PROTEUS
Scalable Online Machine Learning &
Real-Time Interactive Visual Analytics

www.proteus-bigdata.com
@proteus_bigdata
info@proteus-bigdata.com
It all began with the three Vs: Volume, Velocity and Variety
PROTEUS is about the 4th V: Value
PROTEUS is an EU H2020 funded research project to evolve massive online machine learning strategies for predictive analytics and real-time interactive visualization methods – in terms of scalability, usability and effectiveness dealing with extremely large data sets and data streams – into ready to use solutions, and to integrate them into enhanced version of Apache Flink, the EU Big Data platform.
PROTEUS is an EU H2020 funded research project to evolve massive online machine learning strategies for predictive analytics and real-time interactive visualization methods – in terms of scalability, usability and effectiveness dealing with extremely large data sets and data streams – into ready to use solutions, and to integrate them into enhanced version of Apache Flink, the EU Big Data platform.
CONTENTS

1. PROJECT DETAILS
2. SCALABLE ONLINE MACHINE LEARNING
3. REAL-TIME INTERACTIVE VISUAL ANALYTICS
4. APACHE FLINK
5. USE CASE VALIDATION: STEEL PRODUCTION
6. CONCLUSIONS
1. PROJECT DETAILS
2. SCALABLE ONLINE MACHINE LEARNING
3. REAL-TIME INTERACTIVE VISUAL ANALYTICS
4. APACHE FLINK
5. USE CASE VALIDATION: STEEL PRODUCTION
6. CONCLUSIONS
Project details

- Life cycle

- **2014:** Inception of project idea
- **2015:**
  - Apr 15: Proposal submission
  - Dic 15: PROTEUS kick-off
- **2016:**
  - Aug 15: Notification of acceptance
- **2017**
- **2018:**
  - Nov 18: PROTEUS closing

Duration: 36 months
Project details

**Consortium**

- **Coordinator**
  - ICT company specialised on Big Data & Analytics solutions
  - Creator of Lambdoop

- **The world's leading integrated steel and mining company**
  - End-user
  - Validation scenario

- **ICT start-up specialised on streaming analytics**
  - Cloud-based online machine learning as a Service
  - Evolution of Lambdoop

- **Academic research**
  - Focus on online predictive analytics
  - Institute of Data Science

- **Big contributor to the Apache Flink project**
  - Intelligent analytics for massive data
  - Scientific research

- **Research consultancy**
  - Ethical & Data management
  - Benchmarks and impact assessment
Project details

- Partner contributions & complementarity and innovation chain

**Research**

**Innovation**

**Market**
Project details

Strategy
Project details

- Work Plan

WP1 – Project Management & Coordination

WP2 – Industrial use-case: integration, validation & demonstration

WP3 – Scalable hybrid architectures

WP4 – Scalable online Machine Learning

WP5 – Real-time interactive visualization

WP6 – Impact assessment, exploitation, communication, dissemination

Continuous integration, evaluation, monitoring and guidance
Project details

- Outcomes
  - Hybrid processing
    - Stream processing engine
    - Declarative Language for batch & streams analytics
  - Scalable Online machine Learning
    - SOLMA Library
  - Real-time interactive Visual Analytics
    - Big Data visual guidelines
    - Web charts library
    - Incremental engine
  - Business Impact
    - Integration in Apache Flink
    - Validation in realistic industrial use case
    - Generic KPIs and benchmarks for technology evaluation
1. PROJECT DETAILS
2. SCALABLE ONLINE MACHINE LEARNING
3. REAL-TIME INTERACTIVE VISUAL ANALYTICS
4. APACHE FLINK
5. USE CASE VALIDATION: STEAL PRODUCTION
6. CONCLUSIONS
Scalable Online Machine Learning

What is Machine Learning (ML)?

- It is programming computers to perform an action using example data or past experience → learn from and make predictions on data
- It is used when:
  - Human expertise does not exist (e.g. navigating on Mars)
  - Humans are unable to explain their expertise (e.g. speech recognition)
  - Solution changes in time (e.g. routing on a computer network)
  - Solution needs to be adapted to particular cases (e.g. user biometrics)
Scalable Online Machine Learning

- **ML Terminology**
  - **Observations**: Items or entities used for learning or evaluation (e.g., emails)
  - **Features**: Attributes (typically numeric) used to represent an observation (e.g., length, date, presence of keywords)
  - **Labels**: Values / categories assigned to observations (e.g., spam, not-spam)
  - **Training and Test Data**: Observations used to train and evaluate a learning algorithm (e.g., a set of emails along with their labels)
    - Training data is given to the algorithm for training
    - Test data is withheld at train time
Scalable Online Machine Learning

- Types of ML
  - **Supervised Learning**: Learning from labelled observations
    - Classification
    - Regression / Prediction
    - Recommendation
  - **Unsupervised Learning**: Learning from unlabelled observations. Learning algorithm must find latent structure from features alone.
    - Clustering
    - Dimensionality Reduction
    - Anomaly detection
  - Others
    - Reinforcement learning
    - Semi-supervised learning
    - Active learning

<table>
<thead>
<tr>
<th>Unsupervised</th>
<th>Supervised</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Clustering &amp; Dimensionality Reduction</td>
<td>• Regression</td>
</tr>
<tr>
<td>○ SVD</td>
<td>○ Linear</td>
</tr>
<tr>
<td>○ PCA</td>
<td>○ Polynomial</td>
</tr>
<tr>
<td>○ K-means</td>
<td>○ Decision Trees</td>
</tr>
<tr>
<td>• Association Analysis</td>
<td>○ Random Forests</td>
</tr>
<tr>
<td>○ Apriori</td>
<td></td>
</tr>
<tr>
<td>○ FP-Growth</td>
<td></td>
</tr>
<tr>
<td>• Hidden Markov Model</td>
<td>○ Classification</td>
</tr>
<tr>
<td></td>
<td>○ KNN</td>
</tr>
<tr>
<td></td>
<td>○ Trees</td>
</tr>
<tr>
<td></td>
<td>○ Logistic Regression</td>
</tr>
<tr>
<td></td>
<td>○ Naive-Bayes</td>
</tr>
<tr>
<td></td>
<td>○ SVM</td>
</tr>
</tbody>
</table>
Scalable Online Machine Learning

- ML: Why now?
  - Big Data
    - Flood of data available
    - Internet, Smartphones, IoT, etc.
  - Higher performance of computer
    - Larger memory in handling the data
    - Greater computational power for calculating
  - Growing progress in available algorithms and theory developed by researchers
  - Increasing support from industries
    - Filter spam
    - Customer segmentation
    - Web advertising
    - Face recognition
    - Product recommendation
    - Fraud detection
Scalable Online Machine Learning

- ML challenge: Scalability
  - Classic ML techniques are not always suitable for modern datasets
  - Data grows faster than Moore’s Law
  - Example:
    - Least Squares Regression: Learn mapping \( w \) from features to labels that minimizes residual sum of squares \( \min_w \|Xw - y\|^2 \)
    - Closed form solution \( w = (X^T X)^{-1} X^T y \) (if inverse exists)
  - Computational bottlenecks
    - Matrix multiply of \( X^T X \): \( O(nd^2) \) operations
    - Matrix inverse: \( O(d^3) \) operations
  - Storage bottlenecks
    - \( X^T X \) and it is inverse: \( O(d^2) \) floats
    - \( X \): \( O(nd) \) floats
  - Other methods have similar complexity
Scalable Online Machine Learning

ML challenge: Data Streams

- Current state of the art of machine learning algorithms for Big Data is dominated by offline learning algorithms that process data-at-rest.
- Plenty of current data sources are streaming (online, data-in-motion): sensors, social networks, clickstream, etc.
- In online learning, the algorithms see the data only once. The traditional meaning of online is that data is processed sequentially one by one but for many epochs.

For $t=1, 2, ..., T$
- Receive an instance $X_t$
- Predict its class label $\hat{y}_t = \text{sgn}(f_t(x_t))$
- Receive the true class label $y_t$
- Suffer loss $\ell(y_t, f_t(x_t))$
- Update the prediction model $f_t(x) \rightarrow f_{t+1}(x)$

Goal: To minimize the total loss suffered:
$$\sum_{t=1}^{T} \ell(y_t, f_t(x_t))$$
Scalable Online Machine Learning

- We need scalable methods (using parallel & distributed computing) that are linear in time and space.
- We need algorithms able to adapt complex and fast-changing environment to deal with online data and evolving concepts.
- **SOLMA**: Scalable Online Machine Learning and Data Mining Algorithms
  - Efficient distributed online algorithms for basic utilities, sketches.
  - Advanced online predictive analytics for various tasks like classification, clustering, regression, ensemble methods, and novelty and change detection.
Scalable Online Machine Learning

- **PROTEUS contribution:** **SOLMA**
  - User-friendly
  - Extensibility
    - Basic scalable stream sketches that enable to query the stream
    - Iterative algorithms for approximating the outcome of offline computation
  - Ready-to-use (supervised & unsupervised) online ML algorithms in Apache Flink
1. PROJECT DETAILS
2. SCALABLE ONLINE MACHINE LEARNING
3. REAL-TIME INTERACTIVE VISUAL ANALYTICS
4. APACHE FLINK
5. USE CASE VALIDATION: STEEL PRODUCTION
6. CONCLUSIONS
Real-time Interactive Visual Analytics

- How does Big Data change the nature of data visualization?
  - We use the same charts since 70s! → Tukey’s Exploratory Data Analysis book
  - Streams → Data-in-motion
    - Temporal context
    - Source, space, relevance, etc.

- How to deal with data interaction in Big Data?
  - Data-at-rest → batch processing
    - $O(n^k)$ when $n$ is huge → Not real-time interaction!
  - Data-in-motion → streaming processing
    - Loss of context

- Machine Learning and interactive visualization
  - The combination of human intuition and input using interactive techniques produce better models than automatic techniques
  - Visualization paradigms would help to explain the behavior of the algorithms
Real-time Interactive Visual Analytics

- PROTEUS contribution
  - Definition of new ways of presenting information in order to make the knowledge derived from extremely large and/or streaming data valuable and actionable.
  - Design and implementation of a new software architecture on top of Apache Flink using an incremental approach to achieve low-latency advanced visualizations and interactions.
  - Development of ready-to-use novel web-based visualization library seamless integrated with the proposed architecture implementing the defined Big Data visualization guidelines for disruptive changes in the visual analysis of data.
Real-time Interactive Visual Analytics

- **Data collector**: in charge of iteratively getting new data from data sources (both static and streaming)
- **Incremental Analytics engine**: incremental partial results in ~ $O(1)$
- **Visualization Layer**: web-based library seamlessly connected to the Incremental Analytics engine
1. PROJECT DETAILS
2. SCALABLE ONLINE MACHINE LEARNING
3. REAL-TIME INTERACTIVE VISUAL ANALYTICS
4. APACHE FLINK
5. USE CASE VALIDATION: STEEL PRODUCTION
6. CONCLUSIONS
What is Apache Flink?

- Apache Flink is a Big Data open source platform for scalable batch and stream data processing.
- Started in 2009 by the Berlin-based database research groups (Stratosphere project).
- Accepted as Apache Incubator project in April 2014. Become Apache Top-Level project since December 2014.
- About 120 contributors, highly active community.
What is Apache Flink?

- Massive parallel data flow engine with unified batch and stream processing
  - Batch (DataSet) and Stream (DataStream) APIs on top of a streaming engine
- Rich set of operators (including native iteration)
  - Map, Reduce, Join, CoGroup, Union, Iterate, Delta Iterate, Filter, FlatMap, GroupReduce, Project, Aggregate, Distinct, Vertex-Update, Accumulators, ...
- Programming APIs for Java and Scala (Python upcoming)
- Flink Optimizer
  - Inspired by optimizers of parallel database systems
  - Physical optimization follows cost-based approach
- Memory Management
  - Flink manages its own memory
  - Never breaks the JVM heap
Apache Flink in the Big Data ecosystem

Applications

Data processing engines

App & resource management

Storage & streams
Apache Flink examples

**Batch Wordcount**

```scala
case class Word (word: String, frequency: Int)
val env = ExecutionEnvironment.getExecutionEnvironment()
val lines: DataSet<String> = env.readTextFile(...) 
lines
  .flatMap { line =>
    line.split(" ").map (word => Word(word, 1) )
  }
  .groupBy("word")
  .sum("frequency")
  .print()
env.execute()
```

**Stream windowed Wordcount**

```scala
case class Word(word: String, count: Long)
val input = env.socketTextStream(host, port);
val words = input flatMap {
  line => line.split("\W+").map(Word(_,1))
    .window(Count.of(20)).every(Count.of(10))
}
val counts = words.groupBy("word").sum("count")
```
Apache Flink comparison

<table>
<thead>
<tr>
<th></th>
<th>Flink</th>
<th>Spark</th>
<th>Flink</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>API</strong></td>
<td>low-level</td>
<td>high-level</td>
<td>high-level</td>
</tr>
<tr>
<td><strong>Data Transfer</strong></td>
<td>batch</td>
<td>batch</td>
<td>pipelined &amp; batch</td>
</tr>
<tr>
<td><strong>Memory Management</strong></td>
<td>disk-based</td>
<td>JVM-managed</td>
<td>Active managed</td>
</tr>
<tr>
<td><strong>Iterations</strong></td>
<td>file system cached</td>
<td>in-memory cached</td>
<td>streamed</td>
</tr>
<tr>
<td><strong>Fault tolerance</strong></td>
<td>task level</td>
<td>task level</td>
<td>job level</td>
</tr>
<tr>
<td><strong>Good at</strong></td>
<td>massive scale out</td>
<td>data exploration</td>
<td>heavy backend &amp; iterative jobs</td>
</tr>
<tr>
<td><strong>Libraries</strong></td>
<td>many external</td>
<td>built-in &amp; external</td>
<td>evolving built-in &amp; external</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Streaming</th>
<th>Spark</th>
<th>Flink</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Streaming</strong></td>
<td>“true”</td>
<td>mini batches</td>
<td>“true”</td>
</tr>
<tr>
<td><strong>API</strong></td>
<td>low-level</td>
<td>high-level</td>
<td>high-level</td>
</tr>
<tr>
<td><strong>Fault tolerance</strong></td>
<td>tuple-level ACKs</td>
<td>RDD-based (lineage)</td>
<td>coarse checkpointing</td>
</tr>
<tr>
<td><strong>State</strong></td>
<td>not built-in</td>
<td>external</td>
<td>internal</td>
</tr>
<tr>
<td><strong>Exactly once</strong></td>
<td>at least once</td>
<td>exactly once</td>
<td>exactly once</td>
</tr>
<tr>
<td><strong>Windowing</strong></td>
<td>not built-in</td>
<td>restricted</td>
<td>flexible</td>
</tr>
<tr>
<td><strong>Latency</strong></td>
<td>low</td>
<td>medium</td>
<td>low</td>
</tr>
<tr>
<td><strong>Throughput</strong></td>
<td>medium</td>
<td>high</td>
<td>high</td>
</tr>
</tbody>
</table>
Why Apache Flink is good for PROTEUS?

- Hybrid batch/streaming engine
  - Easy to develop hybrid architectures (e.g. Lambda & Kappa) suitable for the online machine learning algorithms and incremental engine

- Native support for iterations
  - Better performance for incremental updates (models & partial results)

- Easy to use for end-users
  - Little tuning or configuration required

- EU technology
  - Avoid dependency from US IT companies

Lambda Architecture in Apache Flink

Kappa Architecture in Apache Flink
CONTENTS

1. PROJECT DETAILS
2. SCALABLE ONLINE MACHINE LEARNING
3. REAL-TIME INTERACTIVE VISUAL ANALYTICS
4. APACHE FLINK
5. USE CASE VALIDATION: STEEL PRODUCTION
6. CONCLUSIONS
Steel Industry: Hot Strip Mill

- Steel industry is a **key sector** for the European economy
  - Second largest producer in the world, ~ 11% of global output
- Steel life-cycle
  - From material extraction to usage (and recycling)
- Steel production
  - From slabs to coils
- **Hot Strip Mill**
  - Heats the material \( \rightarrow 1200^\circ C \)
  - Laminate the material \( \rightarrow \) high pressure
  - **Real-time sensors** to control the process
- **Coil parameters** \( \rightarrow \) **steel quality**
  - Thickness
  - Width
  - Flatness measurement
Steel Industry: Hot Strip Mill
Steel Industry: Hot Strip Mill

- Preheating furnace
- Breaking-down mill
Hot Strip Mill: needs

- **Predict** coil parameters (thickness, Width, Flatness) using massive streaming real-time data generated during the Hot Strip Mill process
  - The sooner defects are detected, the sooner the process can be modified
- It is necessary to deal with a continuous learning process as steel composition varies continuously, and so does its mechanical behaviour
  - Most of steel grades produced in 2015 did not exist five years earlier
  - Lack of data due to sensor malfunction
- **Visualization methods** for understanding the process
  - Compare online data with massive historical data
- **Objective**: achieve a reduction of 20% of defections coils and reducing rejected material by 15%
Hot Strip Mill: Big Data scenario

- **32-500 ms.** Stream Data Generation
- **7870 Variables**
- **Structure and Unstructured Data**

- **Historical Data**

- **700,000 Registers for Each Variable**
- **500 GB Times Series and Flatness Maps**

- **Flatness Maps**
- **Sensor Data**
- **Time Series**

- **HOT STRIP MILL PROCESS**

- **Flatness Prediction**

- Scalable Online Machine Learning Engine for Big Data
- Real-Time Results
- Integrated in Apache Flink
1. PROJECT DETAILS
2. SCALABLE ONLINE MACHINE LEARNING
3. REAL-TIME INTERACTIVE VISUAL ANALYTICS
4. APACHE FLINK
5. USE CASE VALIDATION: STEAL PRODUCTION
6. CONCLUSIONS
Conclusions

- PROTEUS is an EU H2020 international research project
- PROTEUS will contribute to the Big Data ecosystem with:
  - An innovative hybrid engine for processing both data-at-rest and data-in-motion
  - SOLMA: An new library for scalable online machine learning
  - Big Data Visual guideless: new ways of presenting and working with Big Data
  - Real-time interactive visualization technology: Incremental engine & web-based library
- PROTEUS will be part of the Apache Flink community
- PROTEUS will validate their innovations in a realistic industrial scenario
- PROTEUS will provide full-scale evaluation and impact assessment including benchmarks, KPIs and anonymized datasets
  - Specific metrics for the ArcelorMittal use case
  - Generic indicators on the advancements in scalable machine learning, hybrid computation and real-time interactive visual analytics.
Thanks for your attention!

Questions?

www.proteus-bigdata.com

Contact us:

- info@proteus-bigdata.com
- @proteus_bigdata
- https://github.com/PROTEUS-H2020