**Machine Learning Pipelines with Apache Spark and Intel BigDL**

M. Migliorini\(^1\), V. Khristenko\(^1\)  
M. Pierini\(^1\), E. Motesnitsalis\(^1\), L. Canali\(^1\), M. Girone\(^1\)

\(^1\) CERN, Geneva, Switzerland; \(^2\) University of Padova, Padova, Italy

### End-to-End ML Pipeline
- The goal of this work is to produce a demonstrator of an end-to-end Machine Learning pipeline using Apache Spark.
- Investigate and develop solutions integrating:
  - Data Engineering/Big Data tools
  - Machine Learning tools
  - Data analytics platform
- Use Industry standard tools:
  - Well known and widely adopted
  - Open the HEP field to a larger community
- The Pipeline is composed by the following stages:
  - Data Ingestion
  - Feature Preparation
  - Parameters Tuning
  - Training

### HEP use case
- The ability to classify events is of fundamental importance and Deep Learning proved to be able to outperform other ML methods.
- See paper: “Topology classification with deep learning to improve real time event selection at LHC” (arXiv:1807.00083v2)

### Machine Learning Pipelines
- **W+j** (63.4%)
- **QCD** (36.2%)
- **tt\(\bar{t}\)** (0.4%)

### The Pipeline
- Filter events: require the presence of isolated leptons
- Prepare input for the classifiers
  - Produce multiple datasets
  - Raw data (list of particles)
  - High Level features
- Store results in parquet files
  - Dev. dataset (100k events)
  - Full dataset (5M events)

### Training
- Trained the three models using various hardware and configurations
  - Observed near linear scalability of Intel BigDL
  - Reproduced the classifiers performance of the source paper

### Summary
- Created an end-to-end ML pipeline using Apache Spark
  - Python & Spark allow to distribute computation in a simple way
  - Intel BigDL scales well and it is easy to use because it has a similar API to Keras
  - Interactive analysis made easier by Jupiter Notebooks
- Future work
  - Test pipeline using cloud resources
  - Further performance improvements on data preparation and training
  - Model Serving: implement inference on streaming data

**https://openlab.cern/**

---

**Data Ingestion**
- EOS storage
  - Input:
    - 10 TB of ROOT files
    - 50M events
  - Access physics data stored in EOS using Hadoop-XRootD Connector
  - Read ROOT files into a Spark DF using Spark-ROOT reader

**Feature Preparation**
- Filter events: require the presence of isolated leptons
- Prepare input for the classifiers
  - Produce multiple datasets
  - Raw data (list of particles)
  - High Level features
- Store results in parquet files
  - Dev. dataset (100k events)
  - Full dataset (5M events)

**Training**
- Trained the three models using various hardware and configurations
  - Observed near linear scalability of Intel BigDL
  - Reproduced the classifiers performance of the source paper

**Summary**
- Created an end-to-end ML pipeline using Apache Spark
  - Python & Spark allow to distribute computation in a simple way
  - Intel BigDL scales well and it is easy to use because it has a similar API to Keras
  - Interactive analysis made easier by Jupiter Notebooks
- Future work
  - Test pipeline using cloud resources
  - Further performance improvements on data preparation and training
  - Model Serving: implement inference on streaming data

**https://openlab.cern/**

---

**Parameters Tuning**
- Scan a grid of parameters to find the best model
- Train multiple models at the same time (one per executor)

---

**Model #1**
- Scan a grid of parameters to find the best model
- Train multiple models at the same time (one per executor)

**Model #2**
- Scan a grid of parameters to find the best model
- Train multiple models at the same time (one per executor)

**Model #3**
- Scan a grid of parameters to find the best model
- Train multiple models at the same time (one per executor)