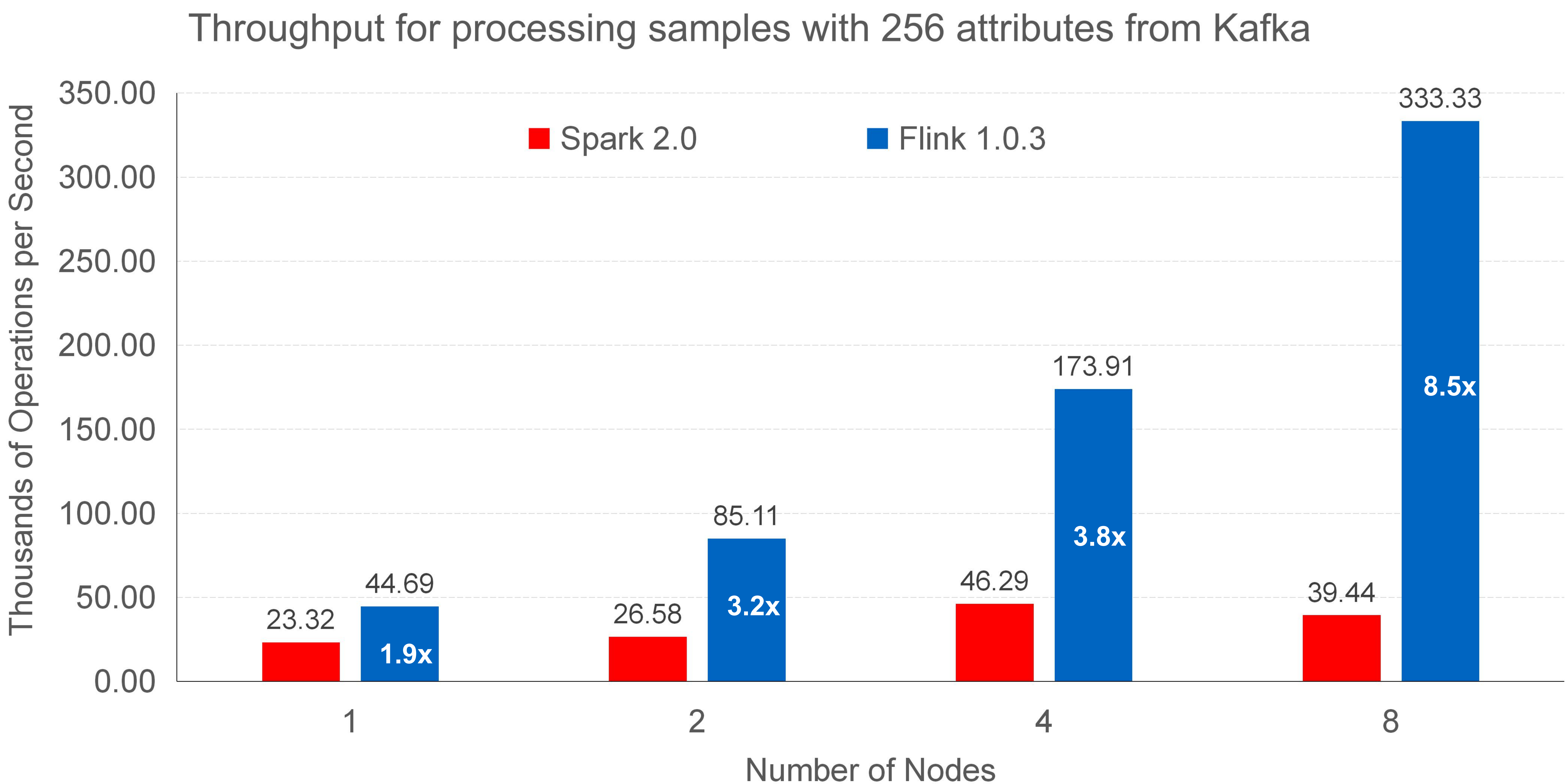
Online SVM vs Batch (Offline) SVM – both in Flink



Online algorithms quickly adapt to workload changes

PARALLEL MACHINES

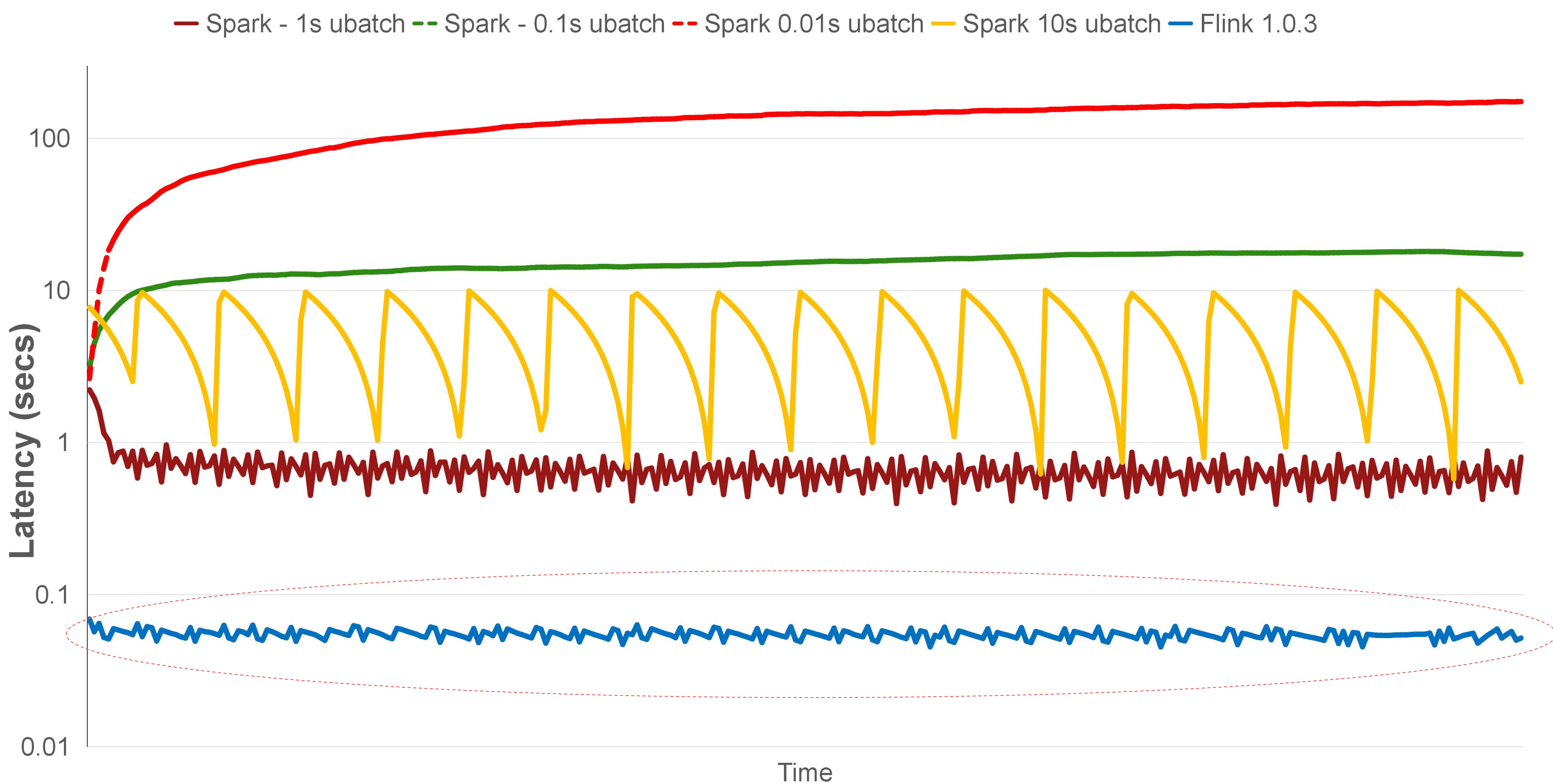
Throughput: Online SVM in Streams and Micro-Batch



Notable performance improvement over micro-batch based solution

PARALLEL MACHINES

Latency: Online SVM in Streams & Micro-batch



Low and predictable latency as needed in Edge

PARALLEL MACHINES

Conclusions

Edge computing & Online learning are needed for real-time analytics

- **Edge Computing:** minimizes the excessive latencies, reaction time
- **Online learning:** can dynamically adapt to changing data / behavior

Online machine learning with streaming on Flink

- Supports *low latency processing* with *scaling* across multiple nodes
- Using real world data, demonstrate improved accuracy over offline algorithms

Parallel Machines
The Machine Learning Management Solution

info@parallelmachines.com