Machine Learning Community White Paper

See page 18 for the full author list

May 29, 2018

Contents

1 Introduction

The main objectives of particle physics in the post-Higgs boson discovery era is to exploit the full physics potential of the Large Hadron Collider (LHC) and its upgrade, the high luminosity LHC (HL-LHC), and present and future neutrino experiments. The HL-LHC will deliver data from 100 times the luminosity compared to the LHC, bringing quantitatively and qualitatively new challenges due to event size, data volume, and complexity. The physics reach of the experiments will be limited by the physics performance of algorithms and computational resources. Machine learning (ML) applied to particle physics promises to provide improvements in both of these areas.

Incorporating machine learning in particle physics workflows will require significant research and development over the next five years. Areas where significant improvements are needed include:

- **Physics performance** of reconstruction and analysis algorithms;
- **Execution time** of computationally expensive parts of event simulation, pattern recognition, and calibration;
- **Realtime implementation** of machine learning algorithms;
- **Reduction of the data footprint** with data compression, placement and access.

1.1 Motivation

The experimental high-energy physics (HEP) program revolves around two main objectives: probing the Standard Model (SM) with increasing precision and searching for new physics. Both tasks require the identification of rare signals in immense backgrounds. Substantially increased levels of pile-up at the HL-LHC will make this a significant challenge.

Machine learning algorithms are already the state-of-the-art in event and particle identification, energy estimation and pile-up suppression applications in HEP. Despite their present advantage, machine-learning algorithms still have significant room for improvement in their exploitation of the full potential of data.

1.2 Brief Overview of Machine Learning Algorithms in HEP

This section provides a brief introduction to the most important machine learning algorithms in HEP, introducing key vocabulary (in *italic*).

Machine learning methods are designed to exploit large datasets in order to reduce complexity and find new features in data. The most frequently used machine learning algorithms in HEP are Boosted Decision Trees (BDTs) and Neural Networks (NN).

Typically, variables relevant to the physics problem are selected and a machine learning *model* is *trained* for *classification or regression* using signal and background events (or *instances*). Training the model is the most human- and CPU- time consuming, while the application, the so called *inference* stage, is relatively inexpensive. BDTs and NNs are typically used to classify particles and events. They are also used for *regression*, where a continuous function is learned, for example to obtain the best estimate of particle's energy based on the measurements from several detectors.

Neural Networks have been used in HEP for some time; however, improvements in training algorithms and computing power have led in the last decade to the so-called Deep Learning revolution, which has made a significant impact on HEP. Deep Learning is particularly promising when there is a large amount of data and features, as well as symmetries and complex non-linear dependencies between inputs and outputs.

There are different types of deep neural networks used in HEP: fully-connected (FCN), convolutional (CNN) and recurrent (RNN). Additionally, neural networks are used in the context of Generative Models, when a Neural Network is trained to mimic multidimensional distributions to generate any number of new instances. Variational AutoEncoders (VAE) and more recent Generative Adversarial Networks (GAN) are two examples of such generative models used in HEP.

A large set of Machine Learning algorithms are devoted to time series analysis and prediction. They are in general not relevant for HEP where events are independent from each other. However, there is more and more interest in these algorithms for Data Quality and Computing Infrastructure monitoring, as well as those physics processes and event reconstruction tasks where time is an important dimension.

1.3 Structure of the Document

Applications of machine learning algorithms motivated by HEP drivers are detailed in Section 2. Section 3 focuses on the machine learning software in HEP and discusses the interplay between internally and externally developed machine learning tools. Recent progress in machine learning was made possible in part by emergence of suitable hardware for training complex models. In Section 4, the resource requirements of training and applying machine learning algorithms in HEP are discussed. Section 5 discusses ways of training the HEP community in machine learning, while Section 6 focuses on outreach and collaboration with the machine learning community. Finally, Section 7 presents the roadmap for the near future.

2 Machine Learning Applications and R&D

This chapter describes the science drivers and high-energy physics challenges where machine learning can play a significant role in advancing the current state of the art. These challenges are selected because of their relevance and potential and also due to similarity with challenges faced outside the field. Despite such similarities, major R&D work will go in adapting and evolving the methods to match the particular HEP requirements.

2.1 Detector Simulation

Particle discovery relies on the ability to accurately compare the observed detector response data with expectations based on the hypotheses of the Standard Model or models of new physics. While the processes of subatomic particle interactions with matter are known, it is intractable to compute the detector response analytically. As a result, Monte Carlo simulation tools, such as GEANT [1], have been developed to simulate the propagation of particles in detectors to compare with the data.

For the HL-LHC, on the order of trillions of simulated collisions are needed in order to achieve the statistical accuracy of the simulations to perform precision hypothesis testing. However, such simulations are highly computationally expensive. For example, simulating the detector response of a single LHC proton-proton collision event takes on the order of several minutes [2]. Particularly time consuming is the simulation of particles incident on the dense material of a calorimeter, the detector used to measure the energy deposited by the particles. Radiative and nuclear interactions result in the production of a multitude of secondary particles, collectively referred to as a shower. The high interaction probability and resulting high multiplicity of particles passing through the dense material make the simulation of such processes highly expensive. This problem is further compounded when particle showers overlap, as in the core of a jet of particles produced by high energy quarks and gluons.

Fast simulation is the process of replacing the slowest components of the simulation chain with computationally efficient approximations. Often such approximations have been done by simplified parameterizations or particle shower look-up tables. These are computationally fast but often suffer from insufficient accuracy for high precision physics measurements and searches.

Recent progress in high fidelity fast generative models, such as GANs and VAEs, which are able to sample high dimensional feature distributions by learning from existing data samples, offer a promising alternative for simulation.

A simplified first attempt at using such techniques saw orders of magnitude increase in simulation speed over existing fast simulation techniques [3], but has not yet reached the required accuracy. Developing these techniques for realistic detector models and understanding how to reach the required accuracy is still needed. The fast advancement in the ML community of such techniques makes this a highly promising avenue to pursue.

Although fast simulation is a necessity, some data analyses will require the highest fidelity simulations from GEANT. There is a large number of parameters that can be used to tune various aspects of the simulation properties. Performing such tuning over high dimensional parameter space is highly non-trivial. Again, machine learning may offer a solution. Modern optimization techniques, such as Bayesian Optimization, allow global optimization of the simulator without the detailed knowledge of its internal details [4]. Applying such techniques to simulation tuning may further improve the output of the simulations.

2.2 Real Time Analysis and Triggering

The traditional approach to data analysis in particle physics assumes that the interesting events recorded by a detector can be selected in real-time (a process known as *triggering*) with a reasonable efficiency, and that once selected, these events can be affordably stored and distributed for further selection and analysis later. However, the enormous production cross-section and luminosities of the LHC mean that these assumptions break down.[1] In particular there are whole classes of events, for example beauty and charm hadrons or low-mass dark matter signatures, which are so abundant that it is not affordable to store all the events for later analysis. In order to

[1] They may well also break down in other areas of high-energy physics in due course.

fully exploit the physics reach of the LHC, it will increasingly be necessary to perform more of the data analysis in real-time [5].

This topic is discussed in some detail in the Reconstruction and Software Triggering chapter, but it is also an important driver of machine learning applications in HEP. Machine learning methods offer the possibility to offset some of the cost of applying reconstruction algorithms, and may be the only hope of performing the real-time reconstruction that enables real-time analysis in the first place. For example, the CMS experiment uses boosted decision trees in the Level 1 trigger to approximate muon momenta. One of the challenges is the trade-off in algorithm complexity and performance under strict inference time constraints. In another example, called the HEP.TrkX project, deep neural networks are trained on large resource platforms and subsequently perform fast inference in online systems.

Real-time analysis poses specific challenges to machine learning algorithm design, in particular how to maintain insensitivity to detector performance which may vary over time. For example, the LHCb experiment uses neural networks for fast fake-track and clone rejection and already employs a fast boosted decision tree for a large part of the event selection in the trigger [6]. It will be important that these approaches maintain performance for higher detector occupancy for the full range of tracks used in physics analyses. Another related application is speeding up the reconstruction of beauty, charm, and other lower mass hadrons, where traditional track combinatorics and vertexing techniques may become too computationally expensive.

In addition, the increasing event complexity particularly in the HL-LHC era will mean that Machine Learning techniques may also become more important to maintaining or improving the efficiency of traditional triggers. Examples of where ML approaches can be useful are triggering of electroweak events with low-energetic objects; improving jet calibration at very early stage of reconstruction allowing jet triggers thresholds to be lowered; or supernovae and proton decay triggering at neutrino experiments.

2.3 Object Reconstruction, Identification, and Calibration

The physical processes of interest in high energy physics experiments occur on time scales too short to be observed directly by particle detectors. For instance, a Higgs boson produced at the LHC will decay within approximately 10^{-22} seconds and thus decays essentially at the point of production. However, the decay products of the initial particle, which are observed in the detector, can be used to infer its properties. Better knowledge of the properties (e.g. type, energy, direction) of the decay products permits more accurate reconstruction of the initial physical process.

Particles are observed in a detector through the energy they deposit when traversing material, which is subsequently digitized. Reconstruction is the process of converting the raw digital signals in the detector into the physical properties of particles. Particle Physics detectors are usually composed of several sub-detectors, each taking advantage of specific interaction mechanism to detect passage of a specific type of particle and measuring its properties. There is a variety of sub-detector technologies, but most belong to one of three categories:

- Tracking Detectors: These detectors measure the trajectory of charge particles by spatially locating ionization. Usually trackers are placed in a magnetic field, so that the particle momentum can be inferred from the curvature of the trajectory. Very precise tracking detectors, such as those that employ silicon, provide sufficient spatial resolution to enable locating the particle creation and/or decay point. The ionization also allows identifying the particle type.
- Calorimeters: These detectors measure the particle energy by causing them to interact and lose the energy in material and counting secondary particles. Highly segmented calorimeters measure the profile of the energy deposition and identify the particle type.
- Particle Identification: These detectors are aimed at determining a specific particle type using a variety of techniques.

Algorithmic reconstruction typically involves several steps that turn the data from the detector electronics (*raw* measurements) into higher level data objects, corresponding to the physical particles that were detected (*features*):

- Feature Extraction: The signal from the passage of particles through a detector element, e.g. a calorimeter cell, is observed above noise in the raw electronic output associated with the element. This signal is then characterized.
- Pattern Recognition: The pattern of signals in geometrically adjacent detector elements is associated with the passage of a signal or group of particles. In calorimeters, this step is commonly referred to as clustering.

- Object Characterization: Properties of the objects are measured. In tracking detectors, this step means fitting a pattern of "hits" to a helix. In calorimeters, this step extracts the energy, location, and other properties of the cluster that for example characterize the shape of the cluster.

- Combined reconstruction: Objects in different detectors are associated together to create a particle candidate.

Machine learning can in principle be applied at any of these steps. For example, experiments have trained ML algorithms on the features from combined reconstruction algorithms to perform particle identification for decades. In the past decade BDTs have been one of the most popular techniques in this domain. More recently, experiments have been able to extract better performance with deep neural networks.

An active area of research is performing particle identification and extracting particle properties on the output of feature extraction with DNNs, in particular for calorimeters or time projection chambers (TPCs), where the data can be represented as a 2D or 3D image and the problems can be cast as a computer vision tasks, in which neural networks are used to reconstruct images from pixel intensities. These neural networks are adapted for particle physics applications by optimizing network architectures for complex, 3-dimensional detector geometries and training them on suitable signal and background samples derived from data control regions. Applications include identification and measurements of electrons and photon from electromagnetic showers, jet properties including substructure and b-tagging, taus and missing energy. Promising deep learning architectures for these tasks include convolutional, recurrent and adversarial neural networks. A particularly important application is to Liquid Argon TPCs (LArTPCs), which is the chosen detection technology for the flagship neutrino program.

For tracking detectors, pattern recognition is the most computationally challenging step. In particular, it becomes computationally intractable for the HL-LHC. The hope is that machine learning will provide a solution that scales linearly with LHC intensity. A current effort called HEP.TrkX investigates deep learning algorithms such as long-term short-term (LSTM) networks for track pattern recognition on many-core processors.

2.4 Sustainable Matrix Element Method

The Matrix Element (ME) Method [7–10] is a powerful technique which can be utilized for measurements of physical model parameters and direct searches for new phenomena. It has been used extensively by collider experiments at the Tevatron for standard model (SM) measurements and Higgs boson searches [11–16] and at the LHC for measurements in the Higgs and top quark sectors of the SM [17–23]. The ME method is based on *ab initio* calculation of the probability density function $\mathcal{P}$ of an event with observed final-state particle momenta $\mathbf{x}$ to be due to a physics process ξ with theory parameters $\boldsymbol{\alpha}$. One can compute $\mathcal{P}_\xi(\mathbf{x}|\boldsymbol{\alpha})$ by means of the factorization theorem from the corresponding partonic cross-sections of the hard-scattering process involving parton momenta $\mathbf{y}$ and is given by

$$\mathcal{P}_\xi(\mathbf{x}|\boldsymbol{\alpha}) = \frac{1}{\sigma_\xi^{\text{fiducial}}(\boldsymbol{\alpha})} \int d\Phi(\mathbf{y}_{\text{final}})\, dx_1\, dx_2\, \frac{f(x_1)f(x_2)}{2sx_1x_2}\, |\mathcal{M}_\xi(\mathbf{y}|\boldsymbol{\alpha})|^2\, \delta^4(\mathbf{y}_{\text{initial}} - \mathbf{y}_{\text{final}})\, W(\mathbf{x},\mathbf{y}) \qquad (1)$$

where and x_i and $\mathbf{y}_{\text{initial}}$ are related by $y_{\text{initial},i} \equiv \frac{\sqrt{s}}{2}(x_i, 0, 0, \pm x_i)$, $f(x_i)$ are the parton distribution functions, $\sqrt{s}$ is the collider center-of-mass energy, $\sigma_\xi^{\text{fiducial}}(\boldsymbol{\alpha})$ is the total cross section for the process ξ (with $\boldsymbol{\alpha}$) times the detector acceptance, $d\Phi(\mathbf{y})$ is the phase space density factor, $\mathcal{M}_\xi(\mathbf{y}|\boldsymbol{\alpha})$ is the matrix element (typically at leading-order (LO)), and $W(\mathbf{x},\mathbf{y})$ is the probability density (aka "transfer function") that a selected event $\mathbf{y}$ ends up as a measured event $\mathbf{x}$. One can use calculations of Eqn. 1 in a number of ways (e.g. likelihood functions) to search for new phenomena at particle colliders.

The ME method brings in several unique and desirable featudres, most notably it (1) does not require training data being an *ab initio* calculation of event probabilities, (2) incorporates all available kinematic information of a hypothesized process, including all correlations, and (3) has a clear physical meaning in terms of the transition probabilities within the framework of quantum field theory.

One drawback to the ME Method is that it has traditionally relied on LO matrix elements, although nothing limits the ME method to LO calculations. Techniques that accommodate initial-state QCD radiation within the LO ME framework using transverse boosting and dedicated transfer functions to integrate over the transverse momentum of initial-state partons have been developed [24]. Another challenge is development of the transfer functions which rely on tediously hand-crafted fits to full simulated Monte-Carlo events.

The most serious difficulty in the ME method that has limited its applicability to searches for beyond-the-SM physics and precision measurements is that it is very *computationally intensive*. If this limitation is overcome, it would enable more widespread use of ME methods for analysis of LHC data. This could be particularly important for extending the new physics reach of the HL-LHC which will be dominated by increases in integrated luminosity rather than center-of-mass collision energy.

Accurate evaluation of Eqn. 1 is computationally challenging for two reasons: (1) it involves high-dimensional integration over a large number of events, signal and background hypotheses, and systematic variations and (2) it involves sharply-peaked integrands[2] over a large domain in phase space. In reference to point (1), the matrix element $\mathcal{M}_\xi(\mathbf{y}|\boldsymbol{\alpha})$ in the method involves all partons in the $n \to m$ process, so when the 4-momentum of particles are not completely measured experimentally (e.g. neutrinos), one must integrate over the missing information which increases the dimensionality of the integration. In reference to point (2), a clever technique to re-map the phase space in order to reduce the sharpness of integrate in that space in an automated way (MADWEIGHT [25]) is often used in conjunction with a matrix element calculation package (MADGRAPH_aMCNLO [26]). In practice, evaluation of definite integrals by the ME approach invokes techniques such as importance sampling (see VEGAS [27, 28] and FOAM [29]) or recursive stratified sampling (see MISER [30]) Monte Carlo integration. Acceleration of some of these techniques on modern computing architectures has been achieved, for example concurrent phase space sampling in VEGAS on GPUs.

Despite the attractive features of the ME method and promise of further optimization and parallelization of the evaluation of Eqn. 1, the computational burden of the ME technique will continue to limit is range of applicability for practical data analysis without new and innovative approaches. The primary idea put forward in this section is to utilize modern *machine learning techniques to dramatically speed up the numerical evaluation of Eqn. 1* and therefore broaden the applicability of the ME method to the benefit of HL-LHC physics.

Applying neural networks to numerical integration problems is plausible but not new (see [31–33], for example). The technical challenge is to design a network which is sufficiently rich to encode the complexity of the ME calculation for a given process over the phase space relevant to the signal process. Deep Neural Networks (DNNs) are strong candidates for networks with sufficient complexity to achieve good approximation of Eqn. 1, possibly in conjunction with smart phase-space mapping such as described in [25]. Promising demonstration of the power of Boosted Decision Trees [34, 35] and Generative Adversarial Networks [36] for improved Monte Carlo integration can be found in [37]. Once a set of DNNs representing definite integrals of the form of Eqn. 1 is generated to good approximation, evaluation of the ME method calculations via the DNNs will be very fast. These DNNs can be thought of as preserving the essence of ME calculations in a way that allows for fast forward execution. They can enable the ME method to be both *nimble* and *sustainable*, neither of which is true today.

The overall strategy is to do the expensive full ME calculations as infrequently as possible, ideally once for DNN training and once more for a final pass before publication, with the DNNs utilized as a good approximation in between. A future analysis flow using the ME method with DNNs might look something like the following: One performs a large number of ME calculations using a traditional numerical integration technique like VEGAS or FOAM on a large CPU resource, ideally exploiting acceleration on many-core devices. The DNN training data is generated from the phase space sampling in performing the full integration in this initial pass, and DNNs are trained either *in situ* or *a posteriori*. The accuracy of the DDN-based ME calculation can be assessed through this procedure. As the analysis develops and progresses through selection and/or sample changes, systematic treatment, etc., the DNN-based ME calculations are used in place of the time-consuming, full ME calculations to make the analysis nimble and to preserve the ME calculations. Before a result using the ME method is published, a final pass using full ME calculation would likely be performed both to maximize the numerical precision or sensitivity of the results and to validate the analysis evolution via the DNN-based approximations.

There are several activities which are proposed to further develop the idea of a Sustainable Matrix Element Method. The first is to establish a cross-experiment group interested in developing the ideas presented in this section, along with a common software project for ME calculations in the spirit of [38]. This area is very well-suited for impactful collaboration with computer scientists and those working in machine learning. Using a few test cases (e.g. $t\bar{t}$ or $t\bar{t}h$ production), evaluation of DDN choices and configurations, developing methods for DNN training from full ME calculations and direct comparisons of the integration accuracy between Monte Carlo and DNN-based calculations should be undertaken. More effort should also be placed in developing compelling applications of the ME method for HL-LHC physics. In the longer term, the possibility of Sustainable-Matrix-Element-Method-as-a-Service (SMEMaaS), where shared software and infrastructure could be used through a common API, is proposed.

2.5 Learning the Standard Model

New physics may manifest itself as unusual or rare events. One approach is to accurately identify the Standard Model processes and search for anomalies. Classifying the Standard Model events is a challenging task, as it consists of many complicated physics processes. Multi-class machine learning algorithms are well-suited for this classification problem. Once an event is classified as likely a known physics process it can be filtered out and remaining events can be further analyzed for hints of new physics. Additionally, unsupervised machine learning techniques can be applied to remaining events to cluster them together. This approach would also be useful in identifying detector problems.

[2] a consequence of imposing energy/momentum conservation in the processes

2.6 Theory Applications

The theoretical physics community has a number of challenges where machine learning can make an impact. These include areas of theoretical model optimization with hundreds of parameters, searches for new models, understanding and estimation of the parton distribution functions and possibly quantum machine learning. The following details one such application: learning of the parton distribution functions with machine learning.

Making progress towards the objectives of the HL-LHC program (see section 1) requires not only obtaining the experimental measurements of the physical processes but also reliable theory inputs to compare to. This becomes increasingly challenging as the experimental data gets more precise. There are numerous examples of phenomenologically relevant processes where the experimental uncertainty is comparable to the estimate of the theoretical uncertainty of the corresponding calculation.

Furthermore, the theory does not predict the value of all the inputs required for the computations (for example the value of the strong coupling constant α_S at the Z mass), and there are situations where the equations resulting from theory cannot be solved to describe the physics adequately, and the corresponding theory inputs must be obtained from data instead. A more complex example is the determination of Parton Distribution Functions (PDFs): Quantum Chromodynamics (QCD) describes the proton collisions at high energy in terms of *partons* (e.g. quarks and gluons), but it is not possible to calculate directly from QCD the momentum carried by each quark or gluon within a proton since QCD is not solvable in its confined regime. Our lack of theoretical knowledge about the characterization of partons within a proton is embedded into a suitable definition of the Parton Distribution Functions (approximately the momentum densities of each of the partons) The PDFs then need to be determined from experimental data. The NNPDF collaboration uses Machine Learning techniques to obtain a PDF determination that is accurate enough to be suitable for high-precision collider data comparison. The NNPDF fitting procedure is described in full details in [39].

The idea is to combine data from all relevant physical processes and fit a neural network representing each PDF. The difficulty of the procedure steams from the fact that multiple experimental inputs need to be combined to obtain a PDF fit. Each of these inputs adjoins only indirect constraints on the PDFs, leaving some regions of the PDF completely unconstrained by data. NNPDF fit includes around 50 datasets from different physical processes, and results that are not always consistent among themselves. Therefore it is crucial to propagate the uncertainty of the experimental inputs into uncertainty on the PDFs.

While the data set is small, each experimental point has an indirect relation to the PDFs, as it is the result of the convolution of one or two PDFs with the corresponding partonic cross section. Code has been developed to compute these convolutions `APFELgrid` [40]. Future research directions include the possibility of using standard ML frameworks to express efficiently the PDF fitting problem. The uncertainties of the theory calculations need to be taken into account as well in the fits. A fully systematic treatment of theory errors in PDFs is a topic of research where Machine Learning could play an important role. The dominant uncertainties in the data are no longer statistical and instead arise from correlated systematics. Determining those systematics accurately is non trivial on the side of the experimental analyses and can have a major impact on the resulting PDFs. The problem grows more complex when ML techniques for which there is no simple recipe to estimate the uncertainty are used extensively in the experimental analysis. Taking full advantage of these advanced methods requires interdisciplinary research and communication on topics such as developing regularization schemes for experimental covariance matrices.

In conclusion, it is not only important to obtain the best fit PDF, but also a reliable estimation of the uncertainty, which in turn requires controlling the uncertainty of the experimental and theoretical inputs.

2.7 Monitoring Detectors, Hardware Anomalies and Preemptive Maintenance

Data-taking of current complex HEP detectors is continuously monitored by physicists taking shifts to monitor the quality of the incoming data. Typically, hundreds of histograms have been defined by experts and shifters are alerted when an unexpected deviation with respect to a reference occurs. It regularly happens that a new type of problem is unseen in a timely manner because it has not been foreseen by the expert.

A whole class of ML algorithms called anomaly detection can be useful for such problems. They are able to learn from data and produce an alert when deviation is seen. By monitoring many variables at the same time such algorithms are sensitive to subtle signs forewarning of imminent failure, so that preemptive maintenance can be scheduled. Such techniques are already used in the industry.

One challenge is that normal drifts in environmental conditions can induce drifts in the data. Beyond just reporting a problem, the natural next step is to connect anomaly detection algorithm to appropriate action: restart an online computer or contact an on-call expert. In the long term, the hardware and data structures of future detectors should be designed to facilitate the operation of anomaly detection algorithms.

2.8 Computing Resource Optimization and Control of Networks and Production Workflows

Data operations is one of the significant challenges for the upcoming HL-LHC. In the current infrastructure, LHC experiments rely on in house solutions for managing the data. While these approaches work reasonably well today, machine learning can help automate and improve the overall system throughput and reduce operational costs.

Machine Learning can be applied in many areas of computing infrastructure, workflow and data management. For example, dataset placement optimization and reduction of transfer latency can lead to a better usage of site resources and an increased throughput of analysis jobs. One of the current examples is predicting the "popularity" of a dataset from dataset usage, which helps reduce disk resource utilization and improve physics analysis time turn-over.

Data volume in data transfers is one of the challenges facing the current computing systems as thousand of users need to access thousands of datasets across the Grid. There is an enormous amount of metadata collected by application components, such as information about failures, file accesses etc. Resource utilization optimization based on this data, including Grid components and software stack layers, can improve overall operations. Understanding the data transfer latencies and network congestion may improve operational costs of hardware resources.

Networks are going to play a crucial role in data exchange and data delivery to scientific applications in HL-LHC era. The network-aware application layer and configurations may significantly affect experiment's daily operations. ML applications can be in network security in identifying anomalies in network traffic; predicting network congestion; bug detection via analysis of self-learning networks, and WAN path optimization based on user access patterns.

2.9 Collaborative Benchmark Datasets

There is a strong incentive for HEP to develop public benchmark datasets, beyond just challenges. Access to a dataset makes the discussion much more concrete and productive. Within the HEP community a common dataset allows to compare algorithms with a much better accuracy and will be very useful for research and development. The same benchmark datasets can also be used for teaching, tutorials and training.

These benchmark datasets could be built based on public simulation engine, or released by experiments within the bounds of their data access policy. Even a small subset of an experiment simulated data can be the base of a very valuable benchmark dataset. For example, the CMS experiment has released a significant amount of its simulated and collected data via the CERN Open Data Portal [41].

To be maximally useful, the subsequent guidelines should be followed when designing a dataset:

- Simplify the dataset as much as possible
- Document the dataset to make it understandable by a non-HEP expert
- Create methodology and metrics for evaluation proposed solutions, and document them
- Prepare an integration plan for incoming ideas and solutions
- Feedback results of successful applications

3 Machine Learning Software and Tools

Machine learning does not exist without software. There are a large variety of algorithms written in different programming languages and general software frameworks that combine many classes of methods into one package. The following sections focus on specific topics and challenges related to machine learning software design in HEP.

3.1 Software Methodology

Presently, there are two machine learning software methodologies in high-energy physics. The first approach focuses on HEP-developed ML toolkits, such as the Toolkit for Multivariate Analysis (TMVA) in ROOT, while the second approach relies on externally developed software, of which there are many examples. Historically, a variety of approaches and competition among them has led to important breakthroughs in the field. On the other hand, having too many choices increases repetition and leads community segmentation and possible issues with reproducibility.

3.2 I/O and Programming Languages

The sheer amounts of data accumulated by HEP experiments require a close look at data access optimization. To train and apply ML techniques on these data, efficient I/O becomes critical, especially for training. I/O performance is very dependent on data formats. Moreover, support for reading data in different formats is required for certain use-cases.

Exploration of new file systems and methods to improve I/O limitations are important and the following R&D studies should take place:

- Explore new file systems to assess I/O limitations;
- Use alternative industry approaches such as Google BigQuery to explore various data access patterns;
- Explore parallel data processing platforms such as Apache Spark for ML training.

Although particle physics has been reliant on C++ over the past decade, the machine learning community has explored other programming languages, in particular the python-based ecosystem.

3.3 Software Interfaces to Acceleration Hardware

Modern machine learning software significantly benefits from using hardware accelerators such as GPUs. At the same time, ML users should not be forced to write platform-dependent code. Various interfaces to different hardware architectures are needed in order to make efficient use of provided computing resources. Emergence of the Open Computing Language (OpenCL) allows programming of high-level interfaces that can run on various hardware platforms.

Machine learning tools often provide different sets of APIs to develop and train the models in one language, and various bindings to use trained models in other programming languages. This is a convenient model for many HEP applications, such as the trigger system, where application latency puts stringent requirements on the software and hardware used.

3.4 Parallelization and Interactivity

Training ML algorithms takes a significant amount of time and parallelization at various levels is desired. For instance, the parallelization of the computations within a single model. Another type of parallelism is data parallelism that targets processing phase of the training with data partitioning and model training using distributed workers. Frameworks like Apache Spark and ideas such as batch training offer promise in this area.

Often one needs to produce many different machine learning models, for example while tuning hyper-parameters or performing k-fold cross-validation, and distribution of these algorithms is key to the reduction of the overall training time.

ML algorithm inference significantly benefits from parallelization as well. For example, in particle physics trigger systems, the stringent latency requirements impose constraints on the type of algorithms that can be easily parallelized in the hardware.

Availability of interactive frameworks, for example Jupyter notebooks, allows for rapid prototype development and testing of ML tools. Such frameworks also ease the connection between the description of models and the data, providing straightforward means of visualizing models and data. HEP has started to exploring interactive frameworks, such as the Service for Web Based Analysis (SWAN). One of the challenges is availability of adequate hardware resources for these systems.

3.5 Internal and external ML tools

Internally developed ML software, such as the Toolkit for Multivariate Analysis (TMVA, [42]), have been developed to apply a variety of machine learning algorithms to HEP challenges. Currently, most published HEP analyses with machine learning have made use of TMVA [2]. There is also software developed in HEP, such as NeuroBayes[43, 44] and RuleFit, that have gained popularity outside of HEP.

At the same time, the ML landscape has evolved and many different ML tools have emerged and gained popularity. There is a growing number of published results based on externally developed tools. The latter, often developed directly by industry for specific applications, are constantly undergoing development, incorporating the latest algorithms from academia. Currently, both internal and external tools are used by the HEP community. TMVA has also undergone significant development in recent years.

In addition, there are smaller tools developed in the HEP community, extending either internal or external ML tools for specific use cases and applications within HEP experiments, such as `hep_ml` [45] or `tmva-branch-adder` [46].

This begs the question: what aspects of ML development and use should the HEP community focus on in the next 5-10 years? There are several aspects to consider including data formats, community size, and interfaces.

Table 1: This table lists various data formats (rows) and ML tools (columns). The ✓ indicates that there is a native solution, while × means that conversion is from one data-format to another is straightforward. The following notations has been used to denote the data-formats: **T** Trees, **F** flat tables, **M** sparse matrices, **R** row-wise arrays, **C** column-wise arrays **S** static data structures

	TMVA	TensorFlow	Theano	Scikit Learn	R	Spark ML	VW	libFM	RGF	Torch
ROOT [**T, C**]	✓									
CSV [**F**]		✓	✓	✓	✓	✓	×	×	×	✓
libSVM [**M**]							×	✓	×	
VW [**M**]							✓			
RGF [**M**]									✓	
NumPy [**R**]		✓	✓	✓	✓	✓	×	×	×	✓
Avro [**S, R**]					✓	✓				
Parquet [**S, C**]					✓	✓				
HDF5 [**S**]		×	×	×						✓
R df [**R**]					✓					

3.5.1 Machine Learning Data Formats

HEP and ML communities currently make use of different data formats. HEP heavily relies on the ROOT software framework for data storage, data processing, and data analysis. The machine learning community uses a large variety of formats, as shown in Figure 1. This figure also shows the relationship of machine learning data formats with ROOT: ROOT file format is very flexible, though requires a significant investment to properly use. Table 1 summarizes the current machine learning toolkits and file formats they use.

3.5.2 Desirable HEP-ML software and data format attributes

A desirable data format should have the following attributes: high read-write speed for efficient training, sparse readability without loading entire dataset into RAM, compression and common use by the machine learning community.

HEP machine learning applications require high performance and flexible algorithms to address the variety of use cases. Some applications, such as triggering, also have to work under tight latency constraints of the order of a few microseconds and below. The data sets are extremely large, which comes with I/O challenges described in section 3.2. This is expected to become even more challenging, as the LHC continues to ramp-up and deliver increasingly large amounts of data.

As discussed in section 3.3, Machine Learning tools use a number of languages. To use them it will be important to offer adequate support. C++ converters or similar tools are also needed to make sure the training result can be efficiently evaluated.

Advantages of using the **external tools** are the size of the community that uses and supports them, being able to easily keep up with progress in the industry and profit from the forefront of the ML research.

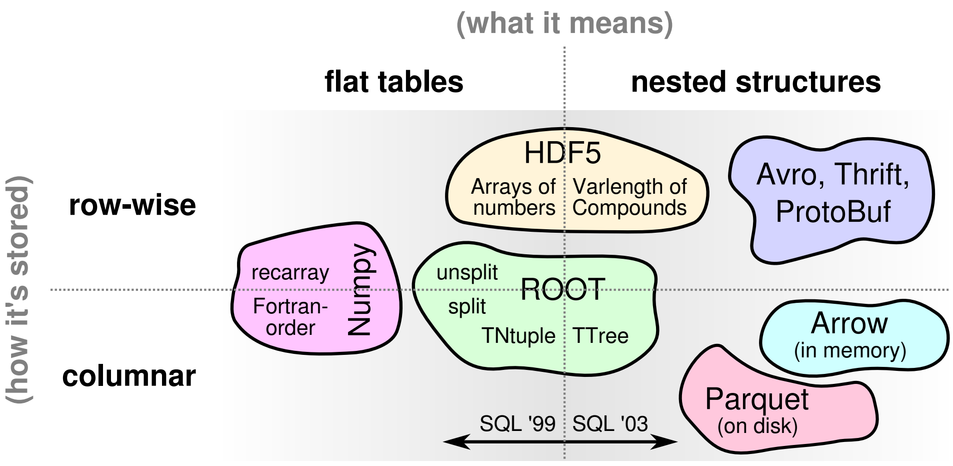

Figure 1: Existing data-formats used by ML communities

Table 2: Middleware solutions translating the ROOT data format to other formats

PyROOT	Python extension module that allows the user to interact with ROOT data/classes [47].
root_numpy	The interface between ROOT and NumPy supported by the Scikit-HEP community [48].
root_pandas	The interface between ROOT and Pandas dataframes supported by the DIANA/HEP project [49].
uproot	A high throughput I/O interface between ROOT and NumPy [50].
c2numpy	Pure C-based code to convert ROOT data into Numpy arrays which can be used in C/C++ frameworks [51].
root4j	The hep.io.root package contains a simple Java interface for reading ROOT files. This tool has been developed based on freehep-rootio [52].
root2npy	The `go-hep` package contains a reading ROOT files. This tool has been developed based on freehep-rootio [52].
root2hdf5	Converts ROOT files containing TTrees into HDF5 files containing HDF5 tables [53].

It should also be noted that some of the recent industrial efforts to develop and maintain ML-tools rely on resources far beyond that of basic research. The deep learning tools of the previous and current generations constitute a demonstration of corresponding quality.

A disadvantage of using external tools: too many choices that are not guaranteed to be supported over the lifetime of particle physics experiments, difficulty of adaptation to HEP specific requirements not among the priorities of the ML community.

Advantages of using **internal tools** are decisions about long-term support remain in the community, can be adapted to specific needs of HEP. Disadvantages include challenges in incorporating new algorithms and ideas on a timely basis and possible lack of resources for long-term maintenance.

3.5.3 Interfaces and middleware

One approach to bridge the gap between internal and external tools is by building interfaces. Some researchers prefer to convert their data to file formats of external tools and work exclusively with external tools. This has the advantage of working as close as possible with the ML community's tools and its documentation, but also ML researchers as possible. At the same time, interfaces have been built between TMVA and external machine learning tools, allowing for their use and direct comparison between their performance. Currently, interfaces to R, scikit-learn, keras and tensorflow have been developed. Those have the advantage of providing a homogeneous interface and require little training overhead for TMVA users.

A more general approach to file format conversion is building middleware solutions that export HEP-specific formats like ROOT to formats used by external machine learning tools. Existing middleware solutions are shown in Table 2.

Approaches to bridge the different languages and data formats inside and outside HEP include providing interfaces or building middleware solutions that translate HEP-specific data formats to external ML tools. It is a topic of current research to determine the most efficient solution.

4 Computing and Hardware Resources

A typical high-energy physics data model consists of a hierarchy of increasingly refined data stores. Each store provides a refined view of a list of ”events”, the self contained records that capture the state of the detector at the time when a particle interaction occurs. At the bottom of the hierarchy is the raw data, a byte-stream of the readout from detector electronics. At the top of the hierarchy are the “high- level” physics objects, such as electrons or jets, providing descriptive information about the quality and topology of physics events. The data stores are typically processed by independent copies of identical code processed in batch computing queues. The result of this processing is filtered data and extracted physics parameters.

At present, training of machine learning algorithms is done using dedicated or private resources. These vary in configuration and processing power, depending on the size of the data and complexity of the algorithm. For a given event, the evaluation of algorithms is performed on a single core producing a single discriminator or regressor output. In order to progress to evaluation of complex machine learning, more computing power is needed in both the training and evaluation stages, as larger amounts of data are needed to feed models with tens or hundreds of thousands of parameters. This implies the expansion of the current computing model to include architectures that are well suited to machine learning tasks, such as many integrated core (MIC), graphical processing units (GPUs) and tensor processing units (TPUs). This is a fundamental departure from the single-core or few-core jobs. These architectures provide a significant computational speed improvement for both training and evaluation of ML algorithms, but require dedicated hardware, drivers, and software configuration.

Similarly, the locality and bandwidth of large data stores will need to be optimized in order to avoid bottlenecks in training and evaluation for analysis. Data placement and the need to use dedicated hardware indicate that a transition to HPC, or HPC-like, architectures may be needed to achieve the desired performance. Due to significant synergy with the direction of industry in this respect, use of commercially available resources should be considered for future high-energy physics computing models.

In the following subsections we will discuss the resource needs for the physics drivers mentioned earlier: fast simulation, real-time analysis, object and event reconstruction and particle identification. The limitations of the current computing model are discussed as well as how those physics driver needs can be met in the future.

4.1 Resource Requirements

The popularity of deep learning methods is to large extent due to the possibility of training these models in a reasonable amount of time with large scale parallelism. In particular, the training stage requires repeated simultaneous access to many data elements and specialized hardware has been developed for training deep learning models.

In contrast, inference can be an operation applied to a single data element at a time and needs to be performed only once. Inference has less demands on I/O and is limited only by the computing power and model complexity. Because inference has real-time applications in high-energy physics, latency and throughput constraints are the main challenges.

4.2 Resource Evaluation

A typical HEP application can require up to 1 GPU-week to train a single model. To obtain the best results and understand the performance of the model, an average of 100 hyper-parameter points optimization is typically performed. A single project could therefore easily require up to a full GPU-year for training.

The speed up of the training process can be obtained by means of faster and more capable hardware, parallelization of single training and over multiple nodes. Resources, such as GPUs, TPUs and MIC need to be evaluated in the context of realistic benchmark particle physics applications.

If different ML techniques can achieve equivalent physics performance but require different processing power, it is important to quantify what is bringing the performance gains and what level of performance-processing power trade-off is is maintainable to achieve the required physics goals.

4.2.1 High Performance Computing

Resource-rich many-core processors such as MIC, GPUs, and TPUs are vital to the optimization of the training time of most modern machine learning algorithms, including deep learning neural networks, generative adversarial networks, autoencoders, etc. Availability of High Performance Computing (HPC) resources equipped with many-core processors and high-performance network storage are essential to distributed running of large-scale machine learning algorithms. Current efforts to bring and expand the availability of HPC resources in high-energy physics computing will be vital to the successful progress of application of machine learning techniques for current and future experiments.

4.2.2 Field Programmable Gate Arrays

Field Programmable Gate Arrays (FPGAs) allow an efficient and low-latent application of machine learning algorithms directly at the level of hardware, as desirable for the high-energy physics trigger systems The following ML algorithms are more suitable for FPGAs due to their simpler parallelization: boosted decision trees, random forests and decision rule ensembles. For example the CMS experiment currently uses boosted decision trees in FPGAs in the trigger system to estimate muon momenta. Further research and development is needed in this area to apply more advanced machine learning techniques like deep learning directly in the hardware. One of the challenges is the limited availability of floating point operations gates and the precision needed to maintain the best performance. The possibility of coupling the FPGAs with a CPU with significant random-access memory (RAM) allows the shift of some of these operations to RAM.

4.2.3 Opportunistic Resources

The current HEP computing model is based on tiered structure where computing resources are mostly large data centers providing CPU resources for collaboration. Although existing resources are gradually moving towards supporting GPUs, it is unlikely to reach all HEP computing centers in the near future. Therefore opportunistic resources are a possible option for training machine learning applications.

Currently, cloud solutions provided by the industry run ML workflows on dedicated hardware and offer interfaces for training machine learning models. The scientific community should work closely with cloud providers to harmonize our analysis computing needs and data access patterns with their business models. Costs of the cloud resources should be compared with the costs of procuring these resources independently.

In order to make the best use of resources available to the community, all resources should ideally be made available through a unique work queue. That implies some uniformization of the software stack, and several specific requirements in the resource management system, especially in terms of data movement.

4.2.4 Data Storage and Availability

Data storage limitations will have a major impact on machine learning applications. Presently, to train machine learning algorithms, it has been possible to take advantage of increases in statistics of Monte-Carlo simulated events needed for other use cases. Further machine learning progress may require more simulated data than what is available today. How to produce and store these additional large amounts of data is a challenge that needs to be overcome.

Availability of data at PByte/EByte scale represents another challenge for ML community. A good solution must provide access to a large data volume for hundred or thousand of users simultaneously. As discussed in the Data Storage chapter [reference], data movement might need to be automatized to make the training data available transparently at high speed local storage with use of automatic caching mechanism. The success of Apache Spark and Google BigQuery platforms may serve as a model. Data streaming, transformation and readout in mini-batches may be required to train models over large data sets. This is in addition to the regular HEP workflows described in the Data Storage and Networking chapter [reference]

4.2.5 Software Distribution and Deployment

To efficiently use the resources described in previous subsections, machine learning software needs to be available on the computing resources. Platforms, such as CERNVM File System (cvmfs), are very useful for software distribution that does not require local installation. Additional tools like docker containers for application shipping can be useful in providing homogeneous software environments across the different systems. Another challenge is that the software layer needs to be agnostic to the hardware back-end.

4.2.6 Machine Learning As a Service

Current cloud providers rely on machine learning as a service model allowing for efficient use of common resources and use interactive machine learning tools. Machine Learning As a Service is not yet widely used in HEP, but examples of successful publications which used Machine Learning As a Service exist, e.g. [54]. Specialized HEP services for interactive analysis, such as CERN's Service for Web-based Analysis (SWAN) may play an important role in adoption of machine learning tools in HEP workflows. In order to use these tools more efficiently, sufficient and appropriately tailored hardware resources described in this chapter are needed.

5 Training the community

In order to address the communication barrier and to speak the same language, the HEP community should be trained in ML concepts and terminology as part of a standard curriculum. The training should focus on well-maintained and well-documented software packages. It should provide lectures on general ML concepts and hands-on tutorials on specific tools based on concrete examples.

Being able to apply machine learning to practical HEP problems requires the understanding of basic ML concepts and algorithms. For this, regular data science lecture series and seminars, like [55], are very useful. At the university level, courses dedicated to machine learning applications in physics research is an excellent way to train undergraduate and graduate students. For example, the "Deep Learning in Physics Research" course with 60 participants consisting of 12 lectures and exercises which are performed on 20 GPUs of the VISPA internet platform [56].

Experiments currently have training activities for newcomers that focus on analysis software and introduction to domain knowledge [57]. Machine learning should next be incorporated into the incoming collaborators training efforts of the experiments.

As discussed in the training chapter, ensuring the development and availability of resources for knowledge transfer is likewise essential to ML.

6 Collaborating with other communities

6.1 Introduction

Discovery science provides a challenge that attracts brilliant minds eager to push the boundaries of scientific understanding of nature. Particle physics has a rich problem domain that offers avenues for intellectual reward. The goal is to achieve vibrant collaboration between data science and high-energy physics communities by finding a common language and working together to further science.

Both communities can benefit from such collaboration. The HEP community can explore new research directions and applications of machine learning, novel algorithms, and direct collaboration on HEP challenges. The ML community can benefit from a diverse set particle physics problems with unique challenges in scale and complexity, and a large community of researchers that can expand machine learning horizon by contributing to solving problems relevant to both communities. For example, the treatment of systematic uncertainties is an important topic for HEP and ML communities. By working together on common challenges the two fields can further progress in solving such problems.

There are a number of existing examples of collaboration between HEP and ML that have produced fruitful results through mostly local connections (e.g. [3, 58]). The HEP community should continue such collaborations and look for additional collaborations with ML.

Domain knowledge can present a barrier to collaboration. The HEP community needs to define its challenges in a language that the ML community can understand. This may involve stripping the domain knowledge entirely, or retaining necessary information with clear and concise explanations as to its relevance. Machine learning likewise has a significant amount of domain knowledge. Ideas and solutions provided by both communities should be presented in an understandable way for scientists without in-depth knowledge.

6.2 Machine Learning Challenges

To engage the wider ML community, challenges such as the Higgs Boson Challenge (2014) or the Flavor Physics Challenge [59, 60] have been organized on Kaggle. These types of challenges draw considerable attention from the Machine Learning community and more of such challenges should be organized in the future.

Organization of a challenge requires a well documented dataset, a starting-kit and an evaluation metric to rank the solutions. This forces the organizers to simplify the problem as much as possible, while retaining its intrinsic complexity.

The drawback of challenges is that once they are launched, participants priority is winning the challenge and not eventual collaboration with HEP. It is important to foresee upfront a way to integrate incoming solutions, for example via forums and post-challenge workshops (like [60]) where a diversity of competitive algorithms can be presented. The challenge dataset and evaluation metric should be released publicly so that further developments can continue.

6.3 Collaborative Benchmark Datasets

As discussed in section 2.9, collaborative benchmark datasets can be useful for developing ML challenges and for collaboration with the ML community. The HEP community should organize and curate a variety of such benchmark datasets covering its current physics drivers and make it publicly available. To improve reproducibility of results and algorithm comparisons, some of the data used for evaluation of the solutions should be kept private.

Additionally, after investing heavily into producing highly-detailed and realistic simulations, the HEP community can provide the machine learning community with labeled datasets with high statistical power to test algorithms and develop novel ideas.

6.4 ML Academic outreach

Conferences and workshops are a core aspect of the academic ML community, and organizing or contributing to key conferences is a means of gaining interest. Organizing sessions or mini-workshops within major ML conferences, such as NIPS, would increase the familiarity of HEP within the ML community and jump-start future collaborations. This has been explored in single cases [59] but is not an established, regular workshop series. At the same time inviting ML experts to HEP workshops as done at [60] and the DS@HEP series [61–63], can foster greater long-term collaboration.

- Organize workshops and conferences open to external collaborators to discuss the applications, algorithms and tools
- Organize thematic workshops around topics relevant to HEP

6.5 Science outreach

HEP should reach out to other scientific communities with similar challenges, for example astrophysics/cosmology, medium energy nuclear physics and computational biology. This can lead to more active partnerships to better collaborate on ideas, techniques, and algorithms.

6.6 Industry Engagement

Industry has been focused on development and adoption of machine learning techniques. In addition to algorithm and software development, one of the promising areas is the adoption of dedicated specialized hardware and high performance co-processors. GPUs, FPGAs, and high core count co-processors all have the potential to dramatically increase performance of machine learning applications relevant to HEP applications.

One of the challenges is gaining the human expertise for development and implementation. Industry brings specific technology opportunities and access to specialized expertise that can be difficult to hire and support internally.

There are specific areas of development where industry has expressed interest in collaborating with HEP. Automated resource provisioning, data placement, and scheduling are similar to industrial applications to improve efficiency. Applications such as data quality monitoring, detector health monitoring and preventative maintenance can be automated using techniques developed for other industrial quality control applications. There are two more forward looking areas that coincide with HEP physics drivers, such as computer vision techniques for object identification and real-time event classification. These present a challenge to industry due to its complexity and benefit outside of HEP.

6.6.1 CERN OpenLab and research-industry collaborative initiatives

CERN OpenLab is a public-private partnership that accelerates the development of cutting-edge solutions for the LHC community and wider scientific research. CERN OpenLab has established the infrastructure to maintain non-disclosure agreements, to arrange ownership of intellectual property and provides an interface between CERN and industry. As part of its upcoming phases, OpenLab plans to explore machine learning applications for the benefit of LHC experiments computing and the HL-LHC. Such initiatives and industry partnerships should be supported in the future.

6.7 ML community at large outreach

Another form of engagement is using the communications mediums to broadcast our challenges and attract interested collaborators. There are a variety of channels which can be leveraged to increase the visibility of our problems and research opportunities in the ML community. These can be popular forums such as reddit, personal or official blogs, social media, and direct contact with influential personalities.

Podcasts have shown to be a great vehicle for reaching a large audience. Listeners are keen to consume material that is outside of their immediate problem domain in a way that is easy to digest. There is an abundance of machine learning podcasts with a large base of listeners that can be targeted for outreach:

- Linear Digressions (co-hosted by former ATLAS Ph.D. Katie Malone)
- Partially Derivative
- Talking Machines
- Data Skeptic
- Becoming a Data Scientist Podcast
- Not So Standard Deviations
- This Week in ML & AI

Another form of engagement is through outreach-style blog posts to explain HEP challenges in a way that is easy to understand by the public.

Another outreach opportunity is to make HEP related presentations at Machine Learning Meetups across the world to generate awareness, engage community, foster cross pollination of ideas between HEP and industry. Some popular ML meetups are:

- NYC: `https://www.meetup.com/NYC-Machine-Learning/`
- Berlin: `https://www.meetup.com/Advanced-Machine-Learning-Study-Group/`

- SF: https://www.meetup.com/SF-Bayarea-Machine-Learning/

In conclusion, existing outreach efforts should be expanded to attract greater collaboration between the HEP and ML communities. By understanding and speaking the same language, the two communities can better collaborate and find solutions to present and future challenges.

7 Roadmap

The incorporation of ML into particle physics experiments must respect two time lines: the schedule of HL-LHC and funding agencies, and the experiments' need for extensive validation of the algorithms.

The current LHC schedule has Run 3 starting in 2021 and the HL-LHC, if approved, starting in 2026. As software processes and algorithms are re-imagined, their implementation must fit into these time frames if they are to maximize their benefit to the physics. To fit this schedule, a newly proposed implementation would need to show a demonstration in 2018 to prove viability. Two years later, in 2020, the idea needs to attain a level of maturity to be included in the HL-LHC Technical Design Report. The project should then be further refined towards a large scale test around the middle of Run 3, about 2022. Run 3 is scheduled to end in late 2023. The project must then be adapted to the HL-LHC software and physics analysis environment as it will be relied on by the experiment.

The path of taking a ML idea from conception to community-wide acceptance and deployment will entail several stages, as appropriate. There are ample opportunities to make the process more efficient. For example, in many steps having common data sets, as discussed in Section 2.9 will likely accelerate the progress.

1. Problem formulation and data set preparation: Problem formulation is the first step in building an ML algorithm. The inputs and desired output need to be established. The training and validation data sets must be identified and simulated. In many cases, these data sets are large, and resources must be identified to possibly create and store the data. In most cases, the data needs to be processed into a form suitable for input into the algorithm. Since these steps are often lengthy, common data sets with well-defined problems will be very helpful

2. Feasibility/Demonstration: Given a dataset, appropriate ML algorithms need to be investigated and evaluated for ability to solve the problem. In some cases, such studies can be preformed on simplified data sets

3. First application: An application of the solution to one or few specific physics analysis where the ML technique significantly improves the physics result. Here the incorporation of the technique into the computing work-flow will likely be very specific to the application and require significant manual intervention

4. Scaling/Optimization: Evolving from a demonstration to a general solution requires use of realistic data sets with full detector simulation, noise, etc. Furthermore, the solution will also require optimization to achieve nominal physics and computing performance. A good practice would be to apply the solution to a specific physics analysis. This stage will likely require significant computing resources to scale solutions to the full detector and data sets

5. Integration/Validation: The solution needs to be incorporated into the experimental software and work-flow and validated.

As an example, consider the simulation physics driver. An effort has recently started to build generative models that can significantly accelerate simulation of particle showers in calorimeters. These early efforts are based on simplified data sets specifically created for this problem, without the complications of realistic data and limited to a small section of calorimeters. The first papers[3] use GANs to generate calorimetric data which are reasonably faithful, but still require tuning. The next step involves exploration of DNN architecture and systematic hyper-parameter scans on HPCs to achieve the required performance. The technique can be applied to searches at LHC that involve boosted objects, where the required simulation samples require CPU intensive full GEANT-based simulation and are therefore limited in statistics due to resource limitations. The process of employing the new technique in a publication will elicit scrutiny by the full experiment, effectively validating the technique. Once the technique is accepted, it can be generalized beyond this first application and then incorporated into the experiment's software for use by others. Finally, as the technique is applied to an increasing number of physics analyses, the technique will be incorporated into the experiment's production work-flows.

A Author list

- Aaron Sauers
- Aashrita Mangu (CS)
- Adam Aurisano (NOvA)
  University of Cincinnati (US)
- Adrian Bevan (ATLAS)
  University of London (GB)
- Alessandra Forti (ATLAS)
  University of Manchester (GB)
- Alexander Kurepin (ALICE)
  Russian Academy of Sciences (RU)
- Alexander Radovic (NOvA)
  College of William and Mary (US)
- Alexei Klimentov (ATLAS)
  Brookhaven National Laboratory (US)
- Amir Farbin (ATLAS)
  University of Texas at Arlington (US)
- Andrey Ustyuzhanin (Yandex, LHCb)
  Yandex School of Data Analysis (RU)
- Antonio Limosani (ATLAS)
  University of Melbourne (AU)
- Ariel Schwartzman (ATLAS)
  SLAC National Accelerator Laboratory (US)
- Attilio Picazio (ATLAS)
  University of Massachusetts (US)
- Aurelius Rinkevicius (CMS)
  Cornell University (US)
- Ben Hooberman (ATLAS)
  University of Illinois at Urbana-Champaign (US)
- Benedikt Hegner (SFT)
  CERN
- Bob Stienen
  Radboud Universiteit Nijmegen (NL)
- Claire David (ATLAS)
  Deutsches Elektronen-Synchrotron (DESY) (DE)
- Conor Fitzpatrick (LHCb)
  Ecole Polytechnique Federale de Lausanne (CH)
- Daniele Bonacorsi (CMS)
  Universita e INFN, Bologna (IT)
- Dario Menasce (CMS and INFN)
  Universita & INFN, Milano Bicocca (IT)
- David Rousseau (ATLAS)
  Universite de Paris Sud 11 (FR)
- Dick Greenwood (ATLAS)
  Louisiana Tech University (US)

- Dorian Kcira (CMS)
  California Institute of Technology (US)
- Douglas Davis (ATLAS)
  Duke University (US)
- Dustin Anderson (CMS)
  California Institute of Technology (US)
- Eduardo Rodrigues (LHCb)
  University of Cincinnati (US)
- Elias Coniavitis (ATLAS)
  Universitaet Freiburg (DE)
- Federico Carminati (SFT)
  CERN
- Fernanda Psihas (NOvA)
  Indiana University (US)
- Filip Siroky (CMS)
  Masaryk University (CZ)
- Gabriel Perdue (MINERvA)
  Fermi National Accelerator Laboratory (US)
- Gaurav Kaul
  Intel
- Giles Strong (CMS)
  LIP Lisboa (PT)
- Gilles Louppe (ATLAS)
  New York University (US)
- Gordon Watts (ATLAS)
  University of Washington (US)
- Graeme Stewart (ATLAS)
  University of Glasgow (GB)
- Hans Pabst
  Intel
- Harvey Newman (CMS)
  California Institute of Technology (US)
- Helge Meinhard
  CERN
- Horst Severini (ATLAS)
  University of Oklahoma (US)
- Ian Stockdale
- Igor Lakomov (ALICE)
  CERN
- Ilija Vukotic (ATLAS)
  University of Chicago (US)
- Jamal Rorie (CMS)
  Rice University (US)
- Javier Duarte (CMS)
  Fermilab (US)

- Jean-Roch Vlimant (CMS)
  California Institute of Technology (US)
- Jim Kowalkowski
  Fermi National Accelerator Laboratory (US)
- Jim Pivarski (CMS)
  Princeton University (US)
- Jochen Gemmler (Belle2)
  Karlsruher Institut für Technologie (DE)
- Johannes Junggeburth
  Max Planck Institut für Physik (DE)
- John Harvey
  CERN
- Jonas Eschle (LHCb)
  Universität Zürich (CH)
- Jonas Graw
- Jordi Garra-Tico (LHCb)
  University of Cambridge (GB)
- Juan Pedro Araque Espinosa (ATLAS)
  LIP Lisboa (PT)
- Karen Tomko
  Ohio Supercomputer Center (US)
- Kevin Lannon (CMS)
  University of Notre Dame (US)
- Kim Albertsson (ATLAS)
  Lulea University of Technology (SE)
- Konstantin Kanishchev (AMS-02)
  Universita e INFN, Padova (IT)
- Konstantin Skazytkin (ALICE)
  Russian Academy of Sciences (RU)
- Kyle Cranmer (ATLAS)
  New York University (US)
- Laurent Basara (RE1)
  Universita e INFN, Padova (IT)
- Lindsey Gray (CMS)
  Fermi National Accelerator Laboratory (US)
- Lorenzo Moneta (ROOT, CMS)
  CERN
- Louis Capps
- Lukas Heinrich (ATLAS)
  New York University (US)
- Luke Kreczko (CMS, LZ)
  University of Bristol (GB)
- Maria Girone (CERN openlab, CERN)
- Mario Campanelli (ATLAS)
  University of London (GB)

- Mario Lassnig (ATLAS)
  CERN
- Mark Neubauer (ATLAS)
  University of Illinois at Urbana-Champaign (US)
- Martin Erdmann (CMS)
- Martin Vala (ALICE)
  Technical University of Kosice (Slovakia)
- Matthew Feickert (ATLAS)
  Southern Methodist University (US)
- Mauro Verzetti (CMS)
  University of Rochester (US)
- Meghan Kane
  SoundCloud, formerly Massachusetts Institute of Technology (US)
- Michael Andrews (CMS)
  Carnegie-Mellon University (US)
- Michael Kagan (ATLAS)
  SLAC National Accelerator Laboratory (US)
- Michael Williams (LHCb)
  Massachusetts Institute of Technology (US)
- Michela Paganini (ATLAS)
  Yale University (US)
- Michele Floris (ALICE)
  CERN
- Mike Sokoloff (LHCb)
  University of Cincinnati (US)
- Nicolas Köhler
  Max-Planck-Institut fur Physik (DE)
- Nuno Filipe Castro (ATLAS)
  LIP-Lisbon (PT)
- Paolo Calafiura (ATLAS)
  Lawrence Berkeley National Lab (US)
- Paul Glaysher (ATLAS)
  Deutsches Elektronen-Synchrotron (DESY) (DE)
- Paul Seyfert (LHCb)
  CERN
- Pere Mato (SFT, LHCb)
  CERN
- Piero Altoe
  NVidia
- Przemysław Karpiński (CERN openlab)
  CERN
- Rob Kutschke (Mu2e, Intensity Frontier)
  Fermi National Accelerator Laboratory (US)
- Ryan Reece (ATLAS)
  University of California, Santa Cruz (US)

- Savannah Thais (ATLAS)
  Yale University (US)
- Sean-Jiun Wang (CMS)
  University of Florida (US)
- Sergei Gleyzer (CMS)
  University of Florida (US)
- Seth Moortgat (CMS)
  Vrije Universiteit Brussel (BE)
- Sofia Vallecorsa (SFT)
  Gangneung-Wonju National University (KR)
- Stefan Wunsch (CMS)
  Karlsruher Institut für Technologie (DE)
- Stefano Carrazza
  CERN
- Steven Schramm (ATLAS)
  Université de Genève (CH)
- Taylor Childers (ATLAS)
  Argonne National Laboratory (US)
- Thomas Keck (Belle2)
  Karlsruher Institut für Technologie (DE)
- Tobias Golling (ATLAS)
  Université de Genève (CH)
- Tom Hacker
- Uzziel Perez (CMS)
- Valentin Kuznetsov (CMS)
  Cornell University (US)
- Vladimir Vava Gligorov (LHCb)
  Centre National de la Recherche Scientifique (FR)
- Wahid Bhimji (NERSC)
- Wenjing Wu (ATLAS)
  Chinese Academy of Sciences (CN)
- Xavier Vilasís-Cardona (LHCb)
  University of Barcelona (ES)
- Omar Zapata (http://oproject.org)
- Zahari Kassabov
  University of Turin (IT)
  University of Milan (IT)

References

[1] S. Agostinelli et al. "GEANT4: A Simulation toolkit." *Nucl. Instrum. Meth.* A506 (2003), pp. 250–303. DOI: 10.1016/S0168-9002(03)01368-8.

[2] *Citation needed.*

[3] M. Paganini, L. de Oliveira, and B. Nachman. "CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks" (2017). arXiv: 1705.02355 [hep-ex].

[4] P. Ilten, M. Williams, and Y. Yang. "Event generator tuning using Bayesian optimization." *JINST* 12.04 (2017), P04028. DOI: 10.1088/1748-0221/12/04/P04028. arXiv: 1610.08328 [physics.data-an].

[5] S. Benson et al. "The LHCb Turbo Stream." *Journal of Physics: Conference Series* 664.8 (2015), p. 082004. URL: http://stacks.iop.org/1742-6596/664/i=8/a=082004.

[6] V. V. Gligorov and M. Williams. "Efficient, reliable and fast high-level triggering using a bonsai boosted decision tree." *JINST* 8 (2013), P02013. DOI: 10.1088/1748-0221/8/02/P02013. arXiv: 1210.6861 [physics.ins-det].

[7] K. Kondo. "Dynamical Likelihood Method for Reconstruction of Events With Missing Momentum. 1: Method and Toy Models." *J. Phys. Soc. Jap.* 57 (1988), pp. 4126–4140. DOI: 10.1143/JPSJ.57.4126.

[8] F. Fiedler et al. "The Matrix Element Method and its Application in Measurements of the Top Quark Mass." *Nucl. Instrum. Meth.* A624 (2010), pp. 203–218. DOI: 10.1016/j.nima.2010.09.024. arXiv: 1003.1316 [hep-ex].

[9] I. Volobouev. "Matrix Element Method in HEP: Transfer Functions, Efficiencies, and Likelihood Normalization." *ArXiv e-prints* (Jan. 2011). arXiv: 1101.2259 [physics.data-an].

[10] F. Elahi and A. Martin. "Using the modified matrix element method to constrain $L_\mu - L_\tau$ interactions." *Phys. Rev.* D96.1 (2017), p. 015021. DOI: 10.1103/PhysRevD.96.015021. arXiv: 1705.02563 [hep-ph].

[11] V. M. Abazov et al. "A precision measurement of the mass of the top quark." *Nature* 429 (2004), pp. 638–642. DOI: 10.1038/nature02589. arXiv: hep-ex/0406031 [hep-ex].

[12] A. Abulencia et al. "Precision measurement of the top quark mass from dilepton events at CDF II." *Phys. Rev.* D75 (2007), p. 031105. DOI: 10.1103/PhysRevD.75.031105. arXiv: hep-ex/0612060 [hep-ex].

[13] T. Aaltonen et al. "First Measurement of ZZ Production in panti-p Collisions at $\sqrt{s} = 1.96$-TeV." *Phys. Rev. Lett.* 100 (2008), p. 201801. DOI: 10.1103/PhysRevLett.100.201801. arXiv: 0801.4806 [hep-ex].

[14] T. Aaltonen et al. "Inclusive Search for Standard Model Higgs Boson Production in the WW Decay Channel using the CDF II Detector." *Phys. Rev. Lett.* 104 (2010), p. 061803. DOI: 10.1103/PhysRevLett.104.061803. arXiv: 1001.4468 [hep-ex].

[15] V. M. Abazov et al. "Observation of Single Top Quark Production." *Phys. Rev. Lett.* 103 (2009), p. 092001. DOI: 10.1103/PhysRevLett.103.092001. arXiv: 0903.0850 [hep-ex].

[16] T. Aaltonen et al. "First Observation of Electroweak Single Top Quark Production." *Phys. Rev. Lett.* 103 (2009), p. 092002. DOI: 10.1103/PhysRevLett.103.092002. arXiv: 0903.0885 [hep-ex].

[17] S. Chatrchyan et al. "Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC." *Phys. Lett.* B716 (2012), pp. 30–61. DOI: 10.1016/j.physletb.2012.08.021. arXiv: 1207.7235 [hep-ex].

[18] S. Chatrchyan et al. "Measurement of the properties of a Higgs boson in the four-lepton final state." *Phys. Rev.* D89.9 (2014), p. 092007. DOI: 10.1103/PhysRevD.89.092007. arXiv: 1312.5353 [hep-ex].

[19] G. Aad et al. "Measurements of Higgs boson production and couplings in the four-lepton channel in pp collisions at center-of-mass energies of 7 and 8 TeV with the ATLAS detector." *Phys. Rev.* D91.1 (2015), p. 012006. DOI: 10.1103/PhysRevD.91.012006. arXiv: 1408.5191 [hep-ex].

[20] V. Khachatryan et al. "Measurement of spin correlations in $t\bar{t}$ production using the matrix element method in the muon+jets final state in pp collisions at $\sqrt{s} = 8$ TeV." *Phys. Lett.* B758 (2016), pp. 321–346. DOI: 10.1016/j.physletb.2016.05.005. arXiv: 1511.06170 [hep-ex].

[21] V. Khachatryan et al. "Search for a Standard Model Higgs Boson Produced in Association with a Top-Quark Pair and Decaying to Bottom Quarks Using a Matrix Element Method." *Eur. Phys. J.* C75.6 (2015), p. 251. DOI: 10.1140/epjc/s10052-015-3454-1. arXiv: 1502.02485 [hep-ex].

[22] G. Aad et al. "Search for the Standard Model Higgs boson produced in association with top quarks and decaying into $b\bar{b}$ in pp collisions at $\sqrt{s} = 8$ TeV with the ATLAS detector." *Eur. Phys. J.* C75.7 (2015), p. 349. DOI: 10.1140/epjc/s10052-015-3543-1. arXiv: 1503.05066 [hep-ex].

[23] G. Aad et al. “Evidence for single top-quark production in the *s*-channel in proton-proton collisions at $\sqrt{s}$ =8 TeV with the ATLAS detector using the Matrix Element Method.” *Phys. Lett.* B756 (2016), pp. 228–246. DOI: 10.1016/j.physletb.2016.03.017. arXiv: 1511.05980 [hep-ex].

[24] J. Alwall, A. Freitas, and O. Mattelaer. “The Matrix Element Method and QCD Radiation.” *Phys. Rev.* D83 (2011), p. 074010. DOI: 10.1103/PhysRevD.83.074010. arXiv: 1010.2263 [hep-ph].

[25] P. Artoisenet et al. “Automation of the matrix element reweighting method.” *JHEP* 12 (2010), p. 068. DOI: 10.1007/JHEP12(2010)068. arXiv: 1007.3300 [hep-ph].

[26] J. Alwall et al. “The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations.” *JHEP* 07 (2014), p. 079. DOI: 10.1007/JHEP07(2014)079. arXiv: 1405.0301 [hep-ph].

[27] G. P. Lepage. “A new algorithm for adaptive multidimensional integration.” *Journal of Computational Physics* 27.2 (1978), pp. 192–203. ISSN: 0021-9991. DOI: http://dx.doi.org/10.1016/0021-9991(78)90004-9. URL: http://www.sciencedirect.com/science/article/pii/0021999178900049.

[28] T. Ohl. “Vegas revisited: Adaptive Monte Carlo integration beyond factorization.” *Comput. Phys. Commun.* 120 (1999), pp. 13–19. DOI: 10.1016/S0010-4655(99)00209-X. arXiv: hep-ph/9806432 [hep-ph].

[29] S. Jadach. “Foam: A general-purpose cellular Monte Carlo event generator.” *Computer Physics Communications* 152.1 (2003), pp. 55–100. ISSN: 0010-4655. DOI: http://dx.doi.org/10.1016/S0010-4655(02)00755-5. URL: http://www.sciencedirect.com/science/article/pii/S0010465502007555.

[30] W. H. Press and G. R. Farrar. “RECURSIVE STRATIFIED SAMPLING FOR MULTIDIMENSIONAL MONTE CARLO INTEGRATION.” *Submitted to: Comp.in Phys.* (1989).

[31] Z. Zhe-Zhao, W. Yao-Nan, and W. Hui. “Numerical integration based on a neural network algorithm.” 8 (Aug. 2006), pp. 42–48.

[32] L. y. Xu and L. j. Li. “The New Numerical Integration Algorithm Based on Neural Network.” *Third International Conference on Natural Computation (ICNC 2007)*. Vol. 1. Aug. 2007, pp. 325–328. DOI: 10.1109/ICNC.2007.730.

[33] L. Yan, J. Di, and K. Wang. “Spline Basis Neural Network Algorithm for Numerical Integration.” *International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering* 7.3 (2013), pp. 458–461. ISSN: PISSN:2010-376X, EISSN:2010-3778. URL: http://waset.org/Publications?p=75.

[34] J. Friedman, T. Hastie, and R. Tibshirani. “Additive logistic regression: a statistical view of boosting (With discussion and a rejoinder by the authors).” *Ann. Statist.* 28.2 (Apr. 2000), pp. 337–407. DOI: 10.1214/aos/1016218223.

[35] J. H. Friedman. “Greedy function approximation: A gradient boosting machine.” *Ann. Statist.* 29.5 (Oct. 2001), pp. 1189–1232. DOI: 10.1214/aos/1013203451.

[36] I. J. Goodfellow et al. “Generative Adversarial Networks.” *ArXiv e-prints* (June 2014). arXiv: 1406.2661 [stat.ML].

[37] J. Bendavid. “Efficient Monte Carlo Integration Using Boosted Decision Trees and Generative Deep Neural Networks” (2017). arXiv: 1707.00028 [hep-ph].

[38] *MeMEMta: Modular Matrix Element Implementation*. URL: https://github.com/MoMEMta.

[39] R. D. Ball et al. “Parton distributions for the LHC Run II.” *JHEP* 04 (2015), p. 040. DOI: 10.1007/JHEP04(2015)040. arXiv: 1410.8849 [hep-ph].

[40] V. Bertone, S. Carrazza, and N. P. Hartland. “APFELgrid: a high performance tool for parton density determinations.” *Comput. Phys. Commun.* 212 (2017), pp. 205–209. DOI: 10.1016/j.cpc.2016.10.006. arXiv: 1605.02070 [hep-ph].

[41] URL: opendata.cern.ch.

[42] A. Hoecker et al. “TMVA: Toolkit for Multivariate Data Analysis.” *PoS* ACAT (2007), p. 040. arXiv: physics/0703039.

[43] M. Feindt. “A Neural Bayesian Estimator for Conditional Probability Densities” (2004). arXiv: physics/0402093.

[44] M. Feindt and U. Kerzel. “The NeuroBayes neural network package.” *Nucl.Instrum.Meth.* A559 (2006), pp. 190–194. DOI: 10.1016/j.nima.2005.11.166.

[45] A. Rogozhnikov. *hep_ml python package*. URL: https://github.com/arogozhnikov/hep_ml.

[46] P. Seyfert. *pseyfert/tmva-branch-adder: Version 0.5.0*. DOI: 10.5281/zenodo.1149690. URL: https://doi.org/10.5281/zenodo.1149690.

[47] *PyROOT*. URL: https://root.cern.ch/pyroot.

[48] N. Dawe et al. *scikit-hep/root_numpy: 4.7.3*. DOI: 10.5281/zenodo.842249. URL: https://doi.org/10.5281/zenodo.842249.

[49] I. Babuschkin et al. *scikit-hep/root_pandas: 0.3.2*. DOI: 10.5281/zenodo.1188928. URL: https://doi.org/10.5281/zenodo.1188928.

[50] J. Pivarski et al. *scikit-hep/uproot: 2.8.17*. DOI: 10.5281/zenodo.1219218. URL: https://doi.org/10.5281/zenodo.1219218.

[51] *c2numpy*. URL: https://github.com/diana-hep/c2numpy.

[52] *root4j*. URL: https://github.com/diana-hep/root4j.

[53] *root2hdf5*. URL: http://www.rootpy.org/commands/root2hdf5.html.

[54] R. Aaij et al. "Search for the lepton flavour violating decay $\tau^- \to \mu^-\mu^+\mu^-$." *JHEP* 02 (2015), p. 121. DOI: 10.1007/JHEP02(2015)121. arXiv: 1409.8548 [hep-ex].

[55] *Third Machine Learning in High Energy Physics Summer School*. 2017. URL: https://indico.cern.ch/event/613571/.

[56] URL: https://vispa.physik.rwth-aachen.de.

[57] L. Bel. "The LHCb Starterkit." *Proceedings of the 38th International Conference on High Energy Physics (ICHEP2016). 3-10 August 2016. Chicago, USA*. 2016, p. 334. URL: http://pos.sissa.it/cgi-bin/reader/conf.cgi?confid=282.

[58] T. Likhomanenko, D. Derkach, and A. Rogozhnikov. "Inclusive Flavour Tagging Algorithm." *J. Phys. Conf. Ser.* 762.1 (2016), p. 012045. DOI: 10.1088/1742-6596/762/1/012045. arXiv: 1705.08707 [hep-ex].

[59] *ALEPH Workshop @ NIPS 2015 – Applying (machine) Learning to Experimental Physics (ALEPH) and «Flavours of Physics» challenge*. Neural Information Processing Systems (NIPS). 2015. URL: http://yandexdataschool.github.io/aleph2015/.

[60] *Heavy Flavour Data Mining Workshop*. Zurich, 2016. URL: https://indico.cern.ch/event/433556/.

[61] *Data Science @ LHC 2015 Workshop*. 2015. URL: https://indico.cern.ch/event/395374/.

[62] *DS@HEP at the Simons Foundation*. 2016. URL: https://indico.hep.caltech.edu/indico/conferenceDisplay.py?confId=102.

[63] *DS@HEP*. 2017. URL: https://indico.fnal.gov/conferenceDisplay.py?confId=13497.