Computer simulations, machine learning and the Laplacean demon: Opacity in the case of high energy physics*

F. J. Boge† and P. Grünke‡

---

*This is a preprint of a paper forthcoming in Resch, Kaminski, and Gehring (Eds.), *The Science and Art of Simulation II*, Springer (expected 2020). The final version may contain various changes.
†Interdisciplinary Centre for Science and Technology Studies (IZWT), Wuppertal University, Wuppertal, Germany
‡Institute for Technology Assessment and Systems Analysis (ITAS), Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

1 Introduction

High energy physics (HEP) is a field of research in which the use of computer simulations (CSs) abounds, a fact that has attracted the attention of a bunch of philosophers (Karaca, 2017; Massimi and Bhimji, 2015; Morrison, 2015). In recent years, machine learning (ML) techniques have become more and more important in HEP as well, e.g. for the discrimination of signal from background data. Even more recently, the HEP community has begun to explicitly acknowledge the "black box"-nature of ML (Chang et al., 2018, p. 1), i.e., the *opacity* associated with it. CS and ML (including supervised techniques, on which we focus on in this paper) share methodological features as well as epistemological challenges. Both can be used for the exact same purposes (recognition and prediction of statistical patterns) and both raise concerns of opacity, uncertainty, and robustness. This makes HEP a natural study case for investigating philosophical concerns of opacity associated with both CSs and ML in modern science.

In this paper, we will pursue the following three general aims: (I) We will define a notion of *fundamental* opacity and ask whether it can be found in HEP, given the involvement of ML and CS. (II) We will identify two kinds of non-fundamental, *contingent* opacity associated with CS and ML in HEP respectively, and ask whether, and if so how, they may be overcome. (III) We will raise the question of whether any kind of opacity, contingent or fundamental, is unique to ML or CS, or whether they stand in continuity to kinds of opacity associated with other scientific research. In short, we will argue that (I) there *is* a fundamental kind of opacity, that (II) contingent kinds *can* be overcome in HEP, to a certain extent, and that (III) all identified kinds of opacity are *continuous* with opacity in non-computer aided science. In (II) we will also establish that even though the two opacities associated with CS and ML are both contingent, the origins and nature of the respective opacities are significantly different. As a consequence, there are differences in *how* to overcome the contingent opacity associated with CS and ML respectively. Notably, in the overcoming of contingent opacities associated with CSs we will focus closely on HEP, as it presents a field of research in which this is done exceptionally well, whereas the opacities to be overcome in ML are a current hot topic for the ML community, not specifically for physics.

1.1 Computer simulations in high energy physics in a nutshell

CSs[1] are used in HEP for various purposes, ranging from the design of equipment or the guidance of searches and analyses to the estimation of expected 'backgrounds' in experiments. For concreteness, let us briefly zoom in a little more detail on the complex simulation infrastructure used by CERN's ATLAS experiment.[2] Here, as in the other experiments at the Large Hadron Collider (LHC), different

[1] We will here not offer a general account of what counts as a CS, since this has been done in various places in the philosophical literature. The reader may be referred e.g. to Humphreys (2004, Chap. 4) instead. Boge (2019) also offers some insights.
[2] E.g. ATLAS (2010). For a detailed presentation in a philosophical context cf. also Boge and Zeitnitz (2020).

stages of a collision event and subsequent happenings[3] will be overall separated into *event generation* and *detector simulation*. Event generators then again factor into *hard process*, the highest-momentum scattering between two constituent 'partons' (quarks or gluons) inside the protons, *parton shower*, the production of a large number of additional partons produced in subsequent decay and radiation events, *hadronization*, the formation of complex particles out of this shower of partons, and the *decay* of hadrons not stable enough to reach the detector. In addition, an *underlying event* may be simulated, meaning a second comparatively high-energetic interaction between two further partons which leads to added activity in parton shower, hadronization, and decay.

The detector simulation will then use all stable particles from the final stages as input and simulate their interaction with the real-world detector, as well as known background fluctuations. This will be done based on a vast range of models coming from atomic, solid state, nuclear, and particle physics respectively, which will sometimes be largely based on theory, sometimes rather phenomenological, and sometimes just collections of parametrized data.[4] The main detector simulation software used at ATLAS (GEANT 4) will also include a model of the geometry, in order to adequately simulate the spatial distribution of these interactions. This will include the possibility of choosing specific physical conditions of the detector, to test the impact of these on one's experiment.

1.2 Deep learning in a nutshell

ML techniques are today involved in many areas of science and social life, with applications ranging from risk management of water distribution systems or the prediction of severe weather conditions to more straightforward recognition and classification tasks (e.g. Baldi et al., 2014; Goodfellow et al., 2016; Rudin and Wagstaff, 2014). Deep learning (DL) is a specific kind of (ML) using *connectionist* (or 'neural') *networks*, in which the network includes multiple hidden layers (the number of which defines its *depth*).

A neural network can be understood as an arrangement of nodes (or 'neurons') which are ordered into different layers. The neurons of one layer will all perform the same task and then hand the result over to one or multiple nodes of the next layer.[5] A standard way to represent these relations is by appeal to directed graphs, wherein the nodes correspond to labeled vertices $\textcircled{\ell}$ and the input-output relation corresponds to directed edges $\uparrow, \nwarrow\!\!\nearrow$, and so forth (cf. also fig. 1). If there is no previous or next layer, the layer in question will be an input or output layer respectively. All other layers are called 'hidden'.

[3] In fact, whole 'bunches' of 100,000 million protons, with a bunch being of the size of 64 microns, will travel in two beams accelerated to 7 TeV inside the LHC. These will cross every 25 ns which results in around 600 million collisions per second (cf. https://lhc-machine-outreach.web.cern.ch/lhc-machine-outreach/collisions.htm; checked 11/18).

[4] To get an impression, one may consult the GEANT physics reference manual (cf. GEANT Collaboration, 2016).

[5] Strictly speaking, this is only an ideal limit; a network may, for instance, have "skip connections" which jump from layer i to layer $i+2$ (Goodfellow et al., 2016, p. 196). We here also eschew the discussion of other kinds of networks, such as recurrent ones, which may not be organized into proper layers at all. Our discussion will be restricted to *feedforward* networks.

Let's consider a simple example.[6] Assume that one's network were supposed to learn the exclusive-or function $\mathsf{XOR} : \{0,1\}^2 \to \{0,1\}$, which gives 1 for the pairs $(0,1)$, $(1,0)$ and 0 else. Such a network could be mathematically represented as taking an input vector $\boldsymbol{x} \in \{0,1\}^2$ and as successively transforming it by applying a sequence of functions. Each entry x_i of $\boldsymbol{x}$ would then correspond to an input node, and the successively executed functions would correspond to hidden- or output layers respectively.

A simple network capable of performing this task can be constructed with only one hidden layer that executes a vector valued function

$$\boldsymbol{h}(\boldsymbol{x}; W, \boldsymbol{c}) = W\boldsymbol{x} + \boldsymbol{c}. \tag{1}$$

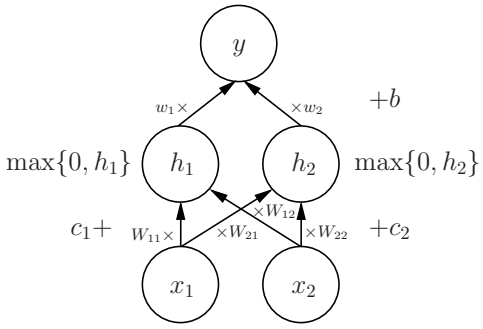

Figure 1: Representation of the network described below by a directed graph. The edges may be interpreted as feeding forward a weighted and biased version of outputs received from a set of 'parent' nodes (vertices). Each 'hidden' vertex may be thought of as computing an activation function on its input. (Reproduced with added detail from Goodfellow et al., 2016, p. 169.)

W here represents a matrix of *weights* that the inputs receive, and $\boldsymbol{c}$ a vector of *biases* that may output some (non-zero) value even if $\boldsymbol{x} = \boldsymbol{0}$. Because of the nature of matrix multiplication, both inputs may contribute to both nodes (h_i) of the hidden layer, depending on the weights w_{ij} contained in W. If the output layer now executes the function

$$f(\boldsymbol{x}; W, \boldsymbol{c}, \boldsymbol{w}, b) = \boldsymbol{w} \cdot \max\{0, \boldsymbol{h}\} + b, \tag{2}$$

where $\boldsymbol{w}$ and b are a vector and scalar with analogous roles as W and $\boldsymbol{c}$, $\cdot$ represents the standard scalar product on $\mathbb{R}^2$, and the max-function is applied component-wise, there is a choice of all the *parameters* (the quantities specified behind the semicola of $\boldsymbol{h}$ and f) such that the network performs the specified task. A function executed by a hidden layer is generally called an *activation function*, and these are often times non-linear. In many cases these are sigmoidal functions, which "ensures there is always a strong gradient whenever the model has the wrong answer." (Goodfellow et al., 2016, p. 177) $\boldsymbol{g}(z) = \max\{0, z\}$ is another "default choice" (ibid., p. 186) called the *rectified linear unit* that allows for several generalizations (cf. ibid., p. 187). All in all the network may be said to execute the composite function $f \circ \boldsymbol{h}$ on $\{0,1\}^2$, which correctly represents XOR for a certain choice of the parameters.

How is this choice of parameters, sometimes collectively denoted by $\boldsymbol{\theta}$, achieved? The standard method is to let the network optimize a *loss* or *cost function* over values of $\boldsymbol{\theta}$, i.e. a function that specifies how far the network's output is from a desired result. The error defined by this loss function is the so called *training error*, which occurs during the stage where the network is confronted with preselected data on

[6] Cf. Goodfellow et al. (2016, p. 166) for the following, which is our main reference on ML/DL here.

which it is tuned for fulfilling the specified task. Neural networks also face the danger of a *generalization error*, which refers to the "expected value of the error on a *new* input." (Goodfellow et al., 2016, p. 107; emph. added) Striking a balance between over- and underfitting, i.e. being too specific or not specific enough to the data the network is trained on, is one major practical task one faces in the design-stage of DL algorithms (ibid., pp. 108 ff.).

Quite usually, optimization is performed by using iterative, *gradient-based* methods, where one 'walks along' the negative gradient in predefined steps.[7] The most important instance certainly is *stochastic gradient descent*, where one updates $\boldsymbol{\theta}$ in small steps $\boldsymbol{\theta} \mapsto \boldsymbol{\theta}' = \boldsymbol{\theta} - \epsilon \boldsymbol{g}$, with

$$\boldsymbol{g} = \frac{1}{m} \nabla_{\boldsymbol{\theta}} \sum_{i=1}^{m} - \ln p(\boldsymbol{y}^{(i)} | \boldsymbol{x}^{(i)}; \boldsymbol{\theta}). \tag{3}$$

Here $p(\boldsymbol{y}^{(i)} | \boldsymbol{x}^{(i)}; \boldsymbol{\theta})$ defines the likelihood of output $\boldsymbol{y}^{(i)}$ given input $\boldsymbol{x}^{(i)}$ (parametrized by $\boldsymbol{\theta}$), ϵ is a predefined step size also called the *learning rate*,[8] and m is either the cardinality of the *training set*, the set of data which the network encounters (and is configured on) before it is put to its actual task, or that of a 'minbatch', a (comparatively small) subset of examples form the (possibly huge) training set. In order to strike a balance between over- and underfitting, it may be useful to add a regularizer $\lambda \Omega(\boldsymbol{\theta})$ to the loss function, with λ a tunable parameter. Stochastic gradient descent effectively means approximating the value of $\boldsymbol{\theta}$ for which the obtaining of $\boldsymbol{x}^{(i)}$ makes the corresponding output $\boldsymbol{y}^{(i)}$ most likely.

The gradient may here be computed with the *backpropagation algorithm*, which "allows the information from the cost to [...] flow backward through the network" (Goodfellow et al., 2016, p. 197), by computing for each node its partial derivatives w.r.t. parent variables and combining them by an appropriate chain rule. This includes observing possible changes of the network's output under changes of the weights. Insofar as the optimization task involved in (stochastic) gradient descent is executed exactly by adjusting these weights (cf. Rumelhart et al., 1986), the algorithm lends a certain autonomy to the network: it can adapt its own infrastructure to perform better at a particular task.

The network discussed above is not really deep but rather 'shallow', as it only has one hidden layer. Deep networks (DNs) used to suffer from a so-called *vanishing gradient problem* in the past, which can be understood roughly as follows: Assume that the network repeatedly multiplies a vector of data by the weight matrix W.[9] In case W has n linearly independent eigenvectors $\boldsymbol{v}^{(j)}$ with eigenvalues λ_j respectively, we can write $W = V \mathrm{diag}(\lambda_j) V^{-1}$, where V is a matrix with the $\boldsymbol{v}^{(j)}$ as columns and inverse V^{-1}, and $\mathrm{diag}(\lambda_j)$ a diagonal matrix with the eigenvalues on its main diagonal. After m repeated

[7] Recall that the gradient ∇f of some function f points in the direction of f's steepest ascent, so moving along $-\nabla f$ will get one in the right direction. Of course one may instead hit a saddle point in this way, so finding (local) minima strictly requires investigating the Hessian as well.

[8] Some algorithms make the steps size a function of prior steps, e.g. by scaling each component by the inverse absolute value at the prior step, so that descent in steep/shallow directions will decrease faster/slower.

[9] This is of course a quite restrictive assumption, but if the weights are similar enough, the problem will arise in similar ways.

applications of W, the effect will be $(V\text{diag}(\lambda_j)V^{-1})^m = V\text{diag}(\lambda_j^m)V^{-1}$, which will vanish quickly in case the λ_j are smaller than 1. When computed by the chain rule, however, the gradient can equally be seen to scale with the weights of previous layers; so the learning may get stuck in early layers of a DN. Today, there is a number of ways to handle this problem, e.g. training the network on simpler tasks first, or even training a simpler network that is made more complex once the simpler one has successfully learned. Cf. Baldi et al. (2014, p. 2) and references therein for further examples.

Moreover, it is not always convenient to specify a *fixed* set or even a fixed number of parameters, whence some models incorporate *non-parametric methods* where no such restriction is imposed. An example is *nearest neighbor regression*, where the network classifies data points $\boldsymbol{x}$ by associating them with entries from the training set that have the minimal distance to $\boldsymbol{x}$ in some chosen norm.

Finally, in the task described above the examples encountered in the training stage are associated with *targets*, i.e., specific outputs $\boldsymbol{y}$ the network is supposed to achieve, when confronted with a respective $\boldsymbol{x}$. Similarly, a network may be required to learn to *predict* a specific target $\boldsymbol{y}$ from an input $\boldsymbol{x}$, usually by learning a conditional distribution $p(\boldsymbol{y}|\boldsymbol{x})$. Algorithms of this kind are sometimes referred to as *supervised*, whereas algorithms that merely gather information about the distribution $p(\boldsymbol{x})$ of a set of data $\boldsymbol{x}$ are called *unsupervised*. Supervised applications include, for instance, classification tasks, regression tasks (essentially: the prediction of a value), or transcription tasks in which a relatively unstructured representation of relevant data is transcribed into textual form. Unsupervised learning tasks include probability density estimation, learning to draw samples from a distribution, or denoising data.

All these will often times rely on a *distributed representation* in neural nets, meaning that different sets of nodes are specialized to processing different features of an object. For instance, in describing different types of vehicles with different colors one set may represent the values of the variable 'color' whereas another may represent the values of the variable 'vehicle type'. Similarly, when representing a group of people, one set may handle person identities whereas another their relationships (cf. Rumelhart et al., 1986, p. 534). The output layer will then indicate the learned relationships, and the network may be said to represent them as distributed patterns of activity in the nodes. Factoring inputs in such a way can have several advantages, particularly when it comes to computational cost and flexibility.

1.3 The rise of deep learning in high energy physics

As can be seen from this brief discussion already, the ML community today has developed a wide range of methods, tuned to specific tasks. Indeed, ML techniques have long been used in *high energy physics* (HEP), but have here received increased interest in recent years, due to the advances in handling vanishing gradients and the associated performance increase of DNs. Uses of ML in HEP include the identification of jet-substructures, which can be used to probe the Standard Model (SM) and increase

sensitivity in finding evidence for physics beyond the SM (Larkoski et al., 2017), uncertainty estimation and combination, or the determination of parton density and fragmentation functions (Carrazza, 2018). Maybe the most important HEP application is, however, in the (ongoing) search for *new* particles (cf. Baldi et al., 2014, p. 2). In such a search, physicists need to "isolate a subspace of their high-dimensional data in which the hypothesis of a new particle or force gives a significantly different prediction than the null hypothesis, allowing for an effective statistical test." (ibid., p. 1)

The crucial quantity for significance evaluation here is the *likelihood ratio*, i.e., the ratio $p(D|\Theta)/p(D|\Theta_0)$, where D represents the data, Θ the hypothesis to be tested, and Θ_0 a null hypothesis. In the case of searches for exotic particles in LHC's proton/proton scattering, these take the generic form:

Θ: $pp \rightarrow$ 'known debris' + (intermediate) exotic particle(s)

Θ_0: $pp \rightarrow$ 'known debris'

For the purposes of experiment, both Θ and Θ_0 have to be translated into suitably connected hypotheses about a *test statistic*, such as total mass or (missing) transverse momentum (cf. James, 2006, p. 255, and below). The two steps indicated in Θ and Θ_0, moreover, may themselves be intermediate steps in a larger chain of decays that involves only known (and measurable) 'debris' as final state products. These are typically *jets*, meaning conical distributions of hadronic particles, *leptons* such as electrons or muons, together with missing transverse momentum from corresponding (anti-)*neutrinos*, and *photons*, all of which will be registered by appropriate parts of the detector in use.

Quite often, however, the relevant likelihoods "cannot be expressed analytically, so simulated collision data generated with Monte Carlo methods are used as a basis for approximation of the likelihood function." (Baldi et al., 2014, p. 1) Still, the "high dimensionality of data[...] makes it intractable to generate enough simulated collisions to describe the relative likelihood in the full feature space[...]." (ibid.)

Statistical hypothesis testing may be seen as kind of classification task: if a certain set of parameters exceeds certain bounds, we call our observations 'significant' and think of them as possibly indicating the presence of some phenomenon of interest. In the same way, an ML classifier may take "a complicated, high-dimensional input $\boldsymbol{x}$ and summarize it with a single category identity y." (Goodfellow et al., 2016, p. 233) Hence physicists in HEP use ML classifiers as a means of 'dimensionality reduction': a set of criteria is used to define a hypersurface in the feature space beyond which data will count as signal, and below (or on) which they will count as 'background'. This hypersurface then defines a cut-value for a suitable test statistic in hypothesis testing, which equally defines the cut-off between the two classes of the classifier.

In learning to classify correctly, the network may be said to effectively learn to *represent* the corresponding likelihood functions. This should be compared to the toy-example discussed in section 1.2, where the network may equally be said to represent the XOR function in terms of the functions executed

by the individual layers, when the parameters take on appropriate values. One should caution against reading too much into this notion of 'representation', however, since it does not involve any implication of *intentionality* that one might intuitively associate with the term. We will return to this issue below.

So ML classifiers in HEP searches can learn to distinguish 'signal' from 'background' data. But what exactly does that mean? By 'background', one means any kind of data present in the relevant region in which distributions of quantities of interest are measured, but which is not connected to the process of interest. For instance, this could be data with identical (measurable) final state products, but where the distributions of kinematical features will be different from those in the signal data, because of the occurrence of exotic particles as intermediate states in the latter case and their absence in the former. Baldi et al. (2014), in fact, considered two benchmark cases of exactly this kind in order to evaluate the performance of a DN with five hidden layers as a classifier against that of a shallow one with only one hidden layer.

One of these benchmark cases, which we shall discuss in a little more detail, concerned processes that would involve new theoretical Higgs bosons. As is well known, a 'light' Higgs boson, compatible with the predictions of the SM of particle physics, has been found in 2012 by ATLAS and CMS (cf. Aad et al., 2012; Chatrchyan et al., 2012). However, it is a theoretical possibility that there exists a more massive electrically neutral Higgs boson, H^0, which decays into the known 'light' Higgs boson, h^0, via (positively or negatively) electrically-charged Higgs bosons, $H^{\pm}$. More precisely, a set of possible decay chains would be

$$gg \rightarrow H^0 \rightarrow W^{\mp}H^{\pm} \rightarrow W^{\mp}W^{\pm}h^0 \rightarrow W^{\mp}W^{\pm}b\bar{b}, \tag{Θ}$$

where g are gluons contained in the scattering protons, $W^{\pm}$ are electrically charged intermediate vector bosons, and $b, \bar{b}$ are (anti-)quarks known to produce characteristically broad jets (e.g. Llorente and Cantero, 2014). This was to be contrasted with background processes of the form

$$gg \rightarrow g \rightarrow t\bar{t} \rightarrow W^{\mp}W^{\pm}b\bar{b}, \tag{Θ_0}$$

with t the weak isospin-partner of b, in which the heavier Higgs's would not be present, but where the final-state products would be the same. Since neither Ws nor bs are directly detectable, one would have to make a selection on the decay modes of these. The selection in place was on events with one lepton (ℓ) and missing energy from the corresponding neutrino (ν) from the decay of one of the Ws, two jets from the decay of the other one (jj), and two further ones from the decay of the bs (bb). These were required to satisfy the following characteristics: (i) Exactly one electron or muon, with transverse momentum $p_T > 20$ GeV and pseudorapidity satisfying $|\eta| < 2.5$;[10] (ii) at least four jets, each with $p_T > 20$ GeV and

[10] $p_T = \sqrt{p_x^2 + p_y^2}$, where $(p_x, p_y, p_z)^t$ is the particle's three-momentum in a laboratory frame and p_z lies in the beam direction. The pseudorapidity η is defined as $-\ln\tan(\theta/2)$, where θ is the polar angle of a right handed coordinate system

$|\eta| < 2.5$; and (iii) jets that could be 'tagged' as resulting from bs rather than gluons or lighter quarks. Transverse momenta, the reconstructed missing transverse momentum for the neutrino, as well as the 'b-tagging' information were regarded as 'low-level' features, since transverse momenta are more or less directly measured by the detector.

Process (Θ) could, however, also be associated with rather different hypotheses regarding *higher-level* features than process (Θ_0), meaning features that are not measured or directly inferred from measured quantities (like missing momentum) but rather defined as (typically nonlinear) functions thereof. An example are reconstructed invariant masses, where the invariant mass of particle 1, decaying into (detectable) particles 2 and 3, for instance, is reconstructed in natural units as $m_1^2 = m_{23}^2 = (E_2+E_3)^2 - |\boldsymbol{p}_2+\boldsymbol{p}_3|^2$, due to the relativistic energy-momentum relation and conservation laws. According to both, (Θ) and (Θ_0), there should be peaks in the $m_{\ell\nu}$ and m_{jj} distributions at the known mass m_W. However, (Θ_0) would predict peaks in $m_{j\ell\nu}$ and m_{jbb} distributions at the known value for m_t, whereas (Θ) would predict peaks at known m_h^0 in m_{bb} as well as at $m_{H^\pm}$ in m_{Wbb} and at m_{H^0} in m_{WWbb} respectively, for assumed masses $m_{H^\pm} = 325$ GeV and $m_{H^0} = 425$ GeV of $H^\pm$ and H^0 respectively.

Lower level features usually have a limited *discrimination power*: Often the p_T-distributions corresponding to signal and background hypotheses differ only slightly, so classification according to these features is difficult. Shallower networks are hence usually trained on higher-level features, and performance can be further increased if both high- and low-level features are available to them.

Baldi et al. (2014) now compared the performance of shallow and deep networks, where low-level features were used as inputs for the DNs and higher-level ones for the shallow networks. To assess the advantage of adding hidden layers, both kinds of networks were trained with the same number of hidden units and the same *hyper-parameters*, i.e., parameters of the network to be determined in an optimization procedure outside the training for the actual task. Different classifiers were trained on low-, high-, or combined features respectively, in order to assess which training-strategy would yield the best performance. They each encountered 2.6 million training examples from Monte Carlo data, and the training was validated on 100,000 further ones. The performance was then tested on 500,000 simulated events (ibid, p. 5).

In addition, performance was compared to *boosted decision trees*, where decision trees are "tree-structured classifiers that consist of a series of binary splits", effectively dividing "the multi-dimensional observable space into many (rectangular) volumes that are attributed to either signal or background." (Voss, 2013, p. 178) Boosting these means consecutively defining additional trees by increasing the weights on misclassified events and obtaining the final classification as a weighted average of outputs from all these trees. This is done to compensate individual trees' sensitivity to statistical fluctuations in

with origin at the interaction point.

training data, and has lead to "dramatic performance increases." (ibid., p. 179)

DNs fed with low-level inputs now significantly outperformed both boosted decision trees and shallow networks fed with the full feature set on two significance metrics: the discovery significance and the area under the receiver operating characteristic curve (AUC), where the former is the well known measure for the probability of false positives (high discovery significance = low probability), and the latter a joint measure for the probability of correct positives and negatives under changes of a cut-value on the test statistic (large area under the curve meaning a large range of cut-values for which both quantities are high; e.g. Voss, 2013, for details). DNs trained with the full feature set performed even slightly better, but those trained only on the high-level features surprisingly performed worse. The increase in performance upon adding the high-level feature was always less than the difference between the performance of DNs trained with low-level data and that of shallow ones or BDTs trained on high-level data.

The upshot is that while boosted decision trees and shallow networks are already very useful tools when fed with adequately predefined variables, handing over the task of discovering physical connections between measured (or immediately inferred) quantities to a DN will lead to an even better performance. In other words: increased success in using ML techniques comes at the price of giving up more (immediate) insight into the relations between physical variables.

Despite this remarkable usefulness of ML techniques for HEP, physicists have recognized that they essentially

> function as a *black box*: send in some data and out comes a number. While this kind of nonparametric estimation can be extremely useful, a physicist often wants to understand what aspect of the input data yields the discriminating power, in order to learn/confirm the underlying physics or to account for their systematics. (Chang et al., 2018, p. 1; emph. added)

A different way of putting things is that ML is *epistemically opaque*: some amount of insight into what is going on is not being made available to any user. To delineate the *nature* of this opacity in more detail however, and to what extent or in which sense it may or may not be overcome, we will now first turn to the philosophical debate, which has mostly focused on *computer simulations* (CSs). We will then define the unique sense in which ML or DL is opaque, beyond the sense already present in CSs, and subsequently discuss the (in)eliminability of both kinds of opacity.

2 Opacity in computer simulation and deep learning

What could it mean that ML, and particularly DL, is effectively a 'black box', i.e., epistemically opaque? Intuitively, we might refer to any kind of procedure as epistemically opaque if we cannot fully comprehend

how a certain input leads to a certain output. Indeed, several authors in the philosophical literature (e.g. Humphreys, 2004, 2011; Kaminski, 2018; Lenhard, 2011; Saam, 2017) have claimed that this precisely applies to CSs.

From a certain vantage point, however, this seems counterintuitive: the structures involved in CSs are all *algorithmic*, and hence fully determined by the program code. For every step n it is predefined how to get to step $n + 1$, and all the rules are written down and can be reviewed by any agent who has access to the code. Following Lenhard (2011, p. 137), we may refer to this as the *algorithmic transparency* of CSs. The very same may be said, e.g., about a neural network: the execution of functions and optimization of parameters will be implemented as an algorithm, so it is transparent in the same sense. How could CSs and ML create results in an epistemically opaque way in spite of their algorithmic transparency?

A crucial first step to seeing this is to realize that opacity is intuitively relative to a (set of) cognitive agent(s): *we* do not understand the input-output relation in a certain procedure. This is reflected also in the classic definition by Humphreys (2011, p. 139):

> a process is epistemically opaque relative to a cognitive agent X at time t just in case X does not know at t all of the epistemically relevant elements of the process. A process is essentially epistemically opaque to X if and only if it is impossible, given the nature of X, for X to know all of the epistemically relevant elements of the process.

Using this definition, it is clear that many complex CSs come out as *prima facie* epistemically opaque. In CSs used by CERN's ATLAS experiment at the LHC, for instance, no human being will presumably ever be able to review all the steps of the computer program by herself, as $O(10^6)$ lines of code, written in different programming languages, can be relevant for the full simulation of events and detection (cf. ATLAS, 2010, p. 835).[11]

Saam (2017, p. 79), moreover, points out that "while the program code (with several tens of thousands of lines of code) may be epistemically transparent, the compiled code is not (several million lines of code are possible)", where by 'compiled code' one means the translation of the program code into machine code, i.e., into a set of instructions the computer can actually execute (e.g. Clements, 2006, p. 205). Hence compilation may increase the amount of code by two orders of magnitude, so that the code instructing the machines used for CS at ATLAS may require up to $O(10^8)$ lines.

However, what about the notorious Laplacean demon, who would not be subject to human restrictions? For him, studying even complied code would not present an obstacle, and so he would be capable of grasping the content of even the most complex program. Invoking the demon in this way, we can see that even essential opacity is closely tied to the epistemic conditions of an actual agent, and thus, in a

[11] This code is managed by the Athena framework, which allows users to specify properties of the intended simulation through the JobOptions service, but also to some extent in the form of additional Python scripting (cf. ATLAS Computing Group, 2005, Sect. 3.3).

specific sense, not a *principled* inaccessibility. As a first step towards clarification, we may therefore distinguish between a *contingent* kind of opacity, which depends on the actual epistemic conditions of a given cognitive agent X or a community thereof, and a *principled* or *fundamental* one which would apply to any agent X, regardless of their cognitive facilities.

Is not obvious that indisputable instances of principled opacity exist at all. However, in Bohmian versions of quantum theory, for instance, the precise values of quantities such as the velocity of an elementary particle cannot be found out by any conceivable experiment, despite the fact that they exist according to the theory (cf. Dürr et al., 2012, p. 139). Hence there are at least *candidate* cases of principled opacity.

To capture the relevant differences more clearly, let us try and make things more precise by formalizing the relevant notions. Humphreys' definition of opacity O of process p for agent[12] X at time t, with X having nature $N = n$ at t and some piece of information i_p about p being epistemically relevant, R, to X but not known, K, by X at t, is *relational* and triadic:

$$O(p, X, t) \leftrightarrow \exists i_p (N(X, t) = n \wedge \neg K(X, i_p, t) \wedge R(i_p, X, t)) \tag{4}$$

The agent's nature doesn't do any real work here and could be omitted, but it seems nice to keep it in place, because it will do all the work in the subsequent definitions.

In the definition of *essential* opacity, Humphreys doesn't appeal to time anymore, so it is merely dyadic. Moreover, Humphreys assumes the impossibility for X of acquiring everything that is epistemically relevant. Since impossibility implies non-occurrence already in the second weakest modal system however (cf. Hughes and Cresswell, 1996, pp. 41–2), we take it that one may replace talk of impossibility by talk of factive unavailability at all times without buying into any strong modal commitments. A nice (metaphorical) way of putting it is hence that in essential opacity, O_e, time is 'marginalized for', qua being bounded by a universal quantifier:

$$O_e(p, X) \leftrightarrow \forall t \exists i_p (N(X, t) = n \rightarrow \neg K(X, i_p, t) \wedge R(i_p, X, t)) \tag{5}$$

Our definition of *fundamental* opacity, O_f, now goes a step further in 'marginalizing' for both X and t:

$$O_f(p) \leftrightarrow \forall X \forall t \exists i_p (N(X, t) = n \rightarrow \neg K(X, i_p, t) \wedge R(i_p, X, t)) \tag{6}$$

In words: given any agent with any nature, at no time will the agent be able to obtain all relevant pieces of information about p. Only 'supernatural' beings could possibly overcome fundamental opacities, for

[12] Humphreys intends his definition to include human as well as non-human agents. For the purposes of our paper we will only consider human agents.

if one was to overcome one's own nature, one would still end up having some different agent's nature, and hence still miss some piece of information i_p.

Notably, we have chosen to represent an agent's nature by a dynamical variable here, and this may certainly raise some controversy. For should an agent's very nature not be fixed once and for all? What could a sensible alternative formalization that respects this intuition look like? The work done by the mention of an agent X's nature here is to point to certain facilities and restrictions that X faces. The intuition says that these restrictions/facilities should be somewhat strongly tied to X itself. An obvious alternative would hence be to formalize X's nature by letting X's biological and cognitive *state*, $S(X,t)$, range exclusively over a set $\bar{N}$ such that $\bar{N} := \{n | n \text{ is nomologically/metaphysically accessible to } X\}$:

$$O_e(p,X) \leftrightarrow \forall t \exists i_p (S(X,t) \in \bar{N} \rightarrow \neg K(X,i_p,t) \wedge R(i_p,X,t)) \tag{7}$$

However, accepting this definition of X's nature would render the impossibility of knowing i_p somewhat trivial: Given the fact that quantification over time is supposed to track impossibility, the definition in (7) basically says that obtaining i_p is impossible for X if it is impossible for X. Moreover, locutions like 'it was unnatural for me but I did it anyways' suggest that one's nature does *not* connect too intimately to an *impossibility* of sorts. Neither does the fact that transhumanism is, for all we know, both a nomological and metaphysical possibility, and it would be an obvious way to overcome (and thereby change) one's nature. Hence we find definition (5) preferable, and the intuitions motivating (7) somewhat misguided.

Given this level of precision, what is the significance of our earlier appeal to the Laplacean demon? The Laplacean demon, to recall, is "an intelligence sufficiently vast [so that] for it, nothing would be uncertain and the future, as the past would be present to its eyes." (de Laplace, 1814, p. 4) Gillies (2000, p. 14), moreover, points out that the Laplacean demon, at least in Laplace's sense,[13] is just an extrapolation of a human agent's capabilities. Indeed, we find Laplace state that: "The human mind offers, in the perfection which it has been able to give to astronomy, a feeble idea of this intelligence." (de Laplace, 1814, p. 4)

Of course, Laplace was concerned only with deterministic theories, containing rather 'tame', solvable DEQs. Today, this is not feasible anymore for the vast majority of physical theories. However, we are here interested in CSs and ML, methods that can be defined in terms of *algorithms*. Once this level of algorithm has been reached (say by translating the approximate content of a physical model into computer code for simulation), it seems that Laplace's demon would have an easy time figuring out the algorithm's workings.

Algorithmic transparency is hence enough to render even essential opacity *contingent*, qua contingent

[13]There are various precisifications of the notion of a Laplacean demon in the literature: Frigg et al. (2014, pp. 33-4), for instance, define the demon to have access to initial conditions, possess a *perfect* model, and be able to compute *arbitrarily* fast. For our purposes, a more modest demon suffices that could come to know the boundary conditions of a given algorithm, compute *vastly faster* than any human being, and provide a highly detailed *error analysis* regarding hardware and software induced artifacts or the like. This will cover insight into the details of any realistic, humanly written computer program.

on an agent's nature. The Laplacean demon, who would be capable of biting through every step of the algorithm, may serve as an ideal *point of reference* for a 'natural' being here, and only if a method was opaque to him as well could it count as *fundamentally* opaque in our sense.

2.1 Computer simulations: Opacity induced by complexity

So CSs and ML come out as contingently opaque (qua algorithmically transparent) on our account. However, are the contingencies that lead to them the same? Indeed, there seems to be a major difference, which relates to the *kind of code*, i.e., to its purpose and associated definitions.

To see the difference, let's briefly recall the key features of the ATLAS simulation infrastructure: An 'event generator', factored into hard scatter, parton shower, hadronization, decays, and possibly an underlying event, feeds simulated 'final state particles' into a detector simulation, which then models the interaction of these particles with elements of the ATLAS detector, as well as its geometry.

However, as we noted above, detector simulations will rely on a vast range of different models, coming from different parts of physics. In some cases, such a 'model' will be highly phenomenological, i.e., not based on fundamental physical principles, or even simply amount to a function with free parameters, fitted to measured data. Reasons are that detailed information of the microphysics is unavailable because of the material composition and the complexity of interactions, or even missing information about the exact composition and purity of the material and the distribution of type impurities therein.

Moreover, while the structure of the hard scatter is basically dictated by perturbative quantum chromodynamics (QCD), one has to make certain assumptions, such as the transferability of *factorization theorems* to various unproven cases, which imply the factorization of proton/proton cross sections into calculable, elementary ones and a remainder called patron distribution functions. This factorization assumption in turn implies the need for a choice of an arbitrary factorization scale, where the choice has less and less influence with the inclusion of higher terms in the perturbation series.

Similarly, one can use either the separation angle between two partons or their transverse momenta as an evolution variable in parton showers, and different choices have traditionally proven fruitful in the two standard general-purpose generators PYTHIA and Herwig. There is also the need to cut the evolution off at some point, as relevant quantities otherwise diverge and cannot be interpreted in terms of the (probabilistic) Markov evolution that is the basis of standard shower-algorithms. In hadronization, finally, the energies are too low to use perturbative QCD at all, and one has to resort to one out of a range of phenomenological models, with a number of additional free parameters, to bridge the gap.

In midst this jungle of choices of modeling assumptions, parameters, conditions etc., who can really tell what a given simulation set up at ATLAS is precisely going to do? What is the effect of choosing certain parameters? Certain models/parametrizations in detector simulations? Certain (versions of) event

generators? Combinations between them? On top of all that, documentation is usually sloppy, and quite a large number of people will have contributed individual pieces to this huge amount of code; so it is not straightforward to read up on the details and impossible to ask a (small group of) developer(s) about their code.

The source of this apparent opacity seems to reside in the *complexity* of the CSs, as well as of the underlying problem set: It is the difficulty to determine many details such as the best choices of models or parameters exactly in advance, as well potential uncertainties arising from the interplay of the various models and choices. It seem that ATLAS is just a paradigmatic example in this respect though: in many detailed CSs in science, the interplay of a vast number of components and the complexity of the problem set will make it very hard to determine the precise goings on during the simulation, if not impossible for non-Laplacean demons.

2.2 Deep learning: Opacity induced by the method

The nature of the opacity associated with ML and DL has so far scarcely been addressed in the philosophical literature, a fact which is presently changing, as reflected by the contributions to this volume.

A brief footnote-remark can be found e.g. in Schembera (2017, p. 60):

> machine learning could change the whole picture. Nevertheless, in terms of the algorithms, there is still strong transparency: All the steps of a computer program are determined, but the opaqueness of the paths of the steps as well as of the results, is increased so that they become not traceable.

This remark underscores our earlier observation of ML's algorithmic transparency: The functions implemented in DNs or other learning algorithms are laid down beforehand, and so ML is in the same sense algorithmically transparent as are CSs. Nevertheless, ML may be *additionally* accompanied by a *special* kind of opacity, which Schembera locates in the "opaqueness of the paths".

Humphreys (2004, p. 149) similarly hints at a unique kind of opacity associated with DNs:

> Epistemic opacity [...] plays a role in arguments that connectionist models of cognitive processes cannot provide an explanation of why or how those processes achieve their goals. The problem arises from the lack of an explicit algorithm linking the initial inputs with the final outputs[...] at a level that correctly captures the structural aspects of the surface representation[...], together with the inscrutability of the hidden units that are initially trained.

Humphreys here of course has in mind Fodor and Pylyshyn's (1988) influential paper on connectionism and classical cognitive architecture (cf. his Fn. 22), and interesting issues arise in this connection indeed.[14]

[14] We will return to these only in section 3.3, when scrutinizing the use of 'representation' in the context of DL, and only to a limited extent.

However, Humphreys blames the lack of an explicit algorithm at a *humanly understandable* or 'conceptual' level (cf. ibid.), whereas ML is algorithmically fully transparent at the level of programming, as we have seen.

Another way to put the problem is that there is no algorithm that renders the *flow of information* through the net transparent to the epistemic agent. Again, complexity is at least in part to blame for the occurrence of opacity in this sense: the DNs used by Baldi et al. (2014), for instance, included five hidden layers with 300 units each. Assuming that each node in one layer was connected with nonzero weight to each node of the successor layer, this makes $O(10^{12})$ paths along which information can travel between hidden layers. This makes it hardly conceivable that a simple algorithm can be found that allows any human agent to track the flow of information through the net.

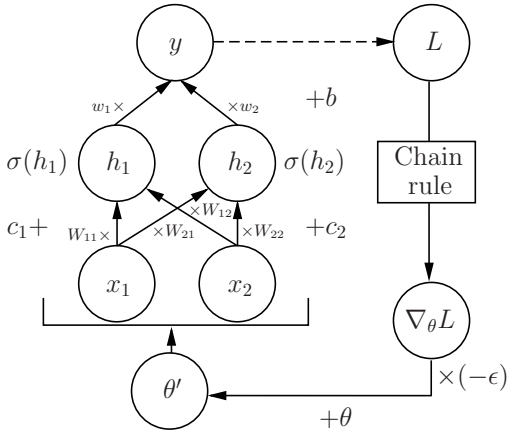

Figure 2: Pseudo-computational graph for stochastic gradient descent in a shallow network with sigmoid activations. (The backpropagation algorithm is not resolved in detail.)

However, complexity *alone* does not seem to be the culprit here: In stochastic gradient descent, for instance, we equip a DN with backpropagation and parameter-updating algorithms and then let it 'learn' itself, based on these algorithms. Hence we convey a partial- or *quasi-autonomy* on the system, by letting it perform itself the iterative tuning of parameters that ultimately leads to success in the given task. This is different in CSs: here the scientist will himself intervene over and over, to tune a simulation program better to a specific task.[15] If this kind of control is given up, it becomes opaque in which ways the network was able to achieve the desired success. It is hence a kind of opacity *induced by the method* that is specific to 'learning' algorithms.

For concreteness, consider the simple, shallow network depicted in Fig. 2, similar to the one from Sect. 1.2. In the network depicted here, the hidden units execute sigmoid functions, which makes their partial derivatives non-trivial. Fig. 2 also displays an additional loop from the output back to the input, representing backpropagation together with stochastic gradient descent, which establishes the development of the network during the training phase. Assuming that the training is supervised, a loss function (denoted L in fig. 2) will be computed which compares the output to a target output t. For simplicity, assume also that the likelihood is Gaußian with variance 1 around t and the output binary. Then the loss function in formula (3) would, up to the regularizer term, yield $L = \frac{1}{2}(y - t)^2$, where y is a function of b, the $w_i, \sigma(h_i), c_i$, and W_{ij}. Derivatives w.r.t. all parameters in $\boldsymbol{\theta}$ will now have to

[15] It may be remarked here that the same usually still applies to a network's hyperparameters (cf. above).

be computed successively, which, by the chain rule, implies the need to compute derviatives such as $\frac{\partial L}{\partial y} = y - t$ or $\frac{\partial \sigma(h_i)}{\partial h_i} = \sigma(h_i)(1 - \sigma(h_i))$ and chain them up appropriately to compute the derivatives further down.

The goal of the training phase, to recall, is to minimize L, thereby effectively searching for a y s.t. $y \approx t$. The stochastic gradient descent will now use the entirety of derivatives computed by backpropagation, plug them into a gradient ∇L,[16] and update the set of parameters $\boldsymbol{\theta}$ by subtracting $\epsilon \nabla L$ from $\boldsymbol{\theta}$. This process will be iterated through a number of 'epochs' until a predefined proximity between y and t is reached. All this will be done without any intervention by the researcher, and therefore quasi-autonomously.

Despite the fact that more hidden layers can speed up the training, 10000s of epochs may be needed, producing a huge amount of individual steps that would need to be reviewed to track exactly how the final network is reached. Note also that we have here not only restricted our presentation to a shallow network, but also to supervised learning. In the case of unsupervised learning, where no target t is specified, it will likely be even more opaque how a network reaches its final configuration.[17]

Supervised or not, it seems impossible to retrodict, from the final configuration, the steps taken by the network, and since individual steps are executed without direct guidance by the researcher, it is also impossible to simply check corresponding records. This justifies our claim that this is another kind of opacity, induced by the quasi-autonomy of the machine during the training phase and the invisibility of these stages in the final configuration, i.e. by the *method* itself. While complexity obviously plays a role, opacity is not purely due to complexity here.

However, this opacity induced by the method is still of a contingent nature: a Laplacean demon could compute all steps of the learning algorithm himself and, assuming knowledge of the inputs and initial weights and biases, thereby inspect all steps undergone by the network during the training epochs.

3 Overcoming opacity

Above, we have clarified three 'levels' of opacity ('ordinary', essential, and fundamental opacity) through careful analysis and formalization. Moreover, we used Laplace's (original) demon as a an ideal point of reference for a 'natural' agent, which allowed us to group opacity and essential opacity into a class we coined 'contingent opacity'. Certainly, clarity is desirable, but what exactly is *interesting* about this analysis?

The fact is that the continuity between the demon and human agents suggests that humans could overcome contingent opacities *to a certain degree*. Without having a formal notion of degrees of opacity

[16] Since derivatives will be taken also w.r.t. matrix elements, one would, strictly speaking, first have to 'flatten everything out' into a column vector to use the procedure exactly as described here. This is just a matter of representation though.

[17] There are already applications of unsupervised learning in HEP. An example for the use of unsupervised learning in discrimination tasks is discussed, e.g., in Andreassen et al. (2018).

in mind, we shall now make it plausible that this could be achieved for ML methods in the future, and is in fact achieved to a remarkable extent for CSs in HEP.

The first step in this endeavor is to isolate the *epistemically relevant* elements that figured crucially in all three definitions. In this connection, Durán (2018, p. 108; emph. added) has recently pointed out that "researchers are only interested in a limited *amount* of information that counts for the *justification* of results." A detailed knowledge of the program code is presumably not relevant for justifying the results of a CS. Rather, validating well-designed code against known data and available benchmarks may justify some amount of trust in its results, insofar as success in these procedures is transferable. Hence justification of the *desired* knowledge, typically knowledge about some real-world target of the simulation, may convey relevance on some piece of information i_p about the simulation process p.[18]

Another hint for delineating epistemic relevance comes from Lenhard (2006), who relates epistemic opacity directly to *understanding*. Several epistemologists (e.g. Brendel, 2013; Kvanvig, 2003) have indeed pointed out that we usually value knowledge not for itself but for its ability to increase understanding. Riggs (2003) and Pritchard (2014), moreover, connect understanding especially to *scientific* knowledge, which is certainly apt for most if not all CS-laden sciences.

Lenhard (2006, pp. 611–3), however, suggests to distinguish between an *insight view* and a *pragmatic view* of understanding. The insight view basically says that one understands a 'device', such as an equation, if one can determine properties of its solution, i.e., determine what it *does*, without necessarily being able to solve it / reproduce its behavior (cf. Feynman et al., 1965, Vol. 2, p. 2-1). The pragmatic view rather says that one understands the device if one is able to use it in order to exert control over phenomena.

While Lenhard bites the bullet and argues that CSs are accompanied by a loss of insight-like understanding regarding their own detailed functioning while promoting control over certain phenomena, thereby increasing our pragmatic understanding of these, insight-like understanding seems clearly required to render a CS (more) transparent. Hence, if we want to argue that CSs in HEP are relatively transparent, we also need to show how physicists gain insight into their workings, not just how they can use them successfully.

Note that the Feynman-approach to insight does *not* require the ability to *solve* the equation / *walk through* the algorithm for claiming insight: It is exactly the ability to determine what either does *without* having to solve / go through it that is crucial for being able to claim insight into it.[19] In a similar spirit,

[18] It may be remarked here that Plato famously already highlighted this systematicity induced by justification, i.e. the ability to 'stabilize' knowledge by other knowledge through justification, as a central value of knowledge over mere true belief in the *Menon* (97A–98A).

[19] By the same token, the Laplacean demon need not immediately gain insight by executing the algorithm by hand (say) (cf. similarly Kaminski et al., 2018, p. 260, for the difference between execution and understanding). However, it is usually far easier to understand what e.g. a DEQ says or means when one already has the solution in hand and can study its properties as well. Similarly, the ability to scrutinize each and every step of an algorithm should equip the demon with an excellent basis for understanding its workings, given his immense computational power.

Durán (2018, p. 108) has recently pointed out that a detailed understanding of each and every line is certainly not required to understand what some code *does*. Rather, an assessment of the way in which it *functions* as well as of the inter-relations between its different parts is in order for such a purpose.

In sum, we take it that any piece of information i_p about process p that connects some 'input' $\boldsymbol{x}$ with an 'output' $\boldsymbol{y}$, where an epistemic agent X is interested in $\boldsymbol{y}$, will only be epistemically relevant to X if i_p either (a) conveys justification on accepting the output as serving an intended epistemic purpose (such as adequately representing the states of a physical system) or (b) serves to convey an insight into the way in which p leads from $\boldsymbol{x}$ to $\boldsymbol{y}$. If all this information was available, p would be perfectly transparent. As we have shown, however, the right amount and level of information to seek for justification or insight respectively does not necessarily reside in the details of the code in a CS or ML technique.

Given these preliminaries on epistemic relevance, how could agents 'overcome their natures' to render essentially opaque processes more transparent? The most obvious answer would be transhumanism, i.e. the integration, say, of suitable equipment into one's brain, or the 'upload' of the mind into a supercomputer. But these are presently just science fiction fantasies, so we'll leave it at that.

More realistically, agents may *team up*, divide the labor, pack the results of individual efforts into a more accessible form, and then achieve transparency collectively. By this we do not mean that the collective as a *whole* achieves transparency, but rather, that through collaboration each *individual* from a skilled group with a good division of labor may achieve a great deal of transparency.

Alternatively, an agent could exploit auxiliary *devices* or techniques whose functioning is sufficiently transparent to her. It is a case-by-case question whether these techniques work – which Humphreys (2009, p. 619), thinks, isn't fulfilled for most CSs. But below we will make a case that this (especially the first option) is realistic at least for CSs utilized by the ATLAS collaboration.

In contrast to this assessment, Kaminski et al. (2018) have recently identified several sources of opacity they consider relevant for CSs, namely *social*, *technological*, and *mathematical* ones (cf. also Durán, 2018, pp. 106–7, for a recap). The social sources reside in the fact that one usually builds on the works of others in science; in a CS, e.g., on libraries or modules that one has not programmed oneself, whereby one typically lacks insight into them. The technological ones connect to the deeper understanding of one's research equipment that one usually lacks as well, which in a CS is simply the computer hardware. Among mathematical reasons, they further distinguish between internal and an external ones, meaning the opacity associated with the problem on a conceptual level on the one hand and the opacity associated with the code implemented in a CS to solve a given mathematical problem on the other.

For Kaminski et al. (2018), the possibility of collective or technology-aided transparency we have sketched above would be doubt worthy, and in a sense we agree: auxiliary devices and other agents may introduce another layer of sources of opacity. However, depending on the communication skills and

technological abilities of individual agents, the opaque remainder may become close to negligible. And as we have claimed above, this may not even be so far from the truth in actual examples.

3.1 Complexity: Learning about model relations

Let's now first consider the overcoming of merely complexity-induced opacity, as it attaches to the simulation infrastructure at ATLAS. Indeed, we already admitted that documentation may often be sloppy, which can make it hard to comprehend. Nevertheless, a tremendous *amount* of documentation exists: The ~250 000 lines of code in the Athena framework specific to ATLAS simulations, for instance, are accompanied by some ~70 000 lines of commentary (ATLAS, 2010), as well as various user manuals and wikis providing overall details on the CSs' functioning to the user (e.g. https://twiki.cern.ch/twiki/bin/view/Main/IUedaAtlasSimulation#Atlas_Simulation or ATLAS Computing Group 2005, for examples). It seems that getting insight into the detailed workings of pieces of code is largely a matter of *personal effort*, not one of principled restrictions.

However, insight also requires knowledge of the interactions between parts, and we had also identified the justification of results as a second key condition for epistemic relevance. Indeed, the Athena code will be constantly validated against benchmarks and tested for integration, i.e. for joint functioning of different components on small and large scales, given individual functioning (cf. ATLAS Computing Group, 2005, p. 78 ff.). Three frameworks are used for this, called AtNight, Kit Validation, and Run Time Tester (RTT). There are differences in how these are used: RTT, for instance, "is currently run daily on a farm at UCL", and the "full set of RTT results currently take a number of hours to generate." (ibid., p. 79) AtNight, on the other hand, is used to test the 'nightly builds', i.e., automatically generated parts of the total code that are still at an early stage of development. Developers have a number of choices in these frameworks as to what to test and how (cf. ibid.).

At least for CSs used at ATLAS, it seems safe to say that "acceptability" is not *only* judged, as Lenhard (2006, p. 613; orig. emph.) has it, "according to the model's *overall* behavior," but also by the components' behaviors and by their interrelations. In other words: pace Lenhard (2006) and Humphreys (2004, p. 148), high energy physicists *do* obtain a fair amount of the "traditional transparency" connected to an "understanding[...] based upon the ability to decompose the process between model inputs and outputs into modular steps, each of which is methodologically acceptable both individually and in combination with the others."

The possibility for this transparency depends in part also on the fact that the code has been written *purposefully* by physicists, and that each part of it will derive from some sort of physics model (cf. Boge and Zeitnitz, 2020). The relations between these models can be understood in detail before programming, and dependencies between model parameters will be known and respected by job specifications allowed

by Athena, or even already compensated for in the code.

An example of such a compensation is matrix element matching: In the simulation of an event and subsequent processes, "double counting can arise when the same final state can be generated both by the parton level calculation, and by the showering algorithm." (Mangano and Stelzer, 2005, p. 564) The details are intricate (cf. ibid., pp. 579–80) and shall not be recaptured here. In essence, however, "a probabilistic rejection procedure [is] applied to events falling in the overlap" between phase space sectors associated with parton showers and hard scatters respectively, "to ensure that a given phase-space configuration is only counted once." (ibid., p. 579) Hence already on the level of physics modeling (with the goal of simulation in mind), insight will be gained by studying relations between different models and modeling steps in detail. Moreover, several pure programming artifacts are known and compensated for as well: an example are 'stuck tracks' in the detector simulation, due to an infinitely dense spacing of the simulated components.

In sum: Basing epistemic relevance in justification and insight-like understanding renders many details of the code for ATLAS CSs epistemically irrelevant, so that LHC physicists may be said to achieve a fair deal of transparency by means of careful modeling, programming, documenting, and testing.[20] It is clear that some aspects, such as certain sources of error, will remain opaque nonetheless. However, in the case of understanding and securing *success*, we could only claim a high degree of opacity for these CSs by imposing rather low standards on epistemic relevance.

3.2 Method: Learning about the learning process

Now what about the acclaimed 'black-box' nature of ML techniques? Complaints of this kind may be connected to some general facts about ML, in particular that certain DNs have been known to suffer from *robustness issues*, in the sense that learned strategies will fail under even minute (though often specific) differences in inputs (cf. Marcus, 2018, p. 8). This first and foremost indicates difficulties in the *justification* of ML *results*, however, and may be compensated for by empirically testing for robustness in specific tasks (which is generally claimed to be rather high in HEP; cf. Guest et al. 2018), determining, as far as possible, the source of the non-robustnes otherwise, and improving on algorithms for individual tasks.

Indeed, there is the so called "no free lunch theorem", which "states that, averaged over all possible data-generating distributions, every classification algorithm has the same error rate when classifying previously unobserved points." (Goodfellow et al., 2016, p. 111) The implication is

[20] One may wonder why this is possible in HEP but maybe not so much in fields like climate science. Two reasons offer themselves: (a) In HEP one relies on proper experiments, which implies a high degree of control over the objects of study (e.g. Radder, 2009; Schurz, 2014), whereas in climate science one is largely limited to field observations. And (b), there is some solid theoretical core in HEP (the Standard Model), which can be used to guide the development of CSs, whereas in climate science there is a plethora of models that start from rather diverse assumptions (e.g. Parker, 2018, Sect. 4.1).

> that we must design our machine learning algorithms to perform well on a specific task. We do so by building a set of preferences into the learning algorithm. When these preferences are aligned with the learning problems that we ask the algorithm to solve, it performs better. (ibid.)

These preferences will be implemented in terms of regularizers $\lambda\Omega(\boldsymbol{\theta})$[21] added to the loss function. An example is the 'weight decay' regularizer $\lambda \boldsymbol{w}^2$, where $\lambda > 0$ will force the weights to become smaller, as $\nabla \boldsymbol{w}^2 = 2\boldsymbol{w}$, which will approximate zero only in case $\boldsymbol{w} \to 0$.

We have argued, however, that the network's ability to self-correct introduces an opacity different in kind from the merely complexity-based one. If opacity is hence connected to this partial autonomy of the network, it may be difficult to determine the source of non-robustness.

Moreover, empirical steps for securing that some DN or boosted decision tree achieves the desired goal, i.e., justifying its results, would do no good for overcoming opacity in the sense of conveying *insight* into learning processes. The case is asymmetric in that we implement a certain (physics) *model* in a CS, and by performing benchmarks and integration testing we get an insight into whether this was done *as intended*. This is different in ML: Here the algorithm is basically just a method for extracting information from data in a largely model-independent fashion,[22] and for the reasons named above, the insight we may desire is exactly as to *how* this information was extracted.

How do ML scientists cope for this kind of opacity? Indeed, protocols here are not as far developed as is the case with CSs at ATLAS, and the quest for transparency, usually subsumed under the header 'explainable artificial intelligence' (XAI) is more of an open field of research (e.g. Russell et al., 2017, p. 94). Yet some remarkable approaches and results already exist that sketch a path to XAI and allow to render specific ML techniques more and more transparent (e.g. Ronen, 2017, for an overview).

To give a concrete example, let us briefly zoom in on the approach by Schwartz-Ziv and Tishby (2017), which used information-theoretic quantities to track the evolution of a DN with stochastic gradient descent. A central insight underlying this approach was that a DN without skip connections (cf. Sect 1) can be seen as a *Markov chain*: each layer corresponds to a random vector whose distribution will depend only on its direct predecessor. Using the so called *information plane* (cf. Schwartz-Ziv and Tishby, 2017, p. 5), the authors were then able to track the evolution of DNs with 1–7 hidden layers of decreasing size (2–12 nodes) towards the output. The information plane is defined by considering the distributions $p(\boldsymbol{h}^{(n)}|\boldsymbol{x})$ and $p(\boldsymbol{y}|\boldsymbol{h}^{(n)})$ for hidden layer $\boldsymbol{h}^{(n)}$ and using the *mutual information* $I(\boldsymbol{h}^{(n)};\boldsymbol{x})$ and $I(\boldsymbol{y};\boldsymbol{h}^{(n)})$

[21] Thomas von Clarmann (private comm., 2019) has pointed out to us that understanding the exact functioning of regularizers often poses additional challenges. This implies that rendering algorithms that include regularizers transparent may require additional effort. It does not impair (but rather underscores) our general point though: that there are conceivable techniques for rendering ML largely transparent, in individual cases.

[22] Cf. also Humphreys (2013) on this point.

respectively as x and y coordinates of a Cartesian coordinate system. Here,

$$I(\boldsymbol{v};\boldsymbol{w}) := \sum_{i,j} p(v_i, w_j) \log_2 \left(\frac{p(v_i, w_j)}{p(v_i)p(w_j)} \right), \tag{8}$$

which provides the average number of bits of (relevant) information $\boldsymbol{v}$ contains about $\boldsymbol{w}$ and vice versa.

Now Schwartz-Ziv and Tishby (2017) performed a bunch of numerical experiments in which for configurations of 12 uniformly distributed points on a sphere a yes/no decision had to be made, yielding $2^{12} = 4096$ possible input configurations. To generate a joint distribution $p(\boldsymbol{x}, y)$, a *logistic sigmoid*,

$$\sigma(f(\boldsymbol{x}) - \theta) = (1 + \exp[-\gamma(f(\boldsymbol{x}) - \theta)])^{-1}, \tag{9}$$

of some spherically symmetric function $f(\boldsymbol{x})$ of the pattern (uniquely indicating the pattern itself) was computed and interpreted as the probability $p(y = 1|\boldsymbol{x})$. This can be understood by realizing that $\sigma(f(\boldsymbol{x}) - \theta)$ approaches the Heaviside distribution $\Theta(f(\boldsymbol{x}) - \theta)$ for growing γ, which corresponds to a deterministic yes/no decision in which $f(\boldsymbol{x})$ is compared to a criterion θ ($f(\boldsymbol{x}) \geq \theta$ means yes, everything else no). The sigmoid hence created a probabilistic rule for a yes/no decision conditional on $\boldsymbol{x}$, which would yield a joint distribution $p(\boldsymbol{x}, y) = p(y = 1|\boldsymbol{x})p(\boldsymbol{x})$, with $p(\boldsymbol{x})$ chosen uniform as indicated above. θ was chosen such that $p(y = 1) = \sum_{\boldsymbol{x}} p(y = 1|\boldsymbol{x})p(\boldsymbol{x}) \approx 0.5$ and γ large enough such that the mutual information $I(\boldsymbol{x}; y)$ was approximately 0.99 bits (the decision close to deterministic).

The network's nodes executed an arctan activation function, which was binned into 30 equal intervals between -1 and 1. This binning was used to compute an approximation for the joint distributions $p(\boldsymbol{h}^{(n)}, \boldsymbol{x})$, from which all other quantities, in particular the information plane-coordinates, could be calculated (cf. Schwartz-Ziv and Tishby, 2017, p. 8). This was repeated for 50 randomly chosen initializations of the network (weights and biases), as well as configurations $\boldsymbol{x}$ (ibid.).

One of the central insights from these experiments was the existence of two distinct phases in stochastic gradient descent:

> In the first and shorter phase the layers increase the information on the labels (fitting), while in the second and much longer phase the layer reduce the information on the input (compression phase). Schwartz-Ziv and Tishby (2017, p. 3)

This is to be understood in the sense that the layers first moved up along the $I(y; \boldsymbol{h}^{(n)})$-axis, thereby increasing the amount of information each layer would provide for y, and then move to towards zero (towards the left) on the $I(\boldsymbol{h}^{(n)}; \boldsymbol{x})$ axis, thereby *reducing* the information each layer contained about $\boldsymbol{x}$. This may be seen exactly as a fit-then-optimize learning strategy: The networks first learned to achieve the goal (produce the desired y from given $\boldsymbol{x}$) and then removes all the information about the input not required to achieve that goal with the same accuracy.

Moreover, by minimizing the quantity $I(\boldsymbol{h}^{(n)};\boldsymbol{x})-\beta I(\boldsymbol{y};\boldsymbol{h}^{(n)})$ independently over all $p(y_i|h_i^{(n)})$, $p(h_i^{(n)})$ and $p(h_i^{(n)}|x_i)$, one finds a set of equations, parametrized by β, that define a curve through the information plane on which $\boldsymbol{h}^{(n)}$ may be said to be an efficient representation of $\boldsymbol{x}$.[23] Ultimately, the networks approximated points on that curve.

A second result of the study, which could be explained with the help of the foregoing one (cf. Schwartz-Ziv and Tishby, 2017, p. 12), is that more layers will speed up the training phase. This may be understood on the basis of the optimization (removal of unnecessary detail) performed by the earlier layers, which provides already suitably compressed information to the later layers and makes their convergence to a desired result easier.

This is an impressive insight into stochastic gradient descent in a specific feedforward network, but of course not a general result about all DL, let alone ML. However, mutual information is invariant under invertible transformations of the variables; so the general approach can be carried over to a whole range of DNs. Moreover, the task was repeated with non-symmetric distribution over the sphere, and no difference in the general results was found. This also suggests a carry-over to a range of similar classification tasks. Finally, this is just one exemplary study in a broader research field, as we have pointed out above. Other studies may shed light on other kinds of learning machines or provide complementary insights on stochastic gradient descent.

Obviously, the procedures explored here are rather different from what is being done to overcome the complexity-induced opacity of involved CSs at ATLAS. This underscores our earlier claim that the opacity in question, and accordingly the information required for achieving transparency, is different in kind. However, on a high level of abstraction, quite similar reasons are relevant to the establishment of transparency: a learning algorithm is *purposefully* designed, implying already a fair deal of knowledge about its constitution. Testing corresponding DNs on a broad range of applications can then delimit the scope of the algorithm's suitability and compensate for lacking a priori knowledge during the design stage. This will covey *justification* on the acceptance of a range of results, to the extent that the tasks resemble each other in relevant respects. Moreover, using methods like that of Schwartz-Ziv and Tishby (2017), one can obtain a fair deal of *understanding* of the process the machine undergoes during the training stage. All in all, this may compensate for a considerable amount of opacity induced by the machine's relative autonomy. When all this is done correctly, the missing epistemically *relevant* information may become almost negligible here as well. But at present, XAI appears to be still pretty much in its infancy, and there is hence a de facto asymmetry to CSs at ATLAS.

[23] This is possible only if the output layer is not a deterministic function of the input layer. However, any deterministic function can be made stochastic by assuming that there is always some noise, and by correspondingly adding some minor stochastic error. This can be implemented exactly in terms of (probabilistically interpreted) sigmoid functions with their width representing the amount of noise (cf. Schwartz-Ziv and Tishby, 2017, p. 6 and below).

3.3 Model-opacity: What do we really learn?

We have thus sketched an exemplary way in which the learning process of a machine may be rendered more transparent, and the opacity induced by the method may be overcome to a considerable degree. However, physicists such as Chang et al. (2018), from which we took the 'black box'-quote, are interested rather in the *physical features* on the basis of which a given DN learns to classify, not exactly in the way in which it does so.

There are, in other words, two *dimensions* to the 'black box'-ness of ML and DL methods, namely *how* the machine learns and *what* it learns. The approach by Schwartz-Ziv and Tishby (2017) discussed above targets only the first dimension. Chang et al. (2018), in contrast, tried to understand what physical features would yield the discriminating power for a DN. To that end, they used what they called a *planing scheme*, which essentially means removing information regarding certain variables in a data set and observing consecutive changes in the DN's discrimination power. The 'planing' refers, more precisely, to the weighting of events by a quantity inversely proportional to the differential cross section $\mathrm{d}\sigma\,(\boldsymbol{x}_i)/\mathrm{d}m\,|_{m=m_i}$, or a relevant distribution more generally, where $\boldsymbol{x}_i$ labels an input vector for event i, and m_i means the value m takes on in that event.

In practice, Chang et al. (2018) inverted histograms, meaning that events would be weighted by $1/(\#\text{counts in bin } i)$ for bins of the histogram corresponding to variable m, making this histogram itself a uniform distribution. Performance drops were then measured in terms of a reduced AUC, for an extension of the SM in which a new vector boson Z' would be introduced that decays into a e^+e^- pair and couples to SM particles. Two cases were considered: Z'_V where the couplings to right-handed[24] and left-handed SM particles would be identical, and Z'_L where the boson would couple exclusively to left-handed SM particles. Data in both cases were produced using a Monte Carlo simulation, with 10^6 proton-proton collisions. The relevant background comprised standard pair creation events $\gamma^* \rightarrow e^+e^-$, with γ^* a(n off-shell) photon. The mass for both Z's was assumed to be $M_{Z'} = 1$ TeV and the characteristic expected lifetime such that the corresponding peak at 1 TeV in the mass distribution would have characteristic width $\Gamma_{Z'} = 10$ GeV. The photon background, in contrast, would simply be decreasing.

Two variables were then planed for in order to investigate performance drops: the reconstructed invariant mass m of the final-state products, as in the benchmark case discussed above, and the rapidity $y = \frac{1}{2}\ln[(E+p_z)/(E-p_z)]$, or rather the difference $\Delta y \equiv |y(e^-)| - |y(e^+)|$, which provides information about the angular distribution and hence indirectly about the asymmetry between couplings to right and left-handed SM particles. Actually, in the case Z'_V, only m was planed for, because of the (symmetric)

[24] Recall that the Dirac equation is solved by a spinor-field with four entries. For massless particles, the upper part then corresponds to negative eigenstates of a helicity operator ('right handed ones'), meaning that their spin lies in the opposite direction as their momentum, whereas for the lower part the two are aligned ('left handed' ones; e.g. Griffiths, 2008, pp. 338 ff., for an overview). The names are common for massive particles as well, although the identification with helicity eigenstates is not possible here.

way the example was set up. The AUC then approached 0.5 upon planing for m in this case, which means[25] that "no noticeable discriminating power remains." (Chang et al., 2018, p. 4) This was also compared to the performance of a linear network (no hidden layer), which would, as expected, perform significantly worse than the DN before planing when only fed with four-momentum information, but almost as good when equipped with joint information on p and m.

In the Z'_L case, on the other hand, some discriminating power was left in both networks when only m was planed (AUC>0.6) and a high correlation (0.90) could be established in the linear responses (before activation with a sigmoid function by the hidden units) with the variable Δy. Planing for Δy as well, the performance of both networks then approached 0.5 again.

Bottom line: planing selectively for certain variables, one gets an indirect insight into the physical characteristics (angular anisotropy of the distribution, therefore unequal couplings to left- and right-handed particles; high information content in mass-distribution, therefore massive new particle) underlying the data. But the model was already known here, and the variables to be planed for were hence suggested on the basis of that model. So even if such a "procedure can be explored systematically", i.e., by planing iteratively for all conceivable physically relevant variables, it is "most efficient in tandem with physics intuition." (Chang et al., 2018, p. 5)

We must stress, at this point, that *one does not retrieve a model from the procedure, and a lot of variation in detail remains possible.* Let us refer to this fact as *model-opacity*. It is the opacity of the underlying physics, which could be captured by a mathematical model and is responsible for the machine's ability to discriminate, i.e., leads to the phenomena made accessible via the machine. It seems that there is no algorithm which leads us from the machine's output to a concise understanding of the underlying physical mechanisms. So even a Laplacean demon might be lost, as, in contrast to the kinds of opacity discussed above, we seem to have come across a variant that is *not* algorithmically transparent.

The problem set associated with this kind of opacity is acknowledged in the ML community typically under the header of 'causality':

> what makes one representation better than another? One answer[...] is that an ideal representation is one that disentangles the underlying causal factors of variation that generated the data, especially those factors that are relevant to our applications. (Goodfellow et al., 2016, p. 544)

In fact, supervised learning and a *distributed representation* may be utilized exactly with the aim of providing the machine with a means for learning the relevant factors of variation (ibid., pp. 544–5). However, we indicated above that one should be cautious about the meaning of 'representation'

[25] The AUC is standardly normalized to unity, so an area of 0.5 means that the best one can get is a joint 50% chance for correct positives and negatives. This in turn essentially says that the classifier is doing guess work instead of systematically classifying events.

here. Consider, for instance, the following observation by van Fraassen (2008, p. 7, emph. added): "A representation is made with a *purpose* or *goal in mind*, governed by criteria of adequacy pertaining to that goal, which guide its means, medium, and selectivity." Similarly Suárez (2003, p. 237; emph. added): "*A* [...] cannot stand in [in a representing relation to *B*] unless it is *intended as* a representation of *B* by some suitably competent and informed *inquirer*." There is, in other words, a good case that representation requires intentionality, which in turn may require conscious content such as agreed-upon standards. In what sense, then, may a network be said to represent a given function, such as a likelihood ratio? We take it that it can represent such a function *to us*, in the sense that it does everything that the likelihood ratio does: it spits out a number of 'events' in the right frequency relative to expected 'background events'.

Now given that a likelihood is of the form $p(\text{data}|\text{hypothesis})$, there is a problem associated with the nature of the input here. For when we train a network on the basis of Monte Carlo data, resulting from the implementation of some approximation to a physical model, we know exactly what the 'hypothesis' is on which we condition and which gives us the input to be classified: it is a part or consequence of the implemented model. When we let a well-trained network loose, in contrast, on a range of previously unexplored data (say in a new energy range achieved with some future collider) and it indicates the presence of a completely unexpected phenomenon to us, then the network really just represents some sort of distribution, not a well defined likelihood; or more colorfully: a likelihood conditioned on an *unknown* hypothesis.

In sum, talk of representation is misleading here in two ways: (a) the network itself will not *literally* represent a given function, in the sense in which a conscious scientific agent would if she had a mathematical representation of it. And (b), even if we can track what the network does and thereby work out the function 'represented' by the network, this would not lead us to a representation of the *underlying reality*, i.e. a model from which the function derives.

Now is model-opacity, as we have called it, fundamental or not? We cannot draw a final verdict on this issue, but we may speculate that it is. Above, we always linked the possibility to overcome the other kinds of opacity to their algorithmic transparency, for instance, when benchmarking is used to check whether the purposefully written algorithm performs as desired, or when information theoretic methods are being employed to track a network's evolution. The apparent lack of algorithmic transparency in model-opacity, however, might leave even a Laplacean demon lost in as yet uncharted domains of data. What methods such as the planing performed by Chang et al. (2018) establish is to narrow the space of possible hypotheses down, at least within a conceptual scheme such as quantum field theory, or maybe present day particle physics more specifically. But that still leaves a lot of room, and it is unclear whether all possible hypotheses can be explored in a systematical – let alone algorithmic – fashion. Hence it is equally unclear whether "there are ways to circumvent *any* form of epistemic opacity", as suggested by

Durán (2018, p. 107, emph. added).

4 Discussion: A novel phenomenon on any level?

Let us now consider whether any of the opacities discussed here is unique to CS or ML. For the complexity based one, it seems that the identification of social or technological sources and their say in the opacity of complex code or its implementation renders this sort of opacity a common scientific phenomenon, not a particulate feature of CSs: In any complex research task, one has to rely on the standards and expertise of others: specialists will perform and optimize pre-defined sub-tasks, such as calibration of equipment, and unless explicitly requested, one will not get insight into the execution of these tasks, regardless of an involvement of CSs . One usually also does not have deeper knowledge of the equipment one uses: Measurements performed by a producer can convey some amount of insight, but it also requires some effort to understand the details of a manufacturer's specifications or even test their adequacy.

Kaminski et al. (2018), however, suggest that the specific opacity associated with CSs resides in the fact that it conveys an opacity in the solution to internally opaque mathematical methods, such as intractable DEQs. Now complex mathematical problems will quite often urge the move to approximation techniques and even educated guesses or assumptions. This is in fact a quite prominent feature of many areas of theoretical physics: An already mentioned example (cf. Sect. 2.1) is the transfer of factorization schemes for different contributions in proton/proton scattering that is only proven for a small range of scatterings (cf. also Boge and Zeitnitz, 2020, and references therein).

These considerations so far only demonstrate that there are in fact many other methods than using CSs to overcome sources of opacity internal to the mathematics of modern science. For Kaminski et al. (2018, pp. 268 ff.), however, what is special about CSs is that they convey external justification on the acceptance of certain results despite the fact that the path to this justification is (essentially) opaque. This may be understood as the justification-analog to Lenhard's approach to the understanding-aspect of epistemic relevance: just as CSs may convey pragmatic understanding of certain phenomena while reducing insight into the means by which this understanding is achieved can they externally justify the acceptance of certain results (to within specifiable margins of error) while rendering the reasons for the obtaining of a particular result opaque.

However, many exact proofs of complex mathematical theorems will be executed by extraordinary mathematicians, and proofs may sometimes remain intransparent, possibly even essentially, for other individuals in the scientific community. A nice example is the so called *Feynman-Dyson split* (Kaiser, 2005, pp. 175 ff.): Feynman had introduced his famous diagrams in an intuitive, heuristic fashion, without providing any rigorous mathematical reasons for their functioning on the basis of quantum field theory. This defect was compensated for by Freeman Dyson, but in private communications with his Friend Ted

Welton, Feynman freely admits to have never thoroughly understood those proofs (cf. ibid., p. 178).

As the example demonstrates, opaque external justification of mathematical results can occur as a social phenomenon in non CS-aided science. Thus the specialty identified by Kaminski et al. (2018, p. 271), that in using CSs for finding (approximate) solutions to mathematical equations we need to appeal to experience while usually mathematics is regarded as an entirely a priori business is not so special after all. As Kripke (1972, p. 159; emph. added) famously put it: "one can learn a mathematical truth a posteriori by consulting a computing machine, *or even by asking a mathematician*."

The same reasoning does not, however, apply to ML techniques: The results here are adaptations of a given machine to a given task, or, in unsupervised learning, to existing structures in the training data. However, the fact that one utilizes, in the quest for XAI, additional methods external to one's understanding of the underlying algorithms can shed some light on connections to opacities that arise in techonolgy-aided but ML-free research.

Consider, for instance, efforts made for understanding learning machnies to those of the ATLAS collaboration for understanding the composition of parts of their detector, e.g. the so called 'insertible *B*-Layer', "a new innermost pixel layer,[...] installed during the shutdown period in 2014" (ATLAS Collaboration, 2017, p. 1). The details of the corresponding study are not terribly important here; in short, data taken in 2015, after insertion, was compared to Monte Carlo simulations in order to reconstruct photon conversion and hadronic vertices in the insertible *B*-Layer, i.e. points of decay of hadrons and photons in virtue of their interaction with the material. Because photon conversion can be reconstructed quite exactly and hadronic decays can be resolved over small angles, these could jointly be utilized to obtain "a detailed radiography of the material, including minute components, e.g. the capacitors mounted on the surfaces of the pixel modules, allowing their location to be determined precisely." (ATLAS Collaboration, 2017, p. 7) This knowledge resulted "in a reduction in the uncertainties associated with the charged-particle reconstruction efficiency determined from simulation." (ibid., p. 1)

The crucial point is this: while learning machines may be interesting objects of study themselves, in HEP they merely serve as a particular kind of instrument. Hence, just as an increased understanding of the insertible *B*-Layer of the ATLAS detector leads to an increased ability to estimate its use in reconstructing trajectories, will an increased understanding of the processes that lead from inputs to outputs in, say, a neural net increase high energy physicists' ability to estimate its uncertainties, scope of applicability, robustness, or more generally speaking its suitability as a scientific instrument. Opacity induced by the method should, in other words, be seen as continuous with other *technology-induced* opacities in scientific research: while the details obviously differ, it is not a special feature of ML-based analyses that the precise functioning of one's instruments will be opaque in many respects.

What, finally, about model-opacity? Is it, at last, unique to ML? The sobering answer is that even

model-opacity is not. For compare the finding of a certain signature by some machine to the discovery of the Balmer series in atomic physics: Balmer (1885) discovered a certain pattern in spectral lines that he described systematically by a certain parametrization, but he did not provide any explanation of it. It was not until the advent of the early quantum theory that at least some sort of explanation became available (e.g. Mehra and Rechenberg, 2000, for the historcial details), and that required the *ingenuity* of extraordinary physicists such as Niels Bohr, and later Heisenberg and Schrödinger, not merely their computational capacities. Who knows what mathematical and conceptual interventions future physics may urge of us, and whether the right kind of genius is on the horizon?

Thus, model-opacity is a fundamental problem of scientific inquiry that occurs in explorative phases, usually when an accepted paradigm fails to provide appropriate guidance w.r.t. a significant range of newly discovered phenomena. Moreover, its presumed non-contingency is basically an invocation of the *problem of unconceived alternatives*:

> The unparalleled breadth, predictive power, precision or other substantive epistemic virtues of some present theories go no distance whatsoever toward showing that we have somehow acquired the ability to exhaust the space of scientifically serious alternatives: indeed, it is in part because these very theories, with the unique advantages over their predecessors they really do enjoy, were themselves previously part of the space of unconceived alternatives that we seem to have every reason to believe that serious, well-confirmed alternatives to our own theories remain presently unconceived.[26] (Stanford, 2006, pp. 44–5)

Hence, *none* of these opacities is unique to CSs or ML: All modern research practices are opaque for contingent reasons such as complex mathematics, involved social structure, involved technology, and so forth. Model-opacity, on the other hand, is closely related to, if not an invocation of, the problem of unconceived alternatives, a long-standing problem in the philosophy of science.

5 Conclusions

In this paper, we have shown that ML is associated with an opacity different in kind from that accompanying CSs, and we have coined them opacity induced by complexity (CSs) and opacity induced by the method (ML) respectively. We have then argued that both kinds are contingent on the epistemic conditions of cognitive agents, and that opacity induced by complexity can be overcome by suitable documentation and testing, as done by the ATLAS collaboration at CERN, and that opacity induced by the method may

[26] Indeed, Stanford (2006, p. 45) holds it "possible to imagine cognitive supercreatures who are adept at conceiving of all possible theoretical explanations for a given set of phenomena (or at least all those that they and/or their successors might regard as scientifically serious)[...]." This would render the problem of unconceived alternatives, as well as model-opacity, contingent. So long as we see no evidence of a systematic method such suprecreatures could employ, it remains equally possible, however, that no such creatures could exist and model-opacity is indeed fundamental.

be overcome in a number of ways, e.g. by tracking the evolution of information in stochastic gradient descent.

We have also distinguished these opacities from a *fundamental* one which would pertain to any agent, and suggested that ML (or DL more specifically) is accompanied by another sort of opacity that may be of this kind: model-opacity. This is so because methods such as data-planing can only limit the range of suitable models within a certain theoretical paradigm, but will not positively suggest a given model.

Finally, we have argued that none of the identified opacities are unique to CSs or ML methods: the sources of opacity present in the complexity-induced opacity of CSs all occur in a socially or technologically induced form also in non-CS aided science; and the method-induced opacity associated with ML is really a special kind of technology-induced opacity. Model-opacity, on the other hand, was identified as being basically an invocation of the problem of unconceived alternatives, and thus intimately connected with a long-standing epistemological problem in all of science.

There is, however, a way in which the involvement of ML techniques renders the occurrence of model-opacity special after all. For unlike with human scientist, we cannot converse with a machine or ask for the physical intuitions that guided its discovery. If ML techniques continue on their successful path, this may change the face of science in such a way that discoveries become abundant but are not driven by theoretical results and hypothesis testing anymore. In this way, the scientific process itself may become more opaque in the future.

Acknowledgments The research for this paper was funded by the German Research Foundation (DFG) and conducted as part of the research unit *The Epistemology of the Large Hadron Collider* (grant FOR 2063). We have also profited from comments by audiences at the *SAS Workshop 2018 – Epistemic opacity in computer simulation and machine learning* in Stuttgart, Germany, as well as by members of the of the *PHILETAS* group at ITAS (KIT, Karlsruhe) and the Research seminar at the Institute for Philosophy, in particular Gregor Betz, Thomas von Clarmann, and Michael Poznic. We owe special thanks to Paul Humphreys for extended commentary and discussion on the manuscript.

References

Aad, G. et al. (2012). Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC. *Physics Letters B*, 716(1):1–29.

Andreassen, A., Feige, I., Frye, C., and Schwartz, M. D. (2018). Junipr: a framework for unsupervised machine learning in particle physics. *arXiv*, 1804.09720 [hep-ph].

ATLAS (2010). The ATLAS simulation infrastructure. *The European Physical Journal C*, 70(3):823–874.

ATLAS Collaboration (2017). Study of the material of the atlas inner detector for run 2 of the lhc. *Journal of Instrumentation*, 12(12):P12009.

ATLAS Computing Group (2005). Computing technical design report. Technical report, CERN. Online: https://cds.cern.ch/record/837738/files/lhcc-2005-022.pdf (checked 11/18).

Baldi, P., Sadowski, P., and Whiteson, D. (2014). Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5:4308.

Balmer, J. J. (1885). Notiz über die Spectrallinien des Wasserstoffs. *Annalen der Physik*, 261(5):80–87.

Boge, F. J. (2019). Why computer simulations are not inferences, and in what sense they are experiments. *European Journal for Philosophy of Science*, 9(1):13. https://doi.org/10.1007/s13194-018-0239-z.

Boge, F. J. and Zeitnitz, C. (2020). Polycratic hierarchies and networks: What simulation-modeling at the LHC can teach us about the epistemology of simulation. *Synthese*. https://doi.org/10.1007/s11229-020-02667-3.

Brendel, E. (2013). *Wissen*. Berlin: De Gruyter.

Carrazza, S. (2018). Machine learning challenges in theoretical hep. *arXiv:1711.10840v2 [hep-ph]*.

Chang, S., Cohen, T., and Ostdiek, B. (2018). What is the machine learning? *Physical Review D*, 97(5):6 pp.

Chatrchyan, S. et al. (2012). Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC. *Physics Letters B*, 716(1):30–61.

Clements, A. (2006). *Principles of Computer Hardware*. Oxford, New York: Oxford University Press, fourth edition.

de Laplace, P. S. (1902 [1814]). *A Philosophical Essay on Probabilities*. London: Chapman & Hall, Ltd. Translated from the 6th french edition by Frederick Wilson Truscott and Frederick Lincoln Emory.

Durán, J. M. (2018). *Computer Simulations in Science and Engineering: Concepts—Practices—Perspectives*. Cham: Springer Nature.

Dürr, D., Goldstein, S., and Zanghì, N. (2012). *Quantum Physics Without Quantum Philosophy*. Berlin, Heidelberg: Springer.

Feynman, R. P., Leighton, R. B., and Sands, M. (1965). *The Feynman Lectures on Physics*. Reading, MA: Addison-Wesley.

Fodor, J. A. and Pylyshyn, Z. W. (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1):3–71.

Frigg, R., Bradley, S., Du, H., and Smith, L. A. (2014). Laplace's demon and the adventures of his apprentices. *Philosophy of Science*, 81(1):31–59.

GEANT Collaboration (2016). Physics reference manual. *GEANT 4*, Release 10.4.

Gillies, D. (2000). *Philosophical Theories of Probability*. London, New York: Routledge.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. Cambridge (MA), London: The MIT Press.

Griffiths, D. (2008). *Introduction to Elementary Particles*. Wiley, second edition.

Guest, D., Cranmer, K., and Whiteson, D. (2018). Deep learning and its application to lhc physics. *Annual Review of Nuclear and Particle Science*, 68:161–181.

Hughes, G. E. and Cresswell, M. J. (1996). *A New Introduction to Modal Logic*. London, New York: Routledge.

Humphreys, P. (2004). *Extending Ourselves: Computational Science, Empiricism, and Scientific Method*. Oxford University Press.

Humphreys, P. (2009). The philosophical novelty of computer simulation methods. *Synthese*, 169(3):615–626.

Humphreys, P. (2011). Computational science and its effects. In Carrier, M. and Nordmann, A., editors, *Science in the Context of Application*, pages 131–142. Dordrecht, Heidelberg: Springer.

Humphreys, P. (2013). Data analysis: Models or techniques? *Foundations of Science*, 18:579–581.

James, F. (2006). *Statistical Methods in Experimental Physics*. World Scientific, 2nd edition.

Kaiser, D. (2005). *Drawing Theories Apart*. The University of Chicago Press, London.

Kaminski, A. (2018). Der Erfolg der Modellierung und das Ende der Modelle. In *Technik–Macht–Raum*, pages 317–333. Springer.

Kaminski, A., Resch, M., and Küster, U. (2018). Mathematische Opazität. Über Rechtfertigung und Reproduzierbarkeit in der Computersimulation. In Friedrich, A., Gehring, P., Hubig, C., Kaminski, A., and Nordmann, A., editors, *Jahrbuch Technikphilosophie 2018: Arbeit und Spiel*, pages 253–278. Nomos Verlagsgesellschaft mbH & Co. KG.

Karaca, K. (2017). The interplay among experiment, simulation and theory at the large hadron collider: A network account of models in high energy physics experiments. Unpublished manuscript, presented at *EPSA17 Exeter*.

Kripke, S. A. (1972). *Naming and Necessity*. Harvard University Press.

Kvanvig, J. (2003). *The Value of Knowledge and the Pursuit of Understanding*. Cambridge: Cambridge University Press.

Larkoski, A. J., Moult, I., and Nachman, B. (2017). Jet substructure at the large hadron collider: A review of recent advances in theory and machine learning. *arXiv:1709.04464v1 [hep-ph]*.

Lenhard, J. (2006). Surprised by a nanowire: Simulation, control, and understanding. *Philosophy of Science*, 73(5):605–616.

Lenhard, J. (2011). Epistemologie der Iteration: Gedankenexperimente und Simulationsexperimente. *Deutsche Zeitschrift für Philosophie*, 59(1):131–145.

Llorente, J. and Cantero, J. (2014). Determination of the b-quark mass m_b from the angular screening effects in the ATLAS b-jet shape data. *Nuclear Physics B*, 889:401–418.

Mangano, M. L. and Stelzer, T. J. (2005). Tools for the simulation of hard hadronic collisions. *Annu. Rev. Nucl. Part. Sci.*, 55:555–588.

Marcus, G. (2018). Deep learning: A critical appraisal. *Computing Research RepositoryR*, abs/1801.00631.

Massimi, M. and Bhimji, W. (2015). Computer simulations and experiments: The case of the higgs boson. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics*, 51:71–81.

Mehra, J. and Rechenberg, H. (2000). *The Historical Development of Quantum Theory, Vol. 1*. Springer New York.

Morrison, M. (2015). *Reconstructing Reality: Models, Mathematics, and Simulations*. Oxford University Press.

Parker, W. (2018). Climate science. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. The Metaphysics Research Lab, Center for the Study of Language and Information (CSLI), Stanford University. online: https://plato.stanford.edu/entries/climate-science/ (checked 12/18).

Pritchard, D. (2014). Knowledge and understanding. In Fairweather, A., editor, *Virtue Epistemology Naturalized: Bridges Between Virtue Epistemology and Philosophy of Science*, pages 315–328. Cham: Springer.

Radder, H. (2009). The philosophy of scientific experimentation: a review. *Automated experimentation*, 1(1):2.

Riggs, W. D. (2003). Understanding 'virtue' and the virtue of understanding. In DePaul, M. and Zagzebski, L., editors, *Intellectual Virtue: Perspectives from Ethics and Epistemology*, pages 203–226. Oxford: Clarendon Press.

Ronen, R. (2017). Why & when deep learning works: Looking inside deep learnings. *CoRR*, abs/1705.03921.

Rudin, C. and Wagstaff, K. L. (2014). Machine learning for science and society. *Machine Learning*, 95(1):1–9.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533.

Russell, S., Moskowitz, I. S., and Raglin, A. (2017). Human information interaction, artificial intelligence, and errors. In Lawless, W., Mittu, R., Sofge, D., and Russell, S., editors, *Autonomy and Artificial Intelligence: A Threat or Savior?*, pages 71–101. Cham: Springer International.

Saam, N. J. (2017). Understanding social science simulations: Distinguishing two categories of simula-

tions. In Resch, M. M., Kaminski, A., and Gehring, P., editors, *The Science and Art of Simulation I: Exploring – Understanding – Knowing*, pages 67–84. Cham: Springer International Publishing.

Schembera, B. (2017). Myths of simulation. In Resch, M. M., Kaminski, A., and Gehring, P., editors, *The Science and Art of Simulation I: Exploring – Understanding – Knowing*, pages 51–63. Cham: Springer International Publishing.

Schurz, G. (2014). *Philosophy of Science. A Unified Approach*. New York, London: Routledge.

Schwartz-Ziv, R. and Tishby, N. (2017). Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*.

Stanford, K. P. (2006). *Exceeding Our Grasp. Science, History, and the Problem of Unconceived Alternatives*. Oxford: Oxford University Press.

Suárez, M. (2003). Scientific representation: Against similarity and isomorphism. *International studies in the philosophy of science*, 17(3):225–244.

van Fraassen, B. C. (2008). *Scientific Representation: Paradoxes of Perspective*. Oxford, New York: Clarendon Press.

Voss, H. (2013). Classification. In Behnke, O., Kröninger, K., Schott, G., and Schörner-Sadenius, T., editors, *Data Analysis in High Energy Physics: A Practical Guide to Statistical Methods*, pages 153–186. Wiley.