Machine Learning Techniques for Accountability

Been Kim, Finale Doshi-Velez

Artificial intelligence systems have provided us with many everyday conveniences. We can easily search for information across millions of webpages via text and voice. Paperwork processing is increasingly automated. Artificial intelligence systems flag potentially fraudulent credit-card transactions and filter our e-mail. Yet these artificial intelligence systems have also experienced significant failings. Across a range of applications, including loan approvals, disease severity scores, hiring algorithms, and face recognition, artificial-intelligence–based scoring systems have exhibited gender and racial bias. Self-driving cars have had serious accidents. As these systems become more prevalent, it is increasingly important that we identify the best ways to keep them accountable.

Our goal, in this short overview article, is to begin mapping the landscape of methods for accountability of artificial intelligence (AI) systems. For our purposes, we shall define *accountability* as being able to ascertain whether an AI system is behaving as promised, which is necessary for determining blame-worthiness. In the context of a self-driving car, AI system accountability could be a question of safety; in the context of credit scoring, AI system accountability could be a question of fairness. In an algorithmic trading system, the AI system accountability could be a question of performance and robustness to certain shocks. In this overview, we will not focus on any particular objective (such as safety, fairness, or robustness); we believe that defining and refining these objectives for each context is a moral decision that must be made by the public and their representatives, not technologists. Rather, our goal is to begin the process of mapping the categories of methods that one could use to assess whether an AI system is meeting its objectives.

Especially as cases involving AI system behaviors are adjudicated via litigation and prescribed via regulation and legislation, thinking about methods for accountability is essential. In this article, we describe several categories of approaches for accountability in AI systems: transparency (data and process and open-source software); interpretable models; post hoc inspection of model outputs; empirical performance (pre-market and post-market); and properties guaranteed by design. While this list is colored by our experiences in interpretability and health, and by no means exhaustive, we believe it provides a menu that begins to cover mechanisms for accountability appropriate for a wide variety of real-world contexts and objectives. Each category has different tradeoffs with respect to the kinds of infrastructure and real-time human involvement required, as well as risk of exposing sensitive information or trade secrets. The right choice will depend on the specific context, and will likely require AI experts, human factors experts, domain experts, and policymakers to come together to understand the tradeoffs from all angles.

Transparency (Data and Process; Open-Source Software)

Our first set of mechanisms for achieving accountability involves transparency in data, process, and software. We divide these into two major categories: *process transparency* (exposing how the AI system came to be before it was deployed, including data collection, data processing, modeling choices, and training quality); and *software transparency* (releasing code).

Approaches

Process transparency recognizes that the choice of data and training have a huge effect on the output of an AI system. Gebru et al. (2018) advocate for a consistent, unified way of summarizing what kinds of data were used to train a model. Mitchell et al. (2019) describes an analogous approach for describing the models and algorithms that a system uses to make predictions, including relevant parameter and training choices. Software transparency gives people the ability to inspect, and perhaps even run, the AI system's actual code. One can release the code needed to train a system as well as the final code for the trained system (including values of all trained coefficients and parameters). Recently, there have also been efforts to encourage sharing not only the code but also the environment in which the code was run, to help with issues surrounding data libraries and compatibility (for example, Forde et al. 2018).

Benefits

Many applications have errors that come from problems with the data. Thus, even knowing what data and labels were used, and how they were preprocessed, can provide important insights. Might one expect face-detection for auto-focus algorithms, trained largely on pale faces, to work well on darker faces? Would we trust a clinical risk-scoring system differently if the labels were provided by an expert doctor, or by a medical student instead? One may also notice that key variables were not included, such as a drug recommendation system that considers the patient's current vitals, but not their prior history; or interactions that were missing, such as a linear model used to make predictions from raw pixel data. More generally, data and model transparency allow one to assess how likely issues of data shift, mislabeling, missing-ness, and bias may have occurred. Having the code available allows one to check for specific bugs and identify previously unnoticed limitations. If the environment is also available, then anyone can simulate the system for what it would do in various scenarios.

Limitations

Information provided by process transparency that is simply descriptive can help someone guess where problems might lie, but may not be sufficient to confirm that a problem does *not* exist. For example, perhaps the lane-following algorithm of the self-driving car that was trained in sunny places does continue to work in snowy places: we simply don't know. *Process transparency* can tell us we might want to check if the car will drive properly in snowy places, but it cannot tell us whether the car will actually be safe. In contrast, *software transparency* does technically let us test how the AI system may behave in different scenarios, but just having the code available does not mean it is readable; complex AI systems may have millions of lines of code and millions of parameters. Thus, simply inspecting the lines of code may not provide the needed insights and may also require significant domain expertise. And it can't be ignored that software transparency may expose trade secrets and other information that might discourage innovation within the private sector.

Interpretable Models

Another way to ensure that the model is working as expected is to build a model that is inherently easy for humans to understand. This generally entails structuring the model in a specific way (for example, a parametric form, as part of a model, selects the most relevant training example for a prediction). Importantly, the model and the explanation of it are the same; we have made a purposeful effort to ensure that humans can understand the full, true AI system.

Approaches

There are many kinds of interpretable models, including many that predate the recent rise of AI. One category of approach employs *regularizers* (for example, the L1 norm) on existing models to reduce the number of nonzero parameters, and render them easier for humans to understand. Another category uses logic-based or symbolic models. For example, decision trees allow humans to follow a set of logic parameters (for example, AND, OR) and rules (for example, patient age > 30) to show how it reached a prediction. More recently, building simple, yet expressive models (see Doshi-Velez, Wallace, and Adams 2015; Gupta et al. 2016; Kim, Rudin and Shah 2014; Kim, Shah, and Doshi-Velez 2015; Lou, Caruana, and Gehrke 2012; Ustun and Rudin 2014) has been widely studied in AI. Lou, Caruana, and Gehrke (2012) built a model such that the impact of each input feature (such as age) on the prediction is always linear or second-order (such as two features at a time). This enables humans to investigate a prediction in a modular way by, for example, plotting a partialML dependency plot for each feature. Yet another category uses exemplars to ground prediction or clustering decisions (Hase et al. 2019; Kim, Rudin, and Shah 2014).

Benefits

Inherently interpretable models offer inspectable internal representations. The fact that the model's explanations were *part* of the predictions (for example, the model had to simultaneously consider how to explain while making a prediction) gives the user more confidence and may reduce chances of potentially conflicting explanations (as happens in the post hoc interpretable methods below). In Kim, Rudin, and Shah (2014), the model uses similarities between a data point and exemplars in each cluster to do clustering, and these examples themselves are the explanations. In addition, there is more room for customization; the form of explanation that works best may differ across domains. By considering the explanation, the model, and the domain at the time of design rather than later, one can maximally customize for the method that is best for their problem. Especially in high-stakes situations, one can argue that interpretable models are the moral gold standard for accountability (Rudin 2019) because anyone can truly understand what the AI system is doing.

Limitations

Especially in domains with a large amount of inherent stochasticity, it is generally possible to find models that are both human-interpretable as well as highly predictive, as noted above. However, this is not always the case: For example, in computer vision, convolutional neural networks outperform other models by a big margin but remain hard to interpret. Another limitation is that while building a new, interpretable model from scratch is a viable solution in some cases, it may not always be possible. For example, in a big company with a long history of building models and an accumulated code-base and much expertise, one may be in a situation where they would have to work with an existing model. As with software transparency, one might also worry that sharing the model may expose trade secrets.

Post Hoc Inspection of Model Outputs

While interpretable models attempt to make the true internals of the AI system understandable to humans, post hoc inspection methods allow one to identify properties of an already developed AI system. These inspections can be used to determine whether the AI system has behaved incorrectly or inappropriately, providing a mechanism for accountability.

Approaches

There are three main categories of post hoc interpretability methods: visualization; classic statistical methods; and algorithmic methods.

Visualization: These methods include either projecting high-dimensional data/results onto three or fewer dimensions (McInnes, Healy, and Melville 2018; van der Maaten and Hinton 2008) or else better developing a user interface and workflow[1] to reduce the burden of parsing the information. Statistical methods: Conducting rigorous classic statistical tests (for example, qq plots, sensitivity tests, and influence functions) is another powerful way to inspect model performance or data characteristics. Algorithmic methods: Some algorithmic methods resemble classic sensitivity tests, using input features such as saliency maps (Selvaraju et al. 2016; Smilkov et al. 2017; Sundararajan, Taly, and Yan 2017) or higher-level concepts such as testing with concept activation vectors (Kim et al. 2018), and others leverage statistical methods (Koh and Liang 2017). For each data point, saliency maps and saliency testing with concept-activation vectors outputs weights for each input feature/concept — indicating how sensitive each input feature is to the prediction.

Benefits

The explanations provided by post hoc methods could provide insights to humans, whether by confirming their hypothesis (for example, is my model biased toward older people?) or exposing something new (for example, wait, why are pixels on the person in front of a cash machine picture highlighted when prediction is trying to detect a cash machine?). Many of these approaches can be applied to black-box models, meaning that one does not need internals of the model for investigate, only the ability to put in certain inputs and observe the outputs. Thus, they expose the model's internals less overtly. Together, these benefits increase the ability of external agencies, who may have limited access or permissions to the model, to still investigate it. Like the previous approaches, by trying to expose all the relevant factors for how an AI system works, rather than only the ones queried, these post hoc methods enable people to discover flaws that they were not originally looking for (*unknown unknowns*).

Limitations

Most of these methods are subject to human biases. For example, it was recently discovered that saliency maps, supposed to explain a prediction, are produced without much *consideration* of prediction (Adebayo et al. 2018; Nie, Zhang, and Patel 2018; Ulyanov, Vedaldi, and Lempitsky 2017). Julius Adebayo et al. (2018) pointed out that many saliency map methods produce visually and quantitatively similar maps when applied to a trained network and an untrained or random network (which makes random predictions). How did we not know this earlier? Partly because saliency maps were showing "what we expected to see," a classic example of confirmation bias. However, interestingly, some of these methods empirically show that their maps help humans complete a task better or faster. These conflicting results highlight the need for expertise in human–computer interaction with AI: we need human subject studies to find which aspects of these post hoc interpretations are the most helpful to humans in identifying faults when they do exist (while also avoiding confirmation bias). And finally, for the many post hoc interpretation methods that are focused on understanding a specific prediction, these explanations may not allow an expert to understand how a system may work overall.

Empirical Performance (Pre-Market and Post-Market)

So far, all of the methods above have focused on ways to somehow understand an AI system and its limitations — that is, interpretability methods. We now turn to methods for accountability that provide humans some kind of evidence that does not require them to understand anything about the model. In this section, we focus on empirical evidence.

Approaches

The key idea here is that we are interested in the AI system being accountable with respect to certain objectives such as safety, fairness, or performance. If those goals can be quantified, we can simply measure to what extent an AI system is meeting those goals. For example, part of a pre-market safety process for a self-driving car may involve measuring certain kinds of safety violations (such as near misses) in a variety of settings over a series of test runs. A recent US Federal Department of Agriculture approval process for a deep learning-based image processing system in medicine involved comparing the performance of experts with and without the AI system in a variety of settings.[2] This performance can continue to be evaluated post-market through reporting systems for adverse events and regular audits. One also has many choices for who does the checks and monitoring. It could be a relevant government body (US Federal Department of Agriculture, US Environmental Protection Agency), a watch-dog organization, or internally, by a company.

Benefits

Unlike the previous approaches, approaches based on pre-market checks and post-market monitoring don't require a human to understand the model, in any form. All that is needed is input to the model, or even some key statistics from the input (for example, race), and the output. It can be applied to AI systems such as credit scoring that are running relatively autonomously at high volume. It can track small errors that accumulate over time (for example, many small disparities in loan decisions) that methods focused on interpretability or human checks may miss. It can tell us when the rate of errors shifts, which may suggest that the current data streams are no longer like the original training stream. And unlike approaches such as process transparency that can highlight when there *may* be an issue, monitoring can tell us when there *is* an issue. Post-market monitoring also requires relatively little work if the AI system is updated in small ways; no human is needed to recheck the model, but one still has a mechanism to catch unexpected events. And no system internals need to be revealed.

Limitations

The biggest limitation is logistical: requiring that a set of measurements be taken requires consensus between technology and policymakers on what systems should be monitored and how, and it requires personnel to carry out that work over an extended period. One must also generally decide, in advance, the type of data to be collected. In some cases, the choice of protected categories may be clear (for example, race, gender, or age in a lending application), but it won't be able to provide insights about whether the AI system is not serving an unexpected population well, or catch why issues are occurring (for example, whether a problem is a model issue or an adversarial attack).

Properties Guaranteed by Design

Our final general category of approaches to accountability also doesn't require a human to understand a system, and does not even require data: sometimes models or training procedures can be created in which certain properties are guaranteed by design.

Approaches

In general, each desired property will require a different modeling or training procedure. Bounded-Lipschitz networks (Anil, Lucas, and Grosse 2018) ensure that changes in the input will never have more than a prespecified size of effect on the output. Monotone models (You et al. 2017 ensure that increasing a feature (although others are held constant) will always increase or decrease the output (which may be a valuable guarantee such as in a credit scoring system, where we may want to guarantee that increasing income will increase the score). In the privacy and data ownership space, there is now a large body of work on machine learning mechanisms that are provably differentially private under certain assumptions (Chaudhuri, Monteleoni, and Sarwate 2011), federated learning algorithms that guarantee that data can stay in an individual's device for better privacy (McMahan et al. 2016), and algorithms that give the user the right to be forgotten (Ginart et al. 2019). There are algorithms that are provably robust to various kinds of data attacks and poisoning schemes (Steinhardt et al. 2017). Similarly, one can prove that certain algorithms will satisfy certain fairness definitions.

Benefits

When they can be found, the advantage of techniques with guarantees is that we know that they will hold (if the underlying assumptions hold). No understanding of the system is needed; little monitoring is needed. Nothing about the system needs to be exposed.

Limitations

The main drawback is that it is often hard to get guarantees to hold for real scenarios. Algorithms that have guarantees based on some assumption (for example, independent and identically distributed data) will generally lose the guarantees when the assumption is no longer true. On the flip side, forcing an algorithm to have guarantees across a range of settings may reduce its prediction quality overall, while another algorithm satisfying the guarantee most of the time but not always might perform much better. One must carefully understand these tradeoffs.

Conclusion

We have described several general categories of mechanisms for accountability in AI systems. We emphasize that our list is not exhaustive, and we will have to develop additional categories as needs arise. This will be made possible with work from AI experts, human–computer interaction experts, policy and technical experts, and many others with relevant expertise.

Finally, we emphasize that the mechanisms listed above presume that accountability has somehow been defined: there is some definition of fairness, and some notion of safety. The biggest difference between accountability for AI systems versus other systems is that these systems are forcing us to quantify our values. Some domains are more ready than others: For example, airline and car industries have established accountability with respect to safety through guidelines and tests. These guidelines are sometimes found to be incomplete, or subverted, but at least the mechanisms are there. In contrast, while equal protections for hiring is a goal, it remains to be determined how exactly it will be formalized in various circumstances. Other areas, such as narrow-casting news and ads, are so new that regulation and public consensus on what should be allowed is not yet here. As we refine our definition of what accountability means in various contexts, we, as a community, will establish the target that our accountability methods must achieve.

Notes

1. arterys.com

2. See J. Wexler, The What-If Tool: Code-Free Probing of Machine Learning Models. https://ai.googleblog.Com/20J. 18/09/The-what-if-Tool-code-free-probing-Of.html

References

Adebayo, J.; Gilmer, J.; Muelly, M.; Goodfellow, I.; Hardt, M.; and Kim, B. 2018. Sanity Checks for Saliency Maps. arXiv preprint. arXiv:1810.03292[cs.CV]. Ithaca, NY: Cornell University Library.

Anil, C.; Lucas, J.; and Grosse, R. 2018. Sorting Out Lipschitz Function Approximation. arXiv preprint. arXiv:1811.05381 [cs.LG]. Ithaca, NY: Cornell University Library.

Chaudhuri, K.; Monteleoni, C.; and Sarwate, A. D. 2011. Differentially Private Empirical Risk Minimization. *Journal of Machine Learning Research* 12:1069–109.

Doshi-Velez, F.; Wallace, B.; and Adams, R. 2015. Graph-Sparse LDA: A Topic Model with Structured Sparsity. In *Proceedings of the Twenty-Ninth Association for the Advancement of Artificial Intelligence (AAAI) Conference on Artificial Intelligence*, 2575–81. Palo Alto, CA: AAAI Press.

Forde, J.; Head, T.; Holdgraf, H.; Panda, Y.; Nalvarete, N.; Ragan-Kelley, B.; and Sundell, E. (2018) Reproducible Research Environments with Repo2Docker. Paper presented at the Reproducibility in Machine Learning Workshop 2018. OpenReview.net.

Gebru, T.; Morgenstern, J.; Vecchione, B.; Vaughan, J.; Wallach, W.; Daumeé, H.; and Crawford, K. 2018. Datasheets for Datasets. arXiv preprint. arXiv:1803.09010[cs.DB]. Ithaca, NY: Cornell University Library.

Ginart, A.; Guan, M.; Valiant, G.; Zou, J. Y. 2019. Making AI Forget You: Data Deletion in Machine Learning. Eds. Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., in *Advances in Neural Information Processing Systems 32* (*NeurIPS 2019*). papers.nips.cc/paper/2019/hash/cb79f8fa58b91d3af6c9c991f63962d3-Abstract.html.

Gupta, M.; Cotter, A.; Pfeifer, J.; Voevodski, K.; Canini, K.; Mangylov, A.; Moczydlowski, W.; and van Esbroeck, A. 2016. Monotonic Calibrated Interpolated Look-Up Tables. *Journal of Machine Learning Research* 17:109:1–109:47.

Hase, P.; Chen, C.; Li, O.; and Rudin, C. 2019. Interpretable Image Recognition with Hierarchical Prototypes. In *Proceedings of the 7th Association for the Advancement of Artificial Intelligence (AAAI) Conference on Human Computation and Crowdsourcing*, 32–40. Palo Alto, CA: AAAI Press.

Kim, B.; Rudin, C.; and Shah, J. 2014. The Bayesian Case Model: A Generative Approach for Case-Based Reasoning and Prototype Classification. In Neural Information Processing Systems 2014. arXiv preprint. ArXiv:1503.01161[stat.ML]. Ithaca, NY: Cornell University Library.

Kim, B.; Shah, J.; and Doshi-Velez, F. 2015. Mind the Gap: A Generative Approach to Interpretable Feature Selection and Extraction. Paper presented at the *Advances in Neural Information Processing Systems 28* (*NIPS 2015*). papers.neurips.cc/paper/5957-mind-the-gap-a-generative-approach-to-interpretable-feature-selection-and-extraction

Credit: mitay, iStock

Kim, B.; Wattenberg, M.; Gilmer, J.; Cai, C.; Wexler, J.; Viegas, F.; and Sayres, R. 2018. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). *Proceedings of Machine Learning Research* 80:2668–77.

Koh, P. W., and Liang, P. 2017. Understanding Black-Box Predictions via Influence Functions. arXiv preprint. arXiv:1703.04730[stat.ML]. Ithaca, NY: Cornell University Library.

Lou, Y.; Caruana, R.; and Gehrke, J. 2012. Intelligible Models for Classification and Regression. In *Proceedings of the 18th Association for Computing Machinery (ACM) Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD) International Conference*. New York: ACM. doi.org/10.1145/2339530.2339556

McInnes, L.; Healy, J.; and Melville, J. 2018. Umap: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv preprint. arXiv:1802.03426[stat.ML]. Ithaca, NY: Cornell University Library.

McMahan, H. B.; Moore, E.; Ramage, D.; Hampson, S.; and Arcas, B. 2016. Communication-Efficient Learning of Deep Networks from Decentralized Data. arXiv preprint. arXiv:1602.05629[cs.LG]. Ithaca, NY: Cornell University Library.

Mitchell, M.; Wu, S.; Zaldivar, A.; Barnes, P.; Vasserman, L.; Hutchinson, B.; Spitzer, E.; Raji, I.; and Gebru, T. 2019. Model Cards for Model Reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 220–9. New York: Association for Computing Machinery. dl.acm.org/doi/abs/10.1145/3287560.3287596

Nie, W.; Zhang, Y.; and Patel, A. 2018. A Theoretical Explanation for Perplexing Behaviors of Backpropagation-Based Visualizations. *Proceedings of Machine Learning Research* 80:3806–15.

Rudin, C. 2019. Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead. *Nature Machine Intelligence* 1(5): 206–15. doi.org/10.1038/s42256-019-0048-x

Selvaraju, R.; Das, A.; Vedantam, R.; Cogswell, M.; Parikh, D.; and Batra, D. 2016. Grad-Cam: Why Did You Say That? Visual Explanations from Deep Networks via Gradient-Based Localization. arXiv preprint. arXiv:1610.02391[cs.CV]. Ithaca, NY: Cornell University Library.

Smilkov, D.; Thorat, N.; Kim, B.; Viégas, F.; and Wattenberg, M. 2017. Smoothgrad: Removing Noise by Adding Noise. arXiv preprint. arXiv:1706.03825[cs.LG]. Ithaca, NY: Cornell University Library.

Steinhardt, J.; Wei, P.; Koh, W.; and Liang, P. S. 2017. Certified Defenses for Data Poisoning Attacks. *In* Advances in Neural Information Processing Systems 30, Annual Conference on Neural Information Processing Systems 2017 (NIPS 2017). papers.nips.cc/paper/2017/hash/9d7311ba459f9e45ed746755a32dcd11-Abstract.html

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic Attribution for Deep Networks. *Proceedings of Machine Learning Research* 70:3319–28.

Ulyanov, D.; Vedaldi, A.; and Lempitsky, V. 2017. Deep Image Prior. arXiv preprint. arXiv:1711.10925[cs.CV]. Ithaca, NY: Cornell University Library.

Ustun, B., and Rudin, C. 2014. Methods and Models for Interpretable Linear Classification. ArXiv preprint. arXiv:1405.4047[stat.ML]. Ithaca, NY: Cornell University Library.

van der Maaten, L., and Hinton, G. 2008. Visualizing Data Using T-SNE. *Machine Learning* 87(1): 33–55.

You, S.; Ding, D.; Canini, K.; Pfeifer, J.; and Gupta, M. 2017. Deep Lattice Networks and Partial Monotonic Functions. arXiv preprint. arXiv:1709.06680[stat.ML]. Ithaca, NY: Cornell University Library.

Been Kim is a research scientist at Google Brain. Her research focuses on improving interpretability in machine learning by building interpretability methods for already-trained models or building inherently interpretable models.

Finale Doshi-Velez is a John L. Loeb associate professor in computer science at the Harvard Paulson School of Engineering and Applied Sciences. She completed her MSc from the University of Cambridge as a Marshall Scholar, her PhD from the Massachusetts Institute of Technology, and her postdoc at Harvard Medical School. Her interests lie at the intersections of machine learning, healthcare, and interpretability.