

Computational Environments as Administrative Procedures as Instruments of Political Control

Colin McCubbins¹
Google Inc.

Abstract

Artificial Intelligence and Machine Learning algorithms are increasingly being used as key decision-makers in government processes due to their speed and ability to process large volumes of data. As with any human actors, deployed algorithms are subject to oversight programs. However, oversight and accountability of these algorithms and the decisions they make are limited due to well known technical factors stemming from their inherent complexity. There have been many proposed solutions to this issue, from calls for algorithmic transparency to design factors from computer science that regulate software deployment to advances in the field of explainable machine learning. Yet, any oversight programs designed to regulate algorithmic decision-making are costly and have high expertise bars for successful implementation.

I argue that these oversight solutions are incomplete. Institutions regulating human actors have design features that manage incentives and intentionality to reduce shirking. Because algorithms do not behave as human actors do, do not have incentives, consider different information, and make decisions differently, more consideration must be paid to the *environment* in which algorithms operate – just as human actors require human-design edifices to manage their intentionality, so do algorithmic actors require software-design edifices to manage their computation. Therefore, I propose a solution based on system-wide infrastructure design that manages not just the algorithm in question, but also the software functions that control the data, computation, and deployment procedures around it. This infrastructure is designed to reduce bespoke-ness and complexity while providing centralized functions for regulation of software artifacts, data, and policy management.

1. Introduction

The core of representative democracy is the mechanisms by which policy decisions are responsive to interests or preferences of citizens. Decision-making authority is modeled as a principal-agent problem, where this authority is delegated from policymakers, who represent the citizenry, to agencies who represent the policymakers. Each delegator in this daisy-chain suffers from informational and time deficits that impede their oversight ability of their delegates. Policymakers have to overcome these deficits to ensure that their intentions will be carried out by agents that have intentions of their own, often conflicting.

¹ Senior Privacy Engineer, Google Inc. I am grateful for helpful advice, incisive comments, and technical design feedback from Chris Den Hartog, Daniel Rodriguez, Steve Sullivan, Abbey Keck, and Vipul Nataraj. With apologies to my father, Mathew D. McCubbins, Roger Noll, and Barry Weingast for borrowing the title of their seminal work.

Now, powerful computation, advancement in algorithmic sophistication, and widespread data collection and usage has created environments where computers are responsible for making decisions that individuals would have made historically. They are a natural extension of “expert systems” that rely on logical rules to standardize and automate decision-making – use of algorithms to automate key agency decisions can theoretically provide transparency, auditability, and repeatability while reducing outcome variability due to their programmatic nature and source-code declaration. They are also high infinitely scalable as computational systems become cheaper and data collection broadens.

Yet, this means that decision-making authority is being increasingly granted to non-human entities. Given the efficiency and scalability of these systems, it is likely that these agency programs will expand in the future, meaning any oversight of administrative entities should take the presence of these algorithmic systems into account, either as extensions of existing agents or as agents themselves.

Decisions made by algorithms also create questions of accountability. Human actors have intentionality and much of the regulatory environment, from sanctions to notice-and-comment rulemaking, deal specifically with regulating potential ill-intentions by forcing individuals to disclose reasons for taking actions, either ex-ante or ex-post. However, the entire edifice of transparency and accountability that has been constructed around human agents is eroded when applied to unintentioned decision-makers. Machines cannot be investigated through traditional regulatory mechanisms of affording transparency and accountability as they cannot give reasons for their decision-making, nor can humans do so on their behalf. Instead they must be investigated through different means.

I argue that, just as an agency environment has been created to manage human intentionality in the policy process, more focus needs to be paid to the *designed environment* in which delegated decision-making algorithms are deployed. Algorithms are rarely deployed on their own; rather, they are part of computational networks which are themselves dependent on systems that govern scheduling, machine provisioning, data ingestion and storage, and data cleaning among other tasks. These system architectures must be standardized if any regulatory effort is going to be undertaken. This affords greater control over the entire process of algorithmic decision-making that is broader than viewing the algorithm in a vacuum. These design choices can focus on translating a priori goals of policymakers into code containers the agencies can use as scaffolding to surround deployment of decision-making algorithms, reducing monitoring costs and ex-ante reducing the potential range of algorithmic slack.

As is the case with human-only administrative law, it is to the advantage of policymakers that infrastructure is less bespoke and more regularized to reduce outcome variability and help diagnose issues. Just as traditional administrative procedures are control mechanisms in that they limit the space in which human agents can operate, so too can controlled computational infrastructure limit variability and opaqueness from algorithmic output. This has been somewhat discussed as a potential solution in the literature, although the problem space has been relegated mostly to governing specific algorithmic inputs or the source code of algorithms themselves. I believe this should go one step deeper into regularizing and, literally, codifying code deployment rules that regulate computational standards, deployments, and data usage.

2. Oversight and Algorithms

2.1 A Brief Overview of Oversight

Oversight programs tend toward two primary mechanisms of controlling how policies are turned into rules by agencies: monitoring programs and procedures. These mechanisms are designed to hold agencies accountable for potential actions they might take when executing policy mandates. Together, they represent the environment in which agencies operate.

The core of the principal-agent relationship, and thus the regulatory environment that is constructed to manage it, is *accountability*. Accountability is institutionally rendered by regulating *intentionality*. Policymakers fear that the agents they delegate to have different beliefs and desires than they do and, thus, will act wilfully as they inherit the responsibilities of deploying policy. Transparency standards in regulatory environments lay bare this intentionality and hold agents accountable for acting in accordance with their own hidden intentions and not that of their delegators.

Monitoring programs act on potential ill-intentions by creating visible punitive mechanisms by which rogue agents can be held accountable. Monitoring programs take two forms, called “police patrols” and “fire alarms” (McCubbins and Schwartz 1984). “Police patrols” programs represent ongoing oversight and evaluation by policymaking arms to ensure adherence to general legislative intent. These programs, by their nature, are costly to implement as they require manpower with sufficient expertise and time allocation to effectively monitor implementing agencies for potential slack. “Fire alarms” programs are more reactive and take the form of avenues by which affected citizens can levy complaints against agencies that harm them.

Monitoring programs act ex-post, after an agency ruling has been rendered. Further, while the exact costs vary, monitoring efforts are costly, as they require some degree of analysis for both the assessment of the degree of noncompliance as well as the appropriate punishment. Similarly, monitoring can only affect what it sees, which renders it typically imperfect, and tends to overfocus on outcomes rather than symptoms. Sanctions and other punitive measures taken as the result of monitoring scenarios tend to be larger in order to affect compliance.

Procedural rules, on the other hand, can be an ex-ante mechanism for inducing agency compliance by structuring the informational environment surrounding an agency’s decision-making process (McCubbins 1985, McNollgast 1987). This process creates accountability through reason-giving: The Administrative Procedures Act codifies the rulemaking process by providing the mechanisms by which information is gathered and arranged by agencies in a manner that would hold up under federal scrutiny. Indeed, when invoked, the APA requires agencies to “explain” their decision-making in court. This process can be “deck-stacked” to favor certain interests over others, in essence driving outcomes and benefits tailored to those favored by the policymaker. Similarly, because agency procedures are established to be responsive to these interests, they can be put on “autopilot” by evolving with the constituencies they serve.

This, McNollgast argues, allows policymakers to drive favorable outcomes without setting up costly monitoring programs by steering agents down preferred paths – these solutions are, in a sense, “engineered” in that they induce consistency, efficiency, and accountability by putting limits on and enforcing visibility into agency intentionality. This logic has been extended by many (McNollGast 1989,

Epstein and O'Halloran 1994 and 1999, Ting 2001, Gailmard 2002, Huber and Shipan 2002) but this core remains the same.

2.2 Oversight of Algorithms – Police Patrols and Fire Alarms Revisited

Artificial Intelligence and Machine Learning Algorithms (AI/ML) are increasingly being used as key decision-makers in government processes due to their speed and ability to process large volumes of data. By 2020, approximately half of all government agencies reported use of these algorithms in the administrative process (Engstrom et al 2020), from adjudicating the distribution of benefits to public risk evaluation. Many of these algorithms even aid decision-making in “high stakes” situations, such as predicting recidivism rates for parole programs, where individuals could be subject to life-altering judgements. As decision-makers, then, it follows that algorithms should be held accountable for decisions rendered just as any other actor would.

It would seem appropriate, then, to have some oversight mechanisms in place to help regulate machine-led decision-making. Here, already, the regulatory apparatus begins to break down. Half of the equation, the traditional oversight mechanisms of monitoring agents, combined with systems of rewards and punishments, do not work on algorithms, which do not respond to incentives in the manner that humans do. As an extension, while the humans who create these algorithms *can* respond to incentives, it is unclear whether there are sufficient technical mechanisms for enforcing compliance, as algorithmic decision-making is difficult to understand in many circumstances even for the engineers and scientists that construct them. This point will be explored in the following section.

Thus, instead, the focus is on mechanisms that grant the ability to *monitor* decision-making. Decision-making accountability has been the main concern of the literature thus far (Citron 2008, Kroll et al 2017, Diakopoulos, 2016, Coglianese and Lehr 2016, Annany and Crawford 2018, Engstrom and Ho 2020, Engstrom et al 2020) as it is fundamental to the existing regulatory structure. At the core of this accountability is *transparency*: how can a decision made by an algorithm be explained to overseers as well as the constituency it affects?

The core of hearings, judicial reviews, and other ex-post monitoring mechanisms is “reason giving”. That is, agents have to disclose the reasons for certain decisions and use of certain information. This, when combined with incentive manipulation in the form of sanctions, hearings, and the like, helps bridge potential information asymmetries that may exist between principal and agent by limiting the willfulness of agents. Applying this to algorithms bears special consideration, however, to the process by which machines make decisions. To answer this, calls for transparency attempt to reconcile some measure of *decision provenance* for algorithms – what were the steps taken to arrive at a particular decision²?

So how is this done? Decision provenance has been widely discussed for human actors and is rooted in an examination of *intentionality*. It has been demonstrated in the cognitive science, philosophy, and increasingly in legal and political science literatures that humans impute intentions to nearly everything (Dennett 1987, Fauconnier and Turner 2002, Boudreau et al 2004), that these imputations concern metaphorical and not actual intent (Dennett 1987), and that interpretation in the legislative and regulatory environment is predicated on ability to decipher communication through various channels (Boudreau et al

² Provenance has a specific meaning in computation – one should be able to determine the chronology of ownership, changes, location, etc. of any object in the system. Singh et al. (2019) define *decision provenance* as using “provenance methods” in computer science to help expose how data is processed in the system as a whole and was used to take a specific action.

2007). These principles guide the framework by which the regulatory environment is maintained, from interpreting legislative intent of statutes to overseeing agency implementation of those statutes.

There have been several arguments about how to amend current rulemaking procedures to fit with increased use of AI/ML. The most commonly suggested of these procedures attempts to analyze algorithmic intent via some mechanism of public disclosure, such as disclosing source code or notifying individuals of the sources and types of data used in decision-making (Citron and Pasquale 2014). Others have argued that this could be done by amending the APA's notice and comment procedure and reviewability criteria, creating oversight boards, and benchmarking decisions against human actors (Engstrom and Ho 2020).

These types of amendments are likely necessary to achieve necessary legal standards and set up appropriate fire alarms among the constituency. However, there are limitations to each, rooted in the fact that the nature of algorithmic decision-making is often inscrutable to humans. Disclosing the inner workings of an algorithm or the data used, even under expert scrutiny, does not necessarily mean that a satisfactory explanation to an outcome will be given nor that the architects will be able to substantially change the outcomes in all appropriate dimensions. In those cases, this will likely lead to an up-or-down decision on AI/ML usage rather than force a substantial improvement to the system itself and any fire alarm implemented is rendered half-useless at achieving its goal. To perform a more thorough ad-hoc systematic analysis of the entire decision-making pipeline would be costly, both in terms of expert necessity as well as time to review, and still may not provide desired remediations if the algorithm is sufficiently complex or the data used is highly dimensional.

2.3 Unpacking Algorithmic Decision-Making

So how do machines make decisions? Answering this question, it is argued, impacts broader democratic and legal ideals, such as the legitimacy or fairness of decisions that are made (Citron and Pasquale 2014, Calo and Citron 2021). It is clear that machines do not respond to manipulation of incentives in the same way that humans do, so already we are limited in the technical regulatory mechanisms we have. Thus, most of the proposed regulatory programs in this space are focused on full descriptions of the decision-making process. Here, a similar set of standards as applied to human actors is also applied here – discovering the basis for machine decision making is analogous to the notice and comment rulemaking that forces agencies to explain their decision making process. That is, just as the APA sets a mechanism for mapping the decision provenance of an agency, so too would transparency standards be used to account for similar circumstances with algorithmic actors.

The heart of the issue, however, remains:

Adjudication versus Computation

Transparency standards in regulatory environments typically adhere to ideals of intentionality, that the rulemakers should lay bare the reasons for their adjudications. However, the work that algorithms perform is not *adjudication*, it is *computation*. Adjudication implies some level of subjectivity – that the decisions made are rooted in this conceit of individual intentionality. Computation is objective by nature – the decisions are rooted in math³.

³ Understanding the resultant computation can also require some level of modeling proficiency to interpret and design features outside of just the machine learning model itself, as the outputs and explanations are often couched in math and/or are specific to the datasets being evaluated. This is true even using interpretable ML methodologies, where outputs are designed for computer science researchers and not laypeople (see note 2.3, Du et al 2019).

This point is often implicitly determined but never formally explicated when discussing algorithmic regulation. Attempts at machine accountability typically take the form of cramming the machine decision into a human regulated context and forcing these human overseers to impute intentions to the algorithm, which are likely to be incorrect as will be discussed below.

Attempts to rectify this core issue have focused on making algorithmic environments “human readable” such that overseers can apply their own view of the system’s intentionality and ensure decision rules used by algorithms are in line with human expectations. This is not an easy standard to achieve. The limitations in deriving algorithmic transparency and accountability are well known in the literatures:

Algorithms as Black Boxes

The idea that many AI/ML algorithms are black boxes is nigh canonical in the literature.

For most real-world applications, use of data tends towards excessive, as data is “unreasonably effective” at achieving quick model gains due to the potential information present in new data points (Halevy et al 2009). There is a corresponding exponential increase in the volume of data the more dimensions are added to it, which increases the available information for model training. Broad data availability has made increasing dataset dimensionality much cheaper than before, which provides an easy mechanism for increasing the amount of information available to a model. If the goal of the AI/ML is predictive accuracy, it is arguable that dimensionality is one of the biggest boons for this goal (Halevy et al 2009). It is also the case that finding simple models that fare as well as complex ones in terms of accuracy is an NP-hard problem, meaning that use of AI/ML tends towards complexity over simplicity in practice (Semenova et al 2019).

Unless the dataset is convex, which is unlikely in any real-world scenario without severe parameterization, the gradient descent algorithm at the core of the AI/ML system is likely to fall into a non-global minima, of which there are likely many in a super highly dimensional dataset. These local minima likely contain some random distribution of weights of the composition dimensions and tells the operator little about the actual component information used to get there. This creates unpredictable and unascertainable behavior for any of the developers or operators. As such, a typical model has a multitude of realizations, each of which settles in a different local minimum, each having roughly equal error evaluation, but wherein slight perturbations can cause the model to evaluate differently (Breiman 2001).

It is difficult if not impossible to derive any “reasoning” for the algorithm’s placement in this local minima as the result of many small contributions of a large number of data points existing across many dimensions. This is true not just for any observer but also the creator or implementor of the algorithm itself. This problem is exacerbated if there are high levels of recursion present, such as is the case with Neural Network implementations. Thus, it is often the case that a derived “reason” for an algorithm’s outcome are, themselves, model approximations of the black box model or lack sufficient information to provide a holistic explanation (Rudin 2019). While this problem is most acute in complex neural network architectures, even shallower models, such as regressions, classification trees, or support vector machines, can be difficult to interpret given

large numbers of dimensions (Bathae 2017), even when using techniques designed for model interpretability (Du et al 2019, Rudin et al 2021)⁴.

As an extension, moving the onus of decision explanation to the individuals within agencies who would be relying on the algorithm is also rendered ineffective since they will not be any better at rendering explanatory power. Monitoring the human actors, thus, is an incomplete solution.

It has been argued that more formal transparency measures, such as public source code and input/output disclosure (Citron 2008) and platform specific auditing procedures (Sandving et al 2014) can provide a measure of accountability for algorithmic actors by reducing potential information asymmetries⁵. While this does not give a full accounting of the decision-making process, it is assumed that it can be inferred from the system components. Bridging these informational disadvantages is made difficult by the fluency, bespokeness, and expertise barriers present when analyzing these systems and likely requires additional hiring and institutional restructuring to ensure that oversight bodies are properly staffed. However, there are additional issues that this type of oversight presents:

Coding Language Fluency and System Bespokeness

Any individual monitoring one of these systems must be able to read the code to judge what the system is doing. There are numerous coding languages that could be used, such as Java or Python, each with its own syntax and methodology, presenting an expertise barrier. Language fluency may not be sufficient if the language used is more expressive, the code uncommented, or sufficiently complex.

Similarly, algorithms are never deployed on their own but are instead part of broader software systems. High levels of modularization between components can add to the existing complexity, especially if modules are not built from commonly sourced and widely understood binaries. This problem is exacerbated if components are designed to work in concert with other components as part of a broader, dependent infrastructure that shares data across multiple systems⁶. All of these issues can lead to some level of bespoke irregularity in the source code disclosure, making analysis difficult even for software experts.

Monitoring through more technical means attempts to rectify the above issue while maintaining an appropriate standard of accountability. Use of methods such as software verification⁷, cryptographic commitments, and zero-knowledge proofs⁸, have been proposed as an alternative to pure oversight

⁴ There are other accountability issues related to data usage that are more specific to narrow bands of accountability – for example, there is a defined link between information entropy and privacy wherein a privacy preserving system may want to reduce both cross and within dimension entropy – that will not be discussed in detail and left as a thought exercise for the reader.

⁵ There are also nascent solutions in the explainable AI/ML literature, such as “local interpretable model-agnostic explanations (LIME)”, which train other, simpler machines (that can be presumably be explained) around the locality of a decision made by a deployed AI/ML algorithm (Ribiero et al 2016), although these have yet to be seriously discussed in the legal literature.

⁶ As an example, regulating targeted advertising through legal means cannot be done through regulating an individual piece of code, as the entire ecosystem is designed to have common components, such as cookies and device identifiers, that create complicated dependencies in algorithmic input and output. Thus, changes to inputs in an algorithm can have unpredictable changes to the types of ads shown (Datta et al 2015) with very little reasoning as to why.

⁷ Kroll et al 2015 discuss only “formal” software verification, which relies on three main components: 1) a mathematical description of what the designer wants to achieve 2) a piece of verification software that is written to test for this mathematical description and 3) a proof that demonstrates the code satisfies the properties in 1). Formal verification methods are complex in deployment and require the user to not only understand in detail why a system behaves in a certain way but also how to express this logic to the verification system.

⁸ All of the mechanisms discussed in Kroll et al 2015 attempt to account for the presence of “invariant” within a software system – i.e. that specific decision-rules were used or that specific data was used to render the decision. These mechanisms would also

programs (Kroll et al 2015, Desai and Kroll 2017) and are focused further up the development pipeline, in the design specifics of a piece of software. These, in themselves, are oversight mechanisms that attempt to induce “procedural regularity” in the algorithm’s deployment by setting off fire-alarms when some pre-existing commitment is not followed. This reduces monitoring costs by maintaining certain “proofs” through cryptographic signatures and provides important benefits of full source-code disclosure, such as presence of appropriate decision-rules and data, without the difficulty of parsing and interpretation.

These technical policing mechanisms suffer from many of the same limitations as many pure transparency doctrines – that is, the operable components of many algorithms are actually a *pipeline and not just an algorithm*; they are a combination of the algorithm, the data elements being used, and the systems that process them:

Data Agnosticism

While AI/ML systems are not *structurally agnostic* – systems must be able to ingest the data structure they are to classify – they are *data agnostic* in that any data from a wide range of distributions can be used and classified confidently. This is true even if the data inputs are nonsensical or deviate in imperceptible ways. For example, neural networks trained for image recognition make confident classifications even in the absence of salient features that render images uninterpretable (Carter et al 2021). Similarly, use of data that are only very slightly different from one another can lead to systematic misclassifications of images that are otherwise unnoticeable to humans (Szegedy et al 2014, Goodfellow et al 2015) or by replacing words in NLP models with legitimate alternatives that alter semantic outputs (Liang et al 2017, Samanta and Mehta 2017).

Preprocessing of data and its impacts on AI/ML systems is understudied and is usually seen as an accessory exercise to the actual endeavor of machine learning. Yet, the above principle puts more pressure on system-wide verification of not just the computational process the algorithm is undergoing. Rather, the nuances in data usage, aggregation, cleaning, transportation, model training, and other associated activities all have an impact on how the algorithm renders decisions. Any accountability measures, then, cannot be taken at the algorithm decision level but rather is an investigative exercise in tracing system-wide data flow and processing.

Consequently, the presence of adversarial learning attacks puts an increased onus on having strong data security protections and suggests that the entire data pipeline is necessary to regulate. There are several well known attacks, such as the Fast Gradient Sign Method attack, that can ensure misclassification. These types of attacks can impede decision analysis of AI/ML systems, as diagnosis can be difficult just by viewing data inputs and outputs given.

Further, this principle makes human-led review of the decision-making process extremely costly. Without appropriate and standardized documentation, it may be necessary to reconstruct the entire data and model pipeline in order to accurately audit and remediate potential issues. Similarly, improvements to algorithms to account for marginal cases usually takes the form of adding additional classification labels to training data, further increasing the dimensionality of the dataset. Rebalancing algorithms in this way may have unintended consequences on future classifications, especially when small variations in the data

create audit trails to trace system accountability once a decision had been rendered, enabling more accurate ex-post oversight. These methods are all reliant on cryptographic communication to code reviewers and deployers that any number of specific policies were followed (i.e. data would not be pulled from a specific location) – the cryptography provides some level of “proof” or immutability to the results.

matter, and with the need to balance accuracy, extensibility, and future interpretability. This will likely lead to circular oversight and improvement loops.

Lastly, most overviews of algorithmic systems treat the deployment as a comparative static, wherein an algorithm is deployed and runs in a vacuum. In real-world scenarios, algorithms and the populations they affect change constantly. There is almost constant pressure to improve, either in modeling or in data usage and retraining. The software systems housing algorithms are also subject to change as modules that affect data collection, transfer, or processing are rewritten and redeployed. This can lead to significant model, data, and configuration drift over time, impeding any active oversight efforts.

2.4 Conclusion – Towards Different Procedures

The difficulty in unpacking algorithmic decisions and the complexity of surrounding environments renders the entire enterprise of monitoring costly. Mechanisms to reduce cost, such as source code transparency, provide additional impositions that seem to merely defer cost to experts rather than reduce it outright. Indeed, the endeavor of understanding may be impossible, especially for AI/ML systems with high levels of recursion or highly dimensional data.

I argue that monitoring algorithms through transparency doctrines is likely to be untenable given the discussed technical issues – it is either impossible, too costly to implement, or both. In fact, I believe that transparency as defined should not even be the most important goal of the oversight program. Instead, broader focus should be paid to establishing the environment in which algorithms operate. This prospect involves careful design consideration.

System-wide design as a solution to algorithmic oversight is not new to the literature on algorithmic accountability. These solutions harken to the concept of “internal administrative law”, of which the APA is a part, that guides the administrative process through controlling the information procurement and dissemination and limiting agent discretion. Singh et al (2019) discuss and define a specific version of machine decision provenance and argue for better data lineage systems that can more easily trace data usage. Kaminski (2019) and Engstrom and Ho (2020) mention, briefly but specifically, “accountability by design” as a concept in their overview of potential solutions to algorithmic governance, wherein accountability is embedded into technical systems through construction of controls. Annay and Crawford (2016) suggest that greater focus should be placed on the manner in which humans and machines interact as part of the accountability structure. None of these design considerations, however, are specific.

Given the aforementioned difficulty in oversight, I argue that these types of design considerations are the only way to successfully ensure any form of regulatory structure. Any centralized capacity to induce accountability *must start with environmental design* – just as human actors require human-design edifices to manage their intentionality, so do algorithmic actors require software-design edifices to manage their computation. These design considerations are also the best way to take advantage of the uniqueness of software systems; they are codified, logical, produce standard outputs given the same inputs with minimal variance, and there are a wide variety of tools that are already used to ensure these characteristics.

Similarly, where I differ from much of the existing literature is that “accountability by design” has to be *systematic* and not self-contained. Admittedly, “accountability by design” is a riff on the term “privacy by design”, wherein privacy is embedded into the design choices of components of software systems. Privacy, however, fails when individual system components are compromised – if one piece of a system exposes private data, it does not matter whether the rest of the system's components are perfectly

private, the entire system loses its status. So, too, is it in “accountability by design” – a loss of accountability in one system makes true auditability and oversight impossible.

To draw an analog to a well-understood system, the APA is successful in inducing political control in part because it regulates the entire administrative environment and in part because it is a flexible system, wherein Congress does not have to provide piecemeal changes to each agency’s regulatory mandate (McNollGast 1987). Containing design considerations to a single piece of software or even a single system deployment from an agency does not regulate the entire software production environment and would further lead to inflexible piecemeal auditing, since each deployment would require a new design, a new formal verification process, and new cryptographic guarantees, among other considerations. Therefore, the design focus must *not* be placed on software itself but on *broader infrastructure that deploys software* in order to induce control and accountability – infrastructure as administrative procedures as instruments of political control, so to speak⁹.

To extend this logic, I argue that there are very specific design components necessary to provide a baseline for accountability in an algorithmic system. In the next section, I explicate this design, stating the goals of the infrastructure. I then provide an overview of a pseudo-architecture and discuss the component systems. Lastly, I discuss the role of the broader administration in defining goals and policies, translating them into code, and managing the overall environment.

3. Infrastructure as Administrative Procedures as Instruments of Political Control

This section explicates the features of a designed regulatory environment for algorithmic actors. Section 3.1 describes the goals of the overall architecture while 3.2 discusses the specific components and diagrams how they relate to each other in a pseudo-architecture diagram. A discussion of the components in specificity follows.

It is worth noting that this proposed infrastructure and the agency that would be required to deploy it would be expensive and time consuming to create and maintain. In the absence of being able to deploy the system as a whole, some of the components could likely be investigated on their own, such as the presence of standard build systems for software, centralized data storage, or use of versioning systems. However, because of the critical roles of each of the environment components in ensuring accountability, we must take *all* working systems into consideration *together*. A critical failure in any one system can lead to a loss in environmental integrity due to the dependency relationships. Therefore, it is worth considering the environment as a whole before either extending it or making concessions for cost.

Likewise, while the system proposed here is a pseudo-system in that the design is high-level and lacking in specificity, I believe this is an appropriate starting point for discussing how new administrative environments can be maintained.

3.1 Infrastructure Goals

Standardization. The primary goal of this architecture is to provide mechanisms to construct an environment where used components are known, standardized, and verified to increase control over

⁹ Thanks for letting me borrow this, Dad.

software and data. Reducing bespoke-ness aids achieving control, as standard components can be verified and understood before use, reducing ad-hoc oversight complexity.

Create a Process for Centralized Authorization and Approvals. Software artifacts, data stores and usage, and infrastructure improvements should be verified and approved before usage. This verification system has to be centrally managed by a governing agency to ensure appropriate oversight before deployment, when algorithms might become sticky in usage. These approved components are centrally managed by a governing body and exposed through the regulatory environment to encourage reuse and reduce bespoke-ness wherever possible.

Establish Provenance Systems. Provenance systems, both for software artifacts as well as data, are necessary in determining which systems or parts of systems are responsible for decision failures. Computational systems are often made with large numbers of component parts and can draw data of different types from numerous sources. This is especially true for complex multi-part systems transacting large volumes of data where complexity begets opaqueness and oversight is costly or impossible. Importantly for oversight, the context of the transformations done on a particular dataset are lost as soon as that data moves either between software components or, worse, between administrative entities. As much context should be maintained during the movement of data in order to ensure proper oversight capabilities.

Have Strong Versioning and Reproducibility. Data outputs, classifications, and other results must be reproducible through any third-party verification procedures, such as review by an oversight board. Thus, software versioning systems must be employed and copies of datasets that are used in production must be stored as well to ensure that decision-pipelines can be recreated when necessary to aid investigation, especially since small changes to data inputs or algorithms can lead to differences in algorithmic output.

Use “As Code” Systems Whenever Available. Systems should be managed through code rather than manually configured wherever applicable. Coding approaches induce regularity in design and deployment and increase visibility into decision components by exposing system-wide rules. Additionally, code can be verified through the centralized authorization and approvals process, meaning that rules can be decided on and enforced systematically.

3.2 Pseudo-Architecture

The pseudo-architecture proposed here is meant as a starting point to discuss how software systems could be designed to mitigate issues with algorithmic decision making. While this proposed architecture is intended to be a realistic exploration of a potential system design, architectural and design components often need to meet specific considerations of the implementing parties. In any case, it would be worthwhile for any governing body to have a machine translatable description of its goals and values to ensure that the designed system meets necessary standards for software deployment and data usage.

Figure 1 shows a suggested high level architecture¹⁰. This architecture is intended to create an end-to-end deployment system for any AI/ML algorithms used in government systems, in line with the goals specified in Section 3.1.

¹⁰ All of the subcomponents in this architecture require additional engineering design with specifics to the desired policy mechanisms they oversee.

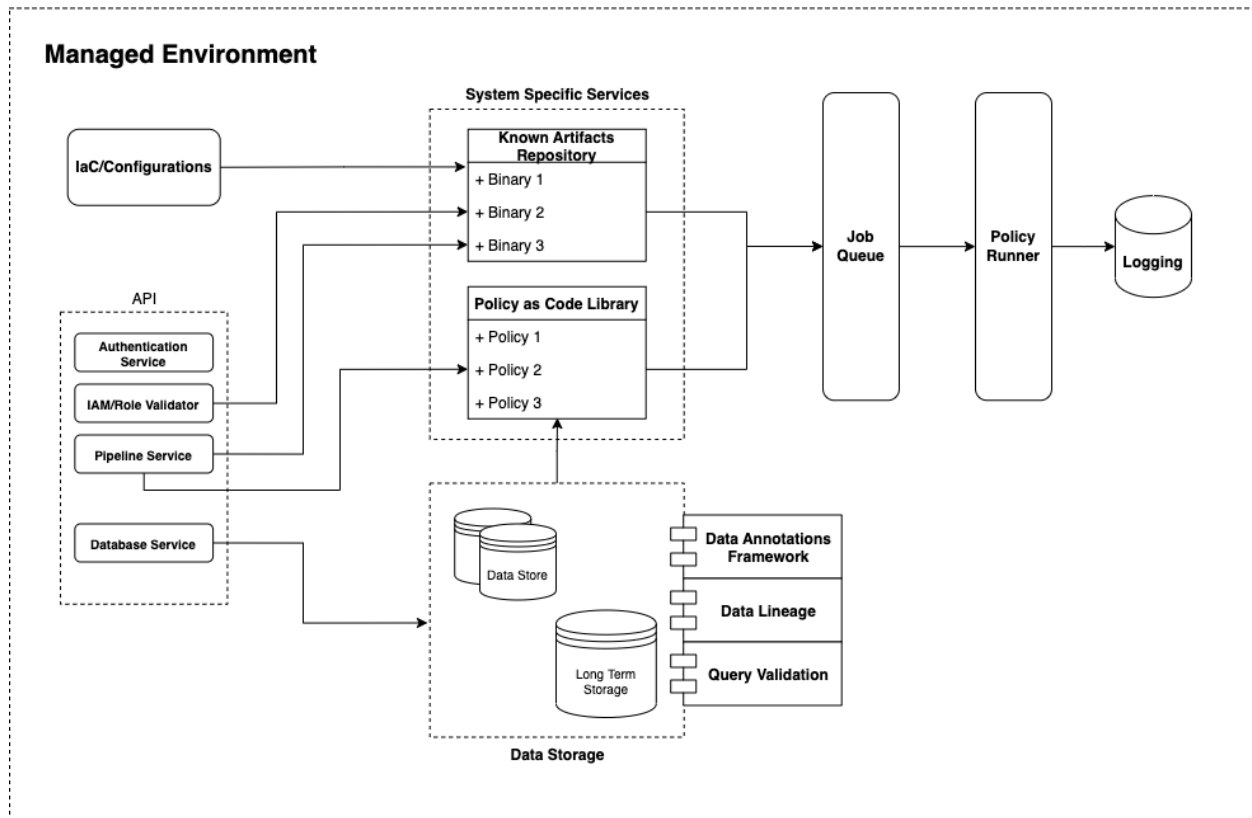


Figure 1 - Architecture Diagram for Sample Regulatory Infrastructure

3.2.1 Architectural Components

In the above environment, agencies seeking to develop and use AI/ML models to make decisions are required to use the environment to manage deployment of their entire pipeline. The managed environment in **Figure 1**, above has several core components.

Configurations. Configurations should be managed in machine-readable code rather than manually configured. Broadly, this is a concept known as “Infrastructure as Code” (IaC). In keeping with the primary goal of the overall architecture, IaC induces standardization in computational resources dedicated to processing data, such as networks and servers, as well as environment details like operating systems or installed software from trusted sources.

Known Artifact Repository. Repository of vetted and verified software artifacts, such as:

- Configuration information for IaC deployments.
- AI/ML pipeline subcomponents such as data extraction, cleaning, and transportation services, models, testing and monitoring functions, and model hosting services.
- Identity and access management (IAM) configurations
- Standard database queries or data extraction scripts.
- Other general development scripts.

Repository artifacts have the following properties:

- They should be labeled with metadata so that policy readers can identify what modules are in use during a pipeline submission and what they are used for.
- Verification tags acknowledging that a particular artifact was reviewed before being committed. Software should be vetted and verified by the administering agency before being committed to the repository and further used in a model pipeline.

Using known artifacts can induce standardization in the development of an entire AI/ML pipeline. It also provides an oversight mechanism for the overseeing agency to ensure consistency in deployments by reducing them to known quantities. This, further, alleviates some security concerns in that important artifacts used in any AI/ML pipeline can be accounted for before use to prevent malicious code deployment.

Data Storage. There are three major components in the data storage system that allow for greater visibility and control into data usage:

- **Data annotations** - metadata about the data contents of a particular data store or extraction of data. I argue that this should at least include (1) source of the data, (2) data categories, and (3) sensitivity levels for security compliance. Data annotations are widely used data governance features in industry and their usage is recommended here to maintain machine-readable visibility on data usage.
- **Data Lineage** - systems to catalog the source of data and how it is updated over time. The lineage system should ensure that data annotations propagate automatically to new data stores when data is copied or aggregated.
- **Query Validation** - Parses database queries to ensure data extraction is compliant with known policies. The validated queries can be validated against stored policies in the policy library to ensure specific data components are not used in conjunction with one another.

Once data is used in an AI/ML pipeline, a copy of that dataset with associated metadata, such as when the job was run, should be sent to Long Term Storage so that the entire pipeline can be rerun.

Policy Library. Policy should be managed in machine-readable code, where applicable, rather than manually overseen. This library defines policies that regulate how software subcomponents behave. The primary use of the Policy Library will be to manage how data is used – policies will check the metadata and data annotations within AI/ML workflows and send the agglomerated metadata to the Policy Runner to fail jobs that don't meet the requirements for data usage. Policies can regulate:

- Annotation requirements on data to ensure that data usage is traceable, since data annotations must often be added to data manually.
- Use of specific AI/ML models
- Presence of unknown artifacts used for important purposes
- Interpretability goals - if interpretability is deemed a goal, complex model usage could require result-set summary statistics to be published alongside result-sets of more interpretable models for later tradeoff analysis.
- Joint presence of specific sensitive data elements that should not be used in the same model
- Security policies, such as authentication requirements and vulnerability scans

Maintaining a policy library can help enforce standardization in how models are used by preventing certain pipelines from executing. It also alleviates some security concerns by ensuring that proper security policies are attached to AI/ML jobs with sensitive data.

Policy Runner. The Policy Runner is a computerized agent that enforces specific behaviors on a job, such as sending requests for new data annotations or failing the job entirely if certain policy criteria are not met. Together, the Known-Artifact Repository and the Policy Library create a store of relevant metadata, including what software was used within the pipeline, whether that software is annotated

correctly, any relevant data tags, and any relevant policies that need to be executed that are attached to a specific AI/ML job. Since this store of metadata is machine-readable, it can be read and relevant policies executed by the Policy Runner.

The policy runner can send logs of job runs to ensure auditability of entire pipelines.

Semantic Versioning. This fulfills one of the primary goals of the architecture, which is to ensure auditability and reproducibility. Strong versioning systems store exact copies of code with dates of change to any artifact and associated differences. Versioning is typically done for code in every software system. I argue that this should also be done for limited time frames for datasets used in production as well. In this way, the entire pipeline, from software to data, can be reproduced to aid auditing efforts.

3.2 The Role of the Administration

I believe that this type of solution should likely be built from within the government to ensure proper controls around software construction, due to the issues of transparency and bespoke-ness discussed above. This capacity would be regulated by an overseeing agency in charge of managing known software artifacts, managing the data environment and storage capacity, developing common metadata frameworks for relevant annotations and policy-as-code implements, and developing deployment code so agencies can in turn develop regulated AI/ML pipelines.

Importantly the overseeing agency also has to regulate goals and other success criteria set by agencies seeking to use these models, since AI/ML usage is littered with tradeoffs, such as well known bias/variance and accuracy/interpretability tradeoffs. As mentioned previously, model usage tends towards complexity in practice, so model usage standards would also have to be developed if model complexity regulation is desired. This agency can also extend more traditional oversight mechanisms, such as through review boards or more intensive processes – given the environment, this overseeing agency would also be responsible for reconstructing pipeline deployments, re-running any developed outputs and analyzing and interpreting the results. Thus, the agency acts as both a general regulatory oversight arm in addition to its job as a software API developer and maintainer.

This is not without challenges. Engstrom and Ho (2020) note specifically that this type of solution would require the agency to have very specialized hires to manage this software environment. Competition for resources with private industry would likely require new salary brackets for specialized engineering, security, and data science hires. Translating policy into code is also difficult and often subjective, so the core of the environment described above would require debate, review, and standard building from the overseeing agency. Developing an environment like this would require a slow, evolving process and this would make it difficult to catch up to current AI/ML usage, which is moving much faster.

4. Conclusion

References

- Ananny, Mike and Kate Crawford, 2018. "Seeing Without Knowing: Limitations of the Transparency Ideal and Its Application to Algorithmic Accountability" *New Media and Society* 20:973-989.
- Bathae, Yavar, 2017. "The Artificial Intelligence Black Box and the Failure of Intent and Causation," *Harvard Journal of Law and Technology* 31: 889
- Boudreau, Cheryl, Mathew D. McCubbins, and Daniel B. Rodriguez, 2004. "Statutory Interpretation and the Intentional(ist) Stance." *Loyola of Los Angeles Law Review* 38: 2131-2146
- Boudreau, Cheryl, Arthur Lupia, Mathew D. McCubbins, and Daniel B. Rodriguez, 2007. "What Statutes Mean: Interpretive Lessons from Positive Theories of Communication and Legislation." *San Diego Law Review* 44:957-992
- Breiman, Leo, 2001. "Statistical Modeling: The Two Cultures," *Statistical Science* 16, no. 3: 199-231
- Calo, Ryan, and Danielle Keats Citron, 2020. "The Automated Administrative State: A Crisis of Legitimacy," *Emory Law Journal* 70: 797
- Carter, Brandon, Siddhartha Jain, Jonas W. Mueller, and David Gifford, 2021. "Overinterpretation Reveals Image Classification Model Pathologies," *arXiv preprint arXiv:1412.6572*
- Citron, Danielle Keats, 2007. "Technological due process." *Washington University Law Review* 85: 1249-1313
- Citron, Danielle Keats, and Frank Pasquale, 2014. "The Scored Society: Due Process for Automated Predictions," *Washington Law Review* 89: 1-33
- Coglianesi, Cary, and David Lehr, 2016. "Regulating by robot: Administrative decision making in the machine-learning era." *Georgia Law Journal* 105:1147-1223
- Dennett, Daniel C, 1987. *The Intentional Stance*. Cambridge: MIT Press.
- Desai, Deven R., and Joshua A. Kroll, 2017. "Trust but verify: A guide to algorithms and the law." *Harvard Journal of Law and Technology*. 31:1
- Diakopoulos, Nicholas, 2015. "Algorithmic accountability: Journalistic investigation of computational power structures." *Digital Journalism* 3: 398-415
- Du, Mengnan, Ninghao Liu, and Xia Hu, 2019. "Techniques for Interpretable Machine Learning." *Communications of the ACM* 63: 68-77
- Engstrom, David Freeman, and Daniel E. Ho, 2020. "Algorithmic Accountability in the Administrative State." *Yale Journal on Regulation* 37: 800-854
- Engstrom, David Freeman and Daniel E. Ho, 2021. "Artificially intelligent government: A review and

agenda." *Research Handbook on Big Data Law*.

Engstrom, David Freeman., Daniel E. Ho, Catherine M. Sharkey, and Mariano-Florentino Cuéllar, 2020. "Government by algorithm: Artificial intelligence in federal administrative agencies." *NYU School of Law, Public Law Research Paper* 20-54.

Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3551505

Epstein, David and Sharyn O'Halloran, 1994. "Administrative Procedures, Information, and Agency Discretion," *American Journal of Political Science* 697-722.

Epstein, David and Sharyn O'Halloran, 1999. *A Transaction Cost Politics Approach to Policy Making Under Separate Powers*. Cambridge: Cambridge University Press.

Fauconnier, Gilles, and Mark Turner, 2002. *The Way We Think: Conceptual Blending and the Mind's Hidden Complexities*. New York: Basic Books.

Gailmard, Sean. 2002. "Expertise, Subversion, and Bureaucratic Discretion." *Journal of Law, Economics, and Organization* 18:536–555.

Goodfellow, Ian J., Jonathon Shlens, and Christian Szegedy, 2015. "Explaining and Harnessing Adversarial Examples." *arXiv preprint arXiv:1412.6572*

Halevy, Alon, Peter Norvig, and Fernando Pereira, 2009. "The Unreasonable Effectiveness of Data." *IEEE*

Intelligent Systems 24, no. 2: 8-12

Huber, John D. and Charles R. Shipan. 2002. *Deliberate Discretion? The Institutional Foundations of Bureaucratic Autonomy*. New York, NY: Cambridge University Press.

Kaminski, Margot E. 2019. "Binary Governance: Lessons from the GDPR's Approach to Algorithmic Accountability," *Southern California Law Review* 92, No. 6:1529

Available: <https://ssrn.com/abstract=3351404>

Kroll, Joshua A., Joanna Huey, Solon Barocas, Edward W. Felten, Joel R. Reidenberg, David G. Robinson, and Harlan Yu, 2017. "Accountable Algorithms," *University of Pennsylvania Law Review* 165: 633-705

Liang, Bin, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi, 2017. "Deep Text Classification Can Be Fooled," *arXiv preprint arXiv:1704.08006*

McCubbins, Mathew D. 1985. "The Legislative Design of Regulatory Structure," *American Journal of Political Science* 29:721-48.

McCubbins, Mathew D. and Thomas Schwartz, 1984. "Congressional Oversight Overlooked: Police Patrols vs. Fire Alarms," *American Journal of Political Science* 28:165-79.

McCubbins, Mathew D., Roger G. Noll and Barry R. Weingast, 1987. "Administrative Procedures as Instruments of Political Control," *Journal of Law, Economics, and Organization* 3:243-277.

- McCubbins, Mathew D., Roger G. Noll and Barry R. Weingast, 1989. "Structure and Process, Politics, and Policy: Administrative Arrangements and the Political Control of Agencies," *Virginia Law Review* 75:431-482.
- Sandvig, Christian, Kevin Hamilton, Karrie Karahalios, and Cedric Langbort, 2014. "Auditing Algorithms: Research Methods for Detecting Discrimination on Internet Platforms." *Data and Discrimination* 22: 4349-4357
- Semanta, Suranjana and Sameep Mehta, 2017. "Towards Crafting Text Adversarial Samples," *arXiv preprint arXiv:1707.02812*
- Semenova, Lesia, Cynthia Rudin and Ronald Parr, 2019. "On the Existence of Simpler Machine Learning Models," *arXiv preprint arXiv:1908.01755*
- Singh, Jatinder, Jennifer Cobbe, and Chris Norval, 2018. "Decision Provenance: Harnessing Data Flow for Accountable Systems." *IEEE Access* 7: 6562-6574
- Szegedy, Christian, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus, 2014. "Intriguing properties of neural networks." *arXiv preprint arXiv:1312.6199*
- Rudin, Cynthia, 2019. "Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead." *Nature Machine Intelligence* 1.5: 206-215
- Rudin, Cynthia, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong, 2022. "Interpretable Machine Learning: Fundamental Principles and 10 Grand Challenges." *Statistics Surveys* 16: 1-85
- Ting, Michael M. 2001. "The 'Power of the Purse' and its Implications for Bureaucratic Policy-Making." *Public Choice* 106:243-274.