Algorithmic Normativities – Original Research Article

BIG DATA & SOCIETY

Big Data & Society
July–December 2019: 1–12

DOI: 10.1177/2053951719858742
journals.sagepub.com/home/bds
SAGE

Perspectives on algorithmic normativities: engineers, objects, activities

Jérémy Grosman[2] and Tyler Reigeluth[1]

Abstract

This contribution aims at proposing a framework for articulating different kinds of "normativities" that are and can be attributed to "algorithmic systems." The technical normativity manifests itself through the lineage of technical objects. The norm expresses a technical scheme's becoming as it mutates through, but also resists, inventions. The genealogy of neural networks shall provide a powerful illustration of this dynamic by engaging with their concrete functioning as well as their unsuspected potentialities. The socio-technical normativity accounts for the manners in which engineers, as actors folded into socio-technical networks, willingly or unwittingly, infuse technical objects with values materialized in the system. Surveillance systems' design will serve here to instantiate the ongoing mediation through which algorithmic systems are endowed with specific capacities. The behavioral normativity is the normative activity, in which both organic and mechanical behaviors are actively participating, undoing the identification of machines with "norm following," and organisms with "norminstitution". This proposition productively accounts for the singularity of machine learning algorithms, explored here through the case of recommender systems. The paper will provide substantial discussions of the notions of "normative" by cutting across history and philosophy of science, legal, and critical theory, as well as "algorithmics," and by confronting our studies led in engineering laboratories with critical algorithm studies.

Keywords

Machine learning, technical normativity, socio-technical normativity, Gilbert Simondon, neural networks, behavioral normativity

This article is a part of special theme on Algorithmic Normativities. To see a full list of all articles in this special theme, please click here: https://journals.sagepub.com/page/bds/collections/algorithmic_normativities.

Introduction

It is generally agreed upon that "algorithmic systems" implementing machine learning techniques have significant normative effects upon the ways items are recommended and consumed, the ways choices are taken and justified. However, the aspect and extent of those "normative effects" are subject to much disagreement. Some claim that "algorithmic systems" significantly affect the way norms are conceived and instituted. Others maintain that algorithmic systems are actually black boxing more traditional normative processes (Diakopoulos, 2013). The problem partly lies in accounting for the different kinds of "normativities" that are, and can be attributed to, "algorithmic systems." Our paper proposes a substantial discussion of "algorithmic normativities" by accounting for the practices we witnessed in diverse engineering laboratories and by confronting literature in sociology, history and philosophy of technology (see for instance Beer 2017 and Ziewitz 2016). Before going any further let us begin by providing working definitions of "algorithm" and "norm," the value of which should be measured by the conceptual and practical relations they give rise to

[1]Université Libre de Bruxelles, Centre de Théorie Politique, Belgium
[2]Université de Namur, Research Center in Information, Law and Society (CRIDS), Belgium

Corresponding author:
Tyler Reigeluth, Universite Libre de Bruxelles—Campus du Solbosch, Bruxelles 1050, Belgium.
Email: tyler.reigeluth@ulb.ac.be

when thinking about, or interacting with, algorithmic systems.

From an engineering standpoint, algorithms can be helpfully approached as stabilized and formalized computational techniques that scientists and engineers recognize and manipulate as an object throughout a variety of disparate *inscriptions* (e.g., formalized expressions, diagrammatic representations, coded instructions, traces of executions) and through a variety of disparate *actions* (e.g., formal proofs, intuitive manipulations, material implementations, concrete experimentations). Computational techniques only become algorithms when calculators find it useful to stabilize and formalize them (see Hill, 2016). Two further features must be added to make sense of what engineers commonly call "algorithms." First, algorithms are, for most computer scientists, objects about which knowledge can be produced and communicated. Second, algorithms are, for most software engineers, things likely to be recognized in many different systems. In other words, to warrant the qualification of algorithm, a stabilized and formalized computational technique needs to possess, and be invested with, a certain significance within a certain practice (Pickering, 1995).

From a generic standpoint, vital, social or technical norms can be approached either as a *description* of a range of events, or as a *prescription* for a range of actions—it has convincingly been argued that, in most cases, these descriptive and prescriptive meanings cannot be satisfactorily disentangled (Canguilhem, 1943; Putnam, 2002). Three features should be emphasized. First, a norm expresses a relation between a set of disparate changes—social norms, for instance, typically describe or prescribe ways of behaving within specific situations. Second, a norm reveals itself when individuals are propelled to contrast their ways of behaving with others'—again, social norms typically manifest themselves whenever individuals experience specific situations as being problematic or conflictual. Third, a norm is both *instituted* and *followed*—social norms are typically instituted as the solution to a collective experience of a problematic or conflictual situation. The notions of "normativity" and "normalization," to which we will hereafter refer, denote the processes through which an activity, respectively, comes to institute or to follow specific norms.

The following pages further investigate how distinct kinds of normativities come into play through these socio-technical assemblages we call algorithmic systems.

- The first section exposes some of the minute actions through which engineers willingly or unwittingly infuse technical systems with social norms. The design of a surveillance system will serve to unfold the ongoing mediations (e.g., collecting data, defining metrics, establishing functions) that allow engineers to normalize the behaviors of algorithmic systems (e.g., how algorithms improve their ability to discriminate between threatening and nonthreatening behaviors).
- The second section shows how successive changes in algorithms' structures and operations reveal distinct kinds of *technical* normativities—be they schematic, formal or material. A liminal genealogy of artificial neural networks shall provide a powerful illustration of this dynamic by engaging with their concrete functioning as well as their unsuspected potentialities (e.g., how the invention of optimization algorithms or parallel processors considerably transformed these networks).
- The third section proposes to conceive of learning machines as exhibiting a genuine social or *behavioral* normativity, insofar as their activities can be seen as exhibiting both "norm following" and "norm instituting" processes. The case of recommender systems shall allow us to identify markers around which these two processes seem to irremediably entangle (e.g., the intractability, the randomness, and the interactivity of algorithmic systems' behaviors).

The conceptual focus of the paper does not dispense us from indicating the empirical sources thanks to which we ground our developments. The surveillance system's empirical material derives mainly from ethnographic data collected between 2012–2015 and 2016–2019 while participating in two Research and Innovation projects (e.g., consortium meetings, interviews with engineers, laboratory visits). The neural network's empirical material is chiefly informed by readings of original technical literature (Rosenblatt, 1962), contemporary technical literature (Bishop, 2006; Mitchell, 1997) and published historical accounts (McCorduck, 1983; Crevier, 1993; Olazaran, 1993; Pickering, 2010). The recommender system's material freely draws upon knowledge acquired along an empirical study following a computational experiment on recommender systems conducted between 2016 and 2017, as well as of the consultation of other more traditional resources such as scientific literature and technical blogs.

Socio-technical normativity and surveillance system

The recent applications of machine learning techniques in assisted and automated decision-making raise hopes and concerns. On the one hand, there are hopes about the possibility to have an informational environment tailored to specific ends or the possibility to increase a

firm's commercial revenues. On the other hand, there are concerns about the inconsiderate discriminations machine learning results might conceal or the unprecedented possibilities for governing ushered in by these techniques. Such concerns appear to be exacerbated by the patent difficulties to hold beings accountable, either because the responsibility of the decision is not attributable in any straightforward way, whether to an algorithmic system or to a moral person, or because the understanding of processes leading to case-specific decisions is threadbare (Burrell, 2016).

We propose to address these key issues by identifying *sites* that are crucial for engineers' representations and interventions as they negotiate encounters between social imperatives and technical constraints (see Bijker et al., 1987; McKenzie and Wajcman, 1985). Two such sites have been identified: *the metrics' definition* and *the databases' collection*.[1] Each respectively enables engineers to evaluate and design their algorithmic systems. For the most part, our claims are grounded in empirical inquiries we undertook over the last five years, within different research groups. The material collected consists of observations, discussions, interviews, and reports. Thus, our first task is to describe the actions through which engineers produce algorithmic systems following specific socio-technical norms as well as the actions through which engineers produce knowledge about their algorithms' behaviors. One case study in particular, the "Privacy Preserving Protection Perimeter Project," will illustrate the matters discussed.[2]

Funded by the European Commission between 2012 and 2015, this European "Research and Innovation" Project gathered a dozen private companies, research universities, and public institutions with the aim of designing a system able to automatically detect threatening behaviors. This rather abstract endeavour inevitably concealed multiple concrete technical challenges, the most predominant of which were the combination of visual and thermal cameras with microwave and acoustic sensors, as well as the development of robust and efficient detection, tracking and classification algorithms (i.e. working in real-time within uncontrolled environments). The consortium rapidly pinned down two "use cases" that would put them to work: a Swedish nuclear power plant (the Oskarshamns Kraftgrupp AB) and a Swedish nuclear waste storage facility (the Centralt mellanlager för använt kärnbränsle). The task impelled the engineers to undertake a socio-technical inquiry enabling them to spell out the various qualities with which users (i.e. the security workers) expected the system to be endowed.[3]

Understandably, the security workers expected the system to maintain the number of false alarms below a certain threshold—otherwise the system would end up *dis*organizing the perimeter surveillance instead of organizing it. The project's partners usually reframe the requirement in slightly more technical terms: they want to minimize the number of "false positives" (i.e. when a behavior is mistaken as threatening) and "false negatives" (i.e. when a behavior is mistaken as nonthreatening). The engineers then translate the socio-technical requirements into fairly simple and intuitive mathematical expressions, called metrics, that set standards for attributing numerical values to different aspects of the system's behaviors, aspects that are deemed particularly important and that can be empirically observed (Dewey, 1939; Porter, 1996). The matter thus lies in understanding how the human-based act of surveilling through categorization can be transformed into a machine-based problem of classification.

The very notions of "false positive" or "false negative" suppose that an algorithm-based classification can be compared to, and overturned by, a human-based classification. This human-based classification, acting as the reference norm against which all machine-based classifications are to be evaluated, is called the *ground truth dataset* (Jaton, 2017). The "dataset" part here refers to an ensemble of recorded scenes, each containing as many numerical sequences as there are available sensors (i.e. visual videos, thermal frames, sound recordings, and Doppler time-series). It provides the system with concrete instances of an abstract classification problem. These instances consist of actual scenes the algorithm must assign to its "correct" category (i.e. threatening or nonthreatening). The "ground truth" part refers to the categories or labels which humans—e.g. domain experts, computer scientists or Amazon turkers—have attributed to each sequence. It supplies the system with answers to the problem: the algorithm now has an external check for assessing the correctness of its classification.

How are these ground truth datasets concretely constituted? Constituting relevant datasets for algorithmic systems presents a genuine socio-technical challenge. Building on their socio-technical inquiry—which involves reading regulations, visiting sites, and interviewing workers—the engineers envisioned scenarios that would seek to encompass the range of threatening and inoffensive behaviors the system was expected to handle (e.g., a jogger running near the nuclear power plant, a boat rapidly approaching the waste storage facility, a truck driving perpendicular to one of the site's fences, etc.). The engineers then gathered on two occasions, each time for about a week, to enact and record the norms scripted in their scenarios (Figure 1). The scene itself is worth visualizing: engineers awkwardly running through an empty field sprinkled with cables and sensors, mimicking the threatening or inoffensive behaviors they want their algorithms to learn.

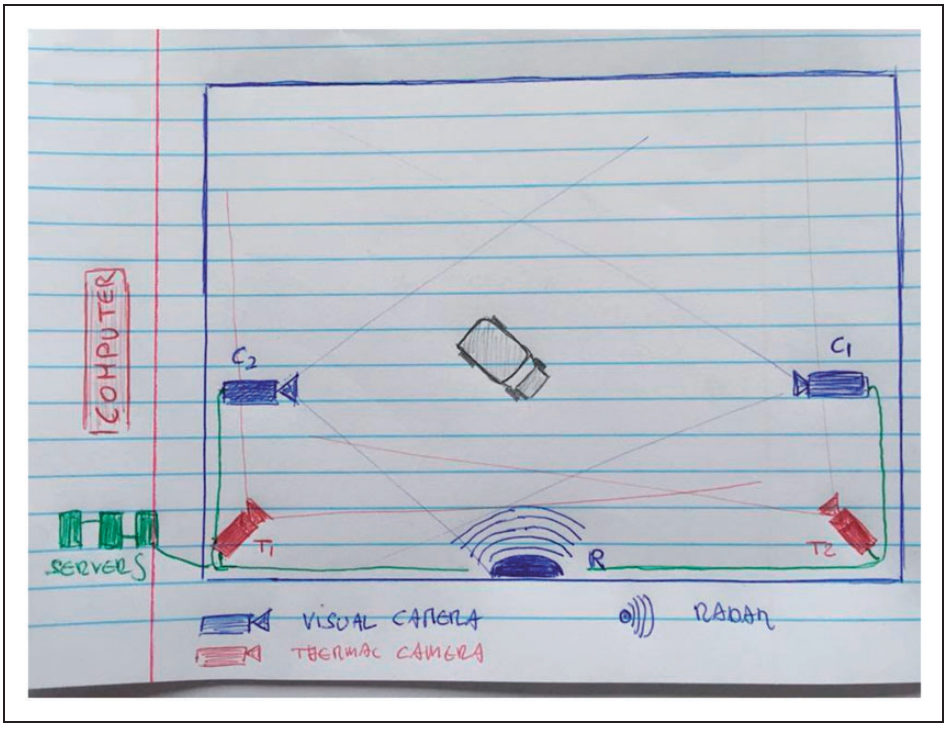

Figure 1. Personal sketch of the data collection field observed in United Kingdom in June 2014.

Returning to their laboratories, they spent a considerable amount of time annotating each frame of each sequence produced by each sensor, e.g. circling moving bodies, labeling their behaviors, etc.[4]

With their metrics defined and their data collected, engineers can, at last, *train* their algorithms, i.e. lead them to embody socio-technical norms, and *evaluate* them, i.e. measure relevant dimensions of their activities. The training process usually implemented first requires engineers to load and initialize the algorithm they wish to train (e.g., support-vector machine, random forest, artificial neural networks, etc.) on their laboratory's computer clusters. They then run a script which, as shown in Figure 2, (i) supplies the algorithm with a batch of scenes which algorithms are asked to classify (i.e. classification), (ii) compares the algorithm's guesses with the ground truth (e.g., evaluation), and (iii) modifies a number of the algorithm's parameters to improve the ways it reacts to specific videos (i.e. correction)—before iterating back to (i) until the training dataset is emptied.

We have argued here—against growing claims that contemporary machine learning practices and techniques are progressively slipping through our hands—that approaches combining empirical and critical perspectives are likely to provide us with the means to productively engage with engineering practices, while also opening up possibilities of critique and intervention. The argument essentially rested on two claims. On

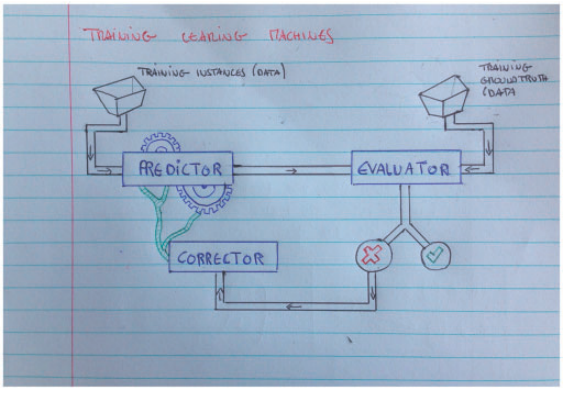

Figure 2. Personal sketch of a slide used by Robin Devooght on June 2017.

the one hand, the "metrics," which mathematically express the system's socio-technical norms, equip engineers with a way of assessing the comparative values of different algorithms. On the other hand, the "ground truth dataset," which comprises the actual and possible instances the system is supposed to handle, provides engineers with local norms for telling the algorithm how to learn in specific instances. *In both cases, the norms are instituted by the engineers and are expected to be followed by the machines*. We believe that by identifying two distinct sites where engineers

decisively mediate between social and technical constraints, we can productively transform the demands we place on algorithmic systems and provide robust indications as to where to intervene in critical cases in which normative conflicts and problems arise.[5]

Technical normativity and artificial neural networks

The recent craze surrounding "machine" or "deep" learning is usually explained in terms of "available data" (i.e. infrastructural changes made data collection accessible), "algorithmic advances" (i.e. technical changes made data processing possible) or "economic promises" (i.e. applications will make data particularly valuable). In focusing on the algorithmic advances, the following paragraphs propose to further explore three *kinds* of norms (imaginal, mathematical, and material) algorithms manifest whenever engineers attempt to endow them with specific capacities. The genealogy of artificial neural networks—a subset of "machine learning" techniques whose applications may be seen in surveillance systems (see "Socio-technical normativity and surveillance system" section) as well as recommender systems (see "Behavioral normativity and collaborative filtering" section)—will shed light on the *technical* norms algorithms impose on engineering practices. Before unfolding the technical transformations artificial neural networks underwent, we will briefly sketch their inception.

The earliest trace of the artificial neural network scheme can be found in a technical report authored by the American psychologist Frank Rosenblatt (Rosenblatt, 1962) when based at the Cornell Aeronautical Laboratory. By that time, several researchers were interested in exploring the nature of learning processes with the embryonic tools of automata theories. Distinctively, Rosenblatt decided to address the problem of learning processes by both investigating neurophysiological systems *and* constructing intelligent machines. Artificial neural networks would eventually emerge from these projects exploring "natural" and "artificial" learning, as the blueprint of a machine capable of "perceiving" and "recognizing" visual patterns, as a machine capable of "learning" to differentiate between geometrical forms. Thus, a neural image of intelligence gradually turned into concrete technical objects that would later be programmed for the IBM 704 and hardwired as the MARK I.[6]

It has been convincingly argued that the singularity of technical objects is best grasped through the schemes describing their operations within different environments, rather than the uses to which they are subject or the practicalities of their actualization (Simondon, 1958: 19–82). Thus, the singularity of artificial neural networks lies in their technical scheme, rather than the general regression, classification or clustering purposes they serve, or their specific implementations in Python's Theano or TensorFlow libraries. A technical scheme describes the *parts* composing the technical object and relates their *operations* within the technical object's functioning (Polanyi, 1966: 38–40). Most schemes are constructed from material traces (objects, descriptions, diagrams, etc.) by technicians or historians interested in thinking about, and intervening upon, specific technical objects.

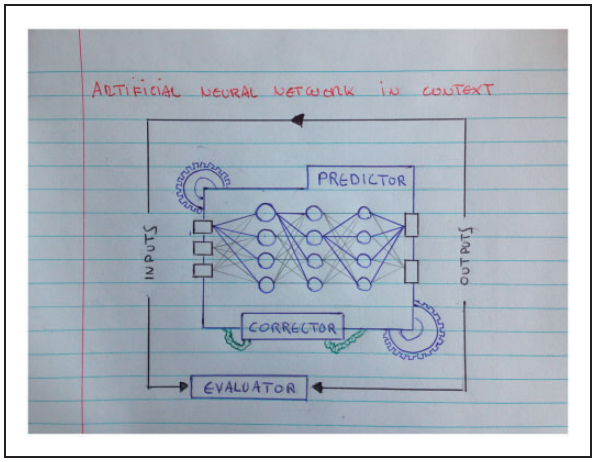

Figure 3. Diagram of artificial neural network within an environment.

In the case of artificial neural networks, the scheme always couples the algorithm to an environment (Figure 3). In Rosenblatt's design, light sensors allow the network to capture signals from its environment that end up being classified and light emitters enable the network to display the results of its classifications. Crucially, Rosenblatt articulated the artificial neural network and its environment with a genuine feedback mechanism, which would work as an experimental controller for correcting the emitter's output until it produced the desired response.

The artificial neural network's scheme consists of two main parts: neural units and synaptic edges (see Figure 4). Each neural unit *receives* an incoming signal and *produces* an outcoming value—the neurons

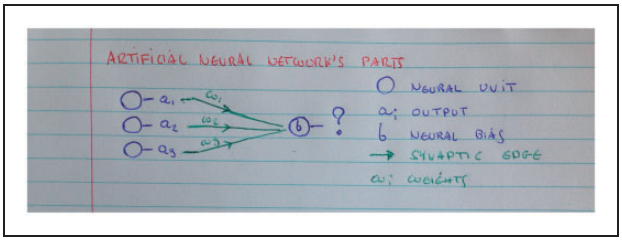

Figure 4. Diagram of the parts composing artificial neural networks.

generally contain numerical parameters (i.e. the bias) and share an activation function. Each synaptic edge connects two separate neural units—the synapses generally contain numerical parameters (i.e. the weight). The question now is how these parts interact.

The first operation is prediction. It typically moves from left to right and leaves the network unchanged (see Figure 5). A cascade of operations combines the input signal and the network's parameters (i.e. synapses' weights and neurons' biases) in order to produce specific output values—depending on the problem, the signal is classified as "triangle" or "circle," as "threatening" or "nonthreatening," etc.

The second operation is learning. It typically moves from right to left and alters the network's parameters (see Figure 6). A cascade of operations compares the outputs' values (i.e. the datasequence the algorithm classified) and the ground truth information (i.e. the sequence data the engineer annotated) in order to update and correct the network's parameters (i.e. synapses weights and neurons' biases).

More than a mere description encompassing different but related technical objects, this technical scheme opens a field of possible manipulations for engineers, ranging from minor modifications (How many neurons? How many synapses? etc.) to major modifications, which lead to new lineages of artificial neural networks (What if the neurons' activation function changes? What if peculiar synaptic connections are allowed? etc.). Thus, technical schemes are always both objectal and imaginal (Beaubois, 2015). If technical objects "have a life of their own," to borrow Ian Hacking's (1983: 262–275) words, the problem then lies in accounting for their schemes' becoming, as it mutates through but also resists, successive inventions (Leroi-Gourhan, 1945; Simondon, 1958: 19–82). The following paragraphs briefly sketch two other episodes neural networks went through between the early 1950s and the late 2000s, further illustrating this dynamic of invention and resistance.

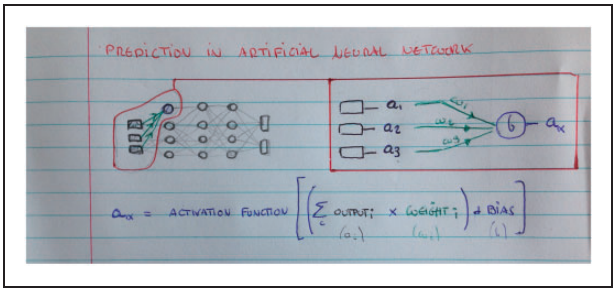

Figure 5. Diagram of the prediction's operation in artificial neural networks.

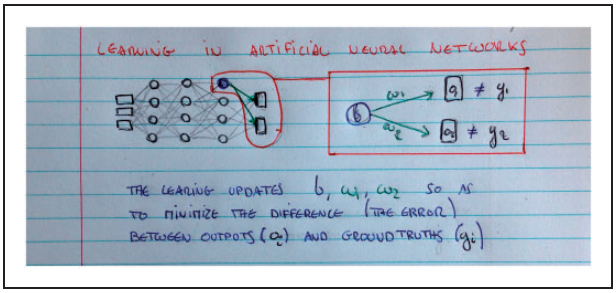

Figure 6. Diagram of the learning's operation in artificial neural networks.

The experimental, mathematical, and commercial successes of artificial neural networks rested upon their ability to adjust synapses' weights and neurons' biases in order to recognize incoming patterns—i.e. to learn to classify. However, the limitations of early artificial neural networks rapidly became apparent and problematic (see Olazaran, 1993: 347–350). The single-layer artificial neural networks proved incapable of solving important families of problems, e.g. they could not learn to recognize resized or disconnected geometrical patterns (Rosenblatt, 1962: 67–70). On the other hand, learning capacities of multilayer artificial neural networks appeared to depend more on engineering skills than on their intrinsic qualities, e.g. there existed no procedure guaranteeing that the learning would converge (see Olazaran, 1993: 396–406). These combined limitations to algorithmic learning capacities are traditionally seen as the onset of a significant drop in financial and scientific interest in artificial neural networks and artificial intelligence.

It was not before the mid-1980s that a satisfying learning rule, named "backpropagation," would be identified for multilayer artificial neural networks—more or less simultaneously and independently by Paul Werbos, David Rumelhart, and Yann Le Cunn (Olazaran, 1993: 390–396). In a nutshell, the learning algorithm measures the difference between the network's and the environment's responses (i.e. error function); it then computes the relative contribution of each synapse and neuron to the measured error (e.g., chain rule's partial derivatives); and finally updates the synapses' weights and neurons' biases so as to reduce the overall error. The learning problem is thus recast as an optimization problem, in which the main objective is to minimize classification errors. Backpropagation's mathematical scheme significantly extended artificial neural networks' problem-solving capacities.

Thereafter, in the late 1980s multilayer artificial neural networks gave rise to interesting applications in areas such as natural language processing, handwriting recognition, etc. (Olazaran, 1993: 405–411). However, the computational resources required during the learning phases quickly presented a significant impediment: depending on the number of layers, on the size of the

datasets and on the power of the processors, it could take up to several weeks to find weights and biases minimizing classification errors. Thus, the applications using artificial neural networks did not stand out from other machine learning algorithms (e.g., ''k-nearest neighbors,'' ''random forests,'' ''support vector machines,'' ''matrix factorization''). Their more recent proliferation—often referred to as ''deep learning''—appears to be intimately tied to advent and generalization of massively parallel processing units.

In the late 1990s, Nvidia Corporation, one of the largest hardware developers and manufacturers for video games, equipped their graphic cards with specific-purpose processors. Contemporary video games typically involve large-scale mathematical transformations that can be performed in parallel, e.g. the changing values of the millions of pixels displayed on our screens are usually processed simultaneously, instead of being calculated sequentially. In the late 2000s, researchers in computational statistics and machine learning revealed the mathematical affinities shared by the graphical operations in video games and the learning operations in pattern recognition. Indeed, both involved countless matricial calculations that could be distributed over several processing cores and executed independently. This rapid outline indicates the extent to which material developments brought forth or actualized latent algorithmic capacities.

Thus far, our approach has emphasized three kinds of technical norms algorithms impose on engineering practices. The first has to do with the algorithm's imaginal affordances in relation to the development of technical systems—in this case, a single abstract representation of neural processes brought about multiple lineages of algorithms. The second deals with the mathematical properties of the algorithm's structures and operations—in this case, the invention of a convergent algorithm significantly extended the abilities of multilayer artificial neural networks. The third bears on the interplay between material constraints and computational possibilities—in this case, the advent of Graphical Processing Units transformed the range of problems algorithms can solve. Thus, the genealogy of artificial neural networks led us not only to a concrete account of algorithmic processes, but more importantly still, to a comprehensive understanding of the imaginal, mathematical and material dynamics that drive their technical becoming.

Behavioral normativity and collaborative filtering

It is generally observed that computers blur an entrenched ontological dichotomy between two kinds of beings: living organisms and automated machines (see Fox-Keller, 2008; 2009). Kant's far-reaching conception of self-organization famously epitomizes the matter at stake: organisms' norms are thought to be intrinsic to their activities—organisms are, in this sense, ''norm-instituting'' beings—and machines' norms are thought to be extrinsic to their activities—machines are, in this sense, ''norm-following'' beings (Barandiaran and Egbert, 2014). The most convincing conceptual markers grounding this longstanding dichotomy are generally to be found in the widely shared reluctance to attribute ''problems'' and ''errors'' to behaviors exhibited by machines, computers, and algorithms (Bates, 2014). Thus, technical errors are usually thought to be *rewritable* either as engineering *failures* or socio-technical *problems* (see, respectively, Canguilhem, 1943; Turing, 1948).

Contemporary algorithmic systems appear to further complicate any a priori partitions between distinctively organic and machinic behaviors. Indeed, *learning* machines are, to a certain extent, capable of modifying their structures so as to respond to modifications in their milieu without any specific human interventions. In this regard, they are often seen as exhibiting a form of behavioral *plasticity*, which was long held to be distinctive of organic activity. The minute operations followed by learning machines and the effects they produce, currently remain beyond satisfactory understandings. As such, there is an aspect of what they do that is *intractable*, that lies beyond our current capacities of prediction and control. This section further investigates these *underdetermined* behaviors and suggests that both the human *and* algorithmic aspects of machine learning systems need to be approached as genuinely behaving, that is as performing together norm-following and norm-instituting activities.

The case of collaborative filtering recommender systems deployed on commercial platforms such as YouTube or Netflix can help us ground these considerations (Seaver, 2018). In a nutshell, these systems seek to foster unprecedented interactions between items and users by looking at past interactions between similar items and users. Practically speaking, the *metrics* deployed for evaluating the performances of such systems generally attempt to measure different aspects of user engagement and the *data* collected for training these algorithmic models largely consists of user interaction histories. The concrete processes, which are to a large extent unscripted, remain difficult to explain, even retrospectively, for the engineers who designed the algorithms. The dynamics of imitation and variation performed on these platforms need to be conceived, we argue, as a *social activity*, involving an interactive and iterative process between the users' and the algorithms' behaviors.[7]

The distinction between norm-instituting and norm-following can be further understood as a difference between two processes of determination. Most technical entities, it has been argued, exhibit a relative ''margin of indetermination'' that makes them more or less receptive to ''external information'' and responsive in terms of ''internal transformations'' (see Simondon, 1958: 134–152), e.g. the progressive wear and tear of a bolt connecting metal parts allows for their mutual adjustment, an engine's governor constantly regulates the train's speed despite changes of load and pressure, a warehouse logistically adapts to a changing book order, a recommender system responds to newly received interactions. The distinction between norm-instituting and norm-following can thus be reframed as a distinction between two kinds of determination: a determination will be said ''convergent'' or ''divergent'' depending on whether it *restricts* or *expands* the behavioral variability of a technical entity. Both empirically and conceptually speaking, the difficulty thus lies in being able to account for the divergent determinations of certain machine learning processes.

In this light, the singularity of machine learning applications should be understood in terms of the relative significance of the margin of indetermination exhibited. Indeed, in most machine learning applications, the behavior of the algorithmic system (i.e. the specific predictions displayed) may be periodically transformed (i.e. the values of the model are updated) depending on the relevant incoming information (i.e. particular sets of data). By and large, engineering work is dedicated to restraining the recommender system's variations—instituting metrics, defining error functions, cleaning datasets—and aligning them with the companies' interests. On the one hand, the determinations of learning may be seen as converging toward rather specific pre-instituted norms. However, the actual learning process, as Adrian Mackenzie (2018: 82) has argued, has more to do with stochastic function-finding than with deterministic function-execution—the resulting model corresponds to one among many possible configurations (see also Mackenzie 2015). Thus, and on the other hand, the concrete determinations of the system's behavior can hardly be seen as converging toward any pre-instituted norms.[8]

Let us take a closer look at the actual processes that unfold when producing recommendations. Collaborative filtering techniques typically seek to achieve a delicate balance between sameness (always recommending similar items) and difference (only recommending different items) solely by looking at and drawing on patterns of interactions between users and items. The collaborative machines—that may well be instantiated by many different techniques, such as k-nearest neighbors, matrix factorization, neural networks—are usually interpreted by recommendation engineers as capable of producing models that represent *actual* relations and suggest *potential* relations between users and items. More concretely, each learning step supposes that the *error function* (also called loss or objective function) measures the recommendation's quality and that the correction algorithm (e.g., back-propagation) adjusts the model's parameters accordingly. We can now, with these developments in mind, rework the problem with which we opened the section: error as a marker for thinking the human–machine distinction.

If, indeed, the very process of learning rests on the possibility of erring (Vygotsky, 1978), we propose, following David Bates (Bates and Bassiri, 2016; see also Malabou, 2017), to seize both entangled senses in which erring may be understood: erring as ''behaving in unexpected ways'' (i.e. errancy), and erring as ''susceptible of being corrected'' (i.e. error). In both cases, erring supposes that a relation between an individual and an environment be experienced as problematic, that the norms instantiated within behavioral performances are experienced as only certain ones among many possible others. The passage from error to errancy depends on recognizing social activity as open-ended enough for error to appear not only as the mere negation of a norm, but rather as the affirmation of a different norm. We would like to suggest, here, that a more cautious look at learning processes could help us frame these algorithmic systems *as erring within a broader social activity*, in this instance: the algorithmic production of cultural tastes.

Two remarks should suffice here to pin down what is at stake. First, the overall learning process is open-ended: although each learning step stops when the model's parameters minimize error and overfitting, the overall learning process consists of an indefinite sequence of learning steps, which are periodically picked up again once there is enough fresh data from incoming interactions. Second, the recommendations are ''moving targets'' (Hacking, 1986): the learning process is shaped by interactions as much as it shapes interactions, i.e. recommender systems learn *and* produce interactions. We therefore contend that the iterativity and interactivity of recommending activities are grounds for conceiving classification operations performed by recommender systems as targeting both certain user behaviors as well as certain algorithmic predictions. The dynamic of these ''looping kinds'' (Hacking, 1986), brings recommendation, as a social activity, closer to errancy than error. Indeed, what would it even mean from this perspective to produce a false recommendation?

The algorithmic system can therefore be seen as genuinely *behaving*, that is as performing an

algorithmic *activity* exhibiting its own form of normativity. The behavioral normativity, once the notion of behavior is released from individualized moorings and reconnected with the social activity within which it unfolds, can be productively conceived as a dynamic of norm following *and* norm institution—rather than the instantiation of social norms within technical ensembles (i.e. socio-technical normativity) or the instantiation of technical norms through acts of invention (i.e. technical normativity). In this specific case, the recommender system needs to be conceived as *a* social partner behaving within a determined social activity: their behaviors bear social significance and affect the value of other behaviors.

Thus, the notion of behavioral normativity leads us to reconsider how the redistributions of norm-following and norm-instituting behaviors between humans and machines actively reshape social activities. We might ask then: ''What is learning and where is learning occurring within the behavioral distributions of a given activity?'' In doing so, the primary *focus* or *locus* of analysis is productively displaced toward the unfolding of social activities, thereby indefinitely deferring the quest to reveal the ''true norms'' that inform algorithmic systems (e.g., ''we are *manipulated* by algorithms'') or social systems (e.g., ''algorithms are *biased*''). We can now better attend to what it means and why it matters for machine learning algorithms to behave in unforeseen and unexpected ways, thus opening the prospect that they become sites of normative invention. We hope, in this way, to have contributed to the concerted efforts that need to be made to integrate algorithmic behaviors into the field of action it analyzes by offering conceptual and methodological tools for claiming their effects. Our contention is that one way of addressing this challenge lies in showing how this normativity makes sense within culture.

The tension experienced by most people when confronted with technical systems can, in part, be understood in terms of the difficulties they have in experiencing and making sense of how, where, and when machine behaviors perform with ours. The cultural images of technical systems have traditionally allowed us to stabilize the perceptive and motor anticipations determining our relationships with them, e.g. the individual body performing with a simple tool or the industrial ensemble organizing human and machine labor (Simondon, 1965-1966; Leroi-Gourhan, 1965). The problems, in the case of learning machines, are tied to the variable distributions of human and machines behaviors (Collins, 1990: 14) and to the unfamiliar ability algorithmic systems have of instituting norms. The tension can thus in part be reframed as the problem of understanding these behaviors in relation to the social activities within which they unfold and that they take part in shaping.

The concept of behavioral normativity foregrounds the importance of social activity and the afferent margin of indetermination it allows behaviors to inform. If there is no room left for error, then all behaviors that do not execute or follow the established, programmed norm will be disregarded or repressed as useless or inefficient. If, on the other hand, those behaviors that do not perform as expected are attributed value and attended to, they can participate in instituting a new norm-following dynamic. Thereby, machine learning forces us to reconsider long standing divides between machines' and organisms' behaviors, between those behaviors that repeat and those that invent norms. Our conceptual proposition invites alternative and more demanding normative expectations to bear on engineering and design practices whereby the margin of indetermination of an algorithmic system would be increased rather than reduced.

Conclusion

The overall aim of this paper was to approach algorithmic normativities in a different light, with different questions in mind, with different norms in sight.

The first investigation into socio-technical normativity showed how engineering practices come to stabilize and embed norms within certain systems by setting up plans or programs for learning machines to execute (metrics, ground truth dataset, optimization function). The socio-technical perspective, although a necessary starting point, is insufficient if taken in and of itself. Indeed, it demands that we be able to define what is ''social'' and what is ''technical'' within a given system, and how their given normativities come to be entangled. In this light, we proposed to qualify social and technical normativities in terms of operations they perform, rather than properties they possess.

This led us to look at different algorithmic systems from the standpoints both of their *technical* operations and their social or *behavioral* activities. The section on notion of technical normativity sketched the mutations of an algorithm's technical scheme and exposed how its norms both induce and resist invention, thereby granting technical objects imaginal, mathematical or material consistencies. The final section on behavioral normativity allowed us to consider certain conditions (indetermination, divergence, errancy) under which an algorithm can be seen as taking part in the norm-following and norm-instituting dynamics that characterize social activities.

This normative pluralism can help understand how contemporary algorithmic systems, comprised of multiple structures and operations, simultaneously fulfill

engineering aims, express technical resistances and participate in ongoing social processes. Generally speaking we can advance that:

- the aims engineers pursue always depend upon certain algorithmic capacities, in this sense algorithmic techniques normalize engineering practices (i.e. the objective function always depends on an efficient learning rule);
- the socio-technical system's behavior is subject to engineering aims that normalize its learning activity (i.e. the metrics and the ground truth datasets determine what counts as an error);
- the algorithmic system's learning processes unfold genuine norm instituting behaviors (i.e. the system's outputs periodically affect the very activities that have to be learned).

The value of our approach lies in its ability to provide a nuanced account of what is too often presented as *one* opaque, impenetrable or ethereal *system*. Algorithmic system*s* must instead be seen, we argue, as inhabited by *normative tensions*—between technical, socio-technical, and behavioral normativities. We sketched this pluralism with specific practical and theoretical problems in mind. No doubt others would be led to explore different normativities pervading algorithmic systems. Our effort has largely consisted in showing that behind each ''norm-following'' instance there is an institution, and conversely, that every institution requires its norms to be performed, to be played out, even at the risk of them being transformed by their very performance.

Acknowledgement

We would like to thank our friends and colleagues Nathan Devos and François Thoreau for discussions and criticisms, the two anonymous reviewers that made very productive critiques and suggestions, as well as the editors of this issue, Francis Lee and Lotta Lotta Björklund Larsen.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This paper was in part made possible thanks to the support of the Belgian FRS-FNRS funded research project ''Algorithmic governmentality'' which sponsored both Jérémy Grosman's and Tyler Reigeluth's PhD research. In addition, J. Grosman benifited from H2020 funding made available by the European Commission for the ''Privacy Preserving Protection Perimeter Project'' and ''Pervasive and User Focused Biometrics Border Project''.

ORCID iD

Tyler Reigeluth https://orcid.org/0000-0002-9908-1052

Notes

1. We do not claim that these sites are the *only* sites where social and technical normativity are entangled—one could also consider: (i) the choice of the relevant features (i.e. that select and neglect what counts as a relevant information, such as speed, trajectories or colors); (ii) the choice of the objective function (that determine the learning by providing a computational metric); (iii) the choice of the machine learning algorithm (that constrains the kind of things to be learned, such as sets or series).
2. The two inquiries, particularly relevant to the argument presented here, were led in the course of another two European ''Research and Innovation'' initiative, respectively, titled ''Privacy Preserving Protection Perimeter Project'' (2013–2016) and ''Pervasive and User Focused Biometrics Border Project'' (2016–2019). The data collected, during these projects, consisted of minutes of consortium meeting, attendance to demonstrations, as well as 30 semi-directed interviews with engineering partners. Grosman J (2018) Ethical and social issues, ''Technical Report in the frame Pervasive and User Focused Biometrics Border Project'' (PROTECT). The most readable general introduction to machine learning is undoubtedly Mitchell (1997). Bishop's (2006) textbook is also an invaluable resource. The course given by Andrew Ng (2011) at Stanford is an equally good place to start.
3. To give a more concrete sense of the specific challenges that need to be overcome: the system needed to remain indifferent to weather variations (e.g., Swedish snow, fog, and sun) as well as to the surrounding fauna (e.g., squirrels, rabbits, foxes, moose, etc.).
4. The ''data collection'' happens to be of tremendous importance not only, from the perspective of the power plant and waste facility's staffs, for getting the system to work correctly, but also, from the perspective of engineers working in computer vision, for getting standard datasets against which to compare their algorithms—mainly: providing benchmarks and organizing competitions.
5. Considering just one short example showing some consequences of our approach to the so-called opacity and unaccountability of algorithmic systems, the problems of algorithmic discrimination can be constructively reappraised as the problems of (i) characterizing the suspected social discrimination, (ii) constructing metrics likely of measuring such discrimination (and ruling out inconsiderate algorithms), and (iii) spelling out requirements that datasets must meet to avoid such discriminative consequences.
6. The empirical material concerning the genesis of neural networks builds on an extensive reading of the existing historical literature: McCurdock (1983), Olazaran (1993), Kay (2001), Pickering (2010), Cardon et al. (2018) as well

as selected readings of the primary literature: Rosenblatt (1962). The contemporary literature provides valuable technical details, see for example the short presentation in Mitchell (1997), the more substantial treatment in Bishop (1995) and the online course given by Geoffrey Hinton for Toronto University dedicated to neural network and available on the Coursera platform in 2012. For a more general overview of the period, see for instance Heims (1991) and Edwards (1996).

7. The most relevant textbooks are Francesco Ricci, Lior Rokach, Bracha Shapira and Paul B. Kantor (2011) and Aggarwal (2016). The annual proceedings of the ACM conference on *Recommender Systems* provide an invaluable overview of the public and private research conducted in recommender systems. Netflix's or Spotify's technical blogs, as well as the less official blogs of their employees, give invaluable insights into recommendation practices. The account is also informed by an empirical inquiry, led between 2016 and 2018, during which we documented a series of computational experiments attempting to uncover the capacities of recurrent neural networks for predicting sequence of interactions for the offline version of a collaborative filtering based recommender problem.
8. This argument rests on the rather reasonable assumption that the empirical learning of machines cannot be completely reduced to the mathematical convergence of a function, the role of which is to find the extremal values of another function—minimization in the case of what is called error or loss function, maximization in the case of what is called objective function.

References

Aggarwal CC (2016) *Recommender Systems: The Textbook*. New York: Springer.

Barandiaran X and Egbert M (2014) Norm-establishing and norm-following. *Artificial Life* 20(1): 5–28.

Bates D (2014) Unity, plasticity, catastrophe: Order and pathology in the cybernetic era. In: Killen A and Lebovic N (eds) *Catastrophe: History and Theory of an Operative Concept*. Berlin: De Gruyter, pp. 33–54.

Bates D and Bassiri N (2016) *Plasticity and Pathology. On the Formation of the Neural Subject*. New York: Fordham University Press.

Beaubois V (2015) Un schématisme pratique de l'imagination. *Appareil*, 16.

Beer D (ed.) (2017) *Information, Communication and Society* special issue: The Social Power of Algorithms 20(1).

Bijker WE, Hughes T and Pinch TJ (1987) *The Social Construction of Socio-Technical Systems: New Directions in the Sociology and History of Technology*. Cambridge: Massachusetts Institute of Technology Press.

Bishop CM (1995) *Neural Network for Pattern Recognition*. Oxford: Oxford University Press.

Bishop CM (2006) *Pattern Recognitions and Machine Learning*. New York: Springer.

Burrell J (2016) How the machine 'thinks': Understanding opacity in machine learning algorithms. *Big Data & Society* 1–12.

Canguilhem G (1943) *Essai sur quelques problèmes concernant le normal et le pathologique*. For the English translation, see Canguilhem G (1991) *The Normal and the Pathological*. Translated by Fawcett CR in collaboration with Cohen RS, New York: Zone Books.

Cardon D, Cointet J and Mazières A (2018) La revanche des neurones: L'invention des machines inductives et la controverse de l'intelligence artificielle. *Réseaux* 211(5): 173–220.

Crevier D (1993) *AI: The Tumultuous History Of The Search For Artificial Intelligence*. New York: Basic Books.

Collins H (1990) *Artificial Experts: Social Knowledge and Intelligent Machines*. Cambridge: Massachusetts University Press.

Dewey J (1939) *Theory of Valuation*. International Encyclopedia of Unified Science (ed. Neurath O) 2(4). Chicago, IL: Chicago University Press.

Diakopoulos N (2013) *Algorithmic Accountability Reporting: On the Investigation of Black Boxes*. New York: Tow Center for Digital Journalism, Columbia University.

Edwards P (1996) *The Close World: Computers and the Politics of Discourse in Cold War America*. Cambridge: Massachusetts Institute of Technology Press.

Fox-Keller E (2008) Organisms, machines, and thunderstorms: A history of self-organization part one. *Historical Studies of Natural Science* 38(1): 45–75.

Fox-Keller E (2009) Organisms, machines, and thunderstorms: A history of self-organization part two. *Historical Studies of Natural Science* 39(1): 1–31.

Grosman J (2018) Ethical and social issues, "Technical Report in the frame Pervasive and User Focused Biometrics Border Project" (PROTECT).

Hacking I (1983) *Representing and Intervening: Introductory Topics in the Philosophy of Natural Science*. Cambridge: Cambridge University Press.

Hacking I (1986) Making up people. In: Heller TC and Brooke-Rose C (eds) *Reconstructing Individualism: Autonomy, Individuality, and the Self in Western Thought*. Palo Alto, CA: Stanford University Press, pp. 222–236.

Hacking I (1999) *The Social Construction of What: Introductory Topics in the Philosophy of Natural Science*. Cambridge, MA: Harvard University Press.

Heims S (1991) *The Cybernetics Group*. Cambridge: Massachusetts Institute of Technology Press.

Hill RK (2016) What an algorithm is? *Philosophy and Technology* 29(1): 36–59.

Hinton G (2012) Neural networks and machine learning. *Coursera*.

Jaton F (2017) We get the algorithms of our ground truths: Designing referential databases in digital image processing. *Social Studies of Science* 46(7): 811–840.

Kay L (2001) From logical neurons to poetic embodiments of mind: Warren S. McCulloch Project in Neuroscience. *Science in Context* 14(14): 591–614.

Leroi-Gourhan A (1945) *Milieu et Techniques*. Paris: Albin Michel.

Leroi-Gourhan A (1965) *Le Geste et la Parole II: La Mémoire et les Rythmes*. Paris: Albin Michel.

McCorduck P (1983) *Machines Who Think: A Personal Inquiry into the History and Prospects of Artificial Intelligence*. Natick, MA: A. K. Peters Ltd.

Mackenzie A (2015) The production of prediction: What does machine learning want? *European Journal of Cultural Studies* 18(4–5): 429–445.

Mackenzie A (2018) *Machine Learners: Archaeology of a Data Practice*. Cambridge: Massachusetts University Press.

McKenzie D and Wajcman J (1985) *The Social Shaping of Technology*. Buckingham: Open University Press.

Malabou C (2017) *Métamorphoses de L'intelligence: Que Faire de Leur Cerveau Bleu?* Paris: Presses Universitaires de France.

Mitchell MT (1997) *Machine Learning*. New York: McGraw-Hill Education, 1995.

Olazaran M. A sociological history of the neural network controversy. *Advances in Computer Science* 37: 335–425.

Ng A (2011) Introduction to machine learning. *Coursera*..

Pickering A (1995) *The Mangle of Practice: Time, Agency and Science*. Chicago, IL: Chicago University Press.

Pickering A (2010) *The Cybernetic Brain: Sketches of Another Future*. Chicago, IL: Chicago University Press, 2010.

Polanyi M (1966) *The Tacit Dimension*. Chicago, IL: The University of Chicago Press.

Porter T (1996) *Trust in Numbers: The Pursuit of Objectivity in Public Life*. Princeton, NJ: Princeton University Press.

Putnam H (2002) *The Collapse of the Fact/Value Dichotomy and Other Essays*. Cambridge, MA: Harvard University Press.

Ricci F, Rokach L, Shapira B, et al. (2011) *Recommender Systems Handbook*. New York: Springer.

Rosenblatt F (1962) *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. New York: Cornell Aeronautical Laboratory.

Seaver N (2018) Captivating algorithms: Recommender systems as traps. *Journal of Material Culture* 1–16.

Simondon G (1958) *Du Mode D'existence des Objets Techniques*. Paris: Editions Aubier-Montaigne [1989].

Simondon G(1965–1966) *Imagination et Invention (1965–1966)*. Chatou: Editions de la Transparence [2008].

Turing A (1948) Intelligent machinery. In: Copeland J (ed.) *The Essential Turing*. Oxford: Oxford University Press [2004], pp. 395–432.

Vygotsky L (1978) *Mind in Society*. Cambridge, MA: Harvard University Press.

Ziewitz M (ed.) (2016) *Science, Technology and Human Value*, special issue: Governing Algorithms, 41(1).