14 / Full Stack Rhetoric

A Response to *Rhetorical Machines*

Annette Vee

The popular notion of the computer as an inflexible object of pure engineering is—in the few corners it might still hold sway—being buffeted by critical work in the history of computing, software studies, digital humanities, algorithm studies, and now computational rhetoric. This work has taught us that the machine with which we labor, socialize, and govern is designed not only using the properties of silicon, magnetic tape, and voltages, but also with social biases, idiosyncratic personal preferences, and sometimes dubious motivations. Tara McPherson has outlined the parallels between the design of the influential operating system UNIX and race relations in mid-twentieth-century America, noting that "computers are themselves encoders of culture . . . [and] code and race are deeply intertwined."[1] Jennifer Light, Janet Abbate, Nathan Ensmenger, Margot Lee Shetterly, and Marie Hicks have all uncovered important and hitherto hidden histories of women in computing, collectively emphasizing that computer hardware and software were not designed with certain groups in mind—and yet those groups made their way into the machines anyhow.[2] Essential examinations of algorithms by Tarleton Gillespie, Nicholas Diakopoulos, and Safiya Umoja Noble have given us insight into the central—and sometimes scary—role these mathematical underpinnings of software have in our lives, and how they sort some lives and experiences differently from others.[3]

Essential to this line of research on the computer's cultural influences is attention to its "full stack," everything from the electrical impulses that we render as source code to the images on its screen. The present collection provides this perspective: we see here a focus on the computer's mathematical programs and engineering history as well as its rendering of images in contemporary videogames—and much in-between. As this collection shows, rhetoric is an apt tool for prying open the multiple layers of the computer's stack, so much so that Lavinia Hirsu and John Jones call the machine itself rhetori-

cal. This collection thus contributes to a growing body of work in computational rhetoric.

Still in its infancy, computational rhetoric has already demonstrated the ways that machines can perform the functions of rhetoric, traditionally thought to be the exclusive domain of humans. When machines perform rhetoric, humans can get pretty uncomfortable—as we have in the case of some bots. Stuart Geiger, studying a particular bot that supplied signatures to unsigned edits to Wikipedia, observed that human Wikipedia editors felt there was a critical difference between generally accepted human norms and a bot that automatically forced compliance with this norm.[4] This bot is just one example of a norm-enforcing machine, a computational encoding of human relations in online discourse. From this study, Geiger concludes that bots are hybrid operators, "both editors and software, social and technical, discursive and material, as well as assembled and autonomous."[5]

Our notion of computation's relationship to rhetoric is shifting with the increasing sophistication of computation and its intertwining with linguistic exchanges, especially online. As a result, Douglas Guilbeault argues that we must expand our ideas of rhetorical agency to accommodate the work that bots do. For instance, social bots on Twitter can take advantage of Tarleton Gillespie's "calculated publics" by harvesting highly rated photos from Hotornot.com, accruing followers and boosting their popularity characteristics, and then leveraging automated conversational tools such as retweeting, emoticons, and link-sharing. Through the calculated popularity measures of Twitter, users such as politicians, celebrities, or businesses can then harness these bots to increase their own perceived popularity. Noting this phenomenon and its influence on politics and discourse, Guilbeault introduces the idea of "platform persuasion," which describes the ways bots leverage design aspects of a platform to have significant influence on online networks. He argues that since bots can influence networks, it is important to move away from a human-centered idea of agency and to think of bots as agents as well—or, in terms of computational rhetoric, as software rhetors.[6] James Brown Jr.'s research on the ways that software networks must grapple with issues of hospitality—also a domain generally considered exclusive to humans—further demonstrates how computational machines are rhetorical entities.[7]

Although not all of it flies under the banner of computational rhetoric, work on the ways that language shapes computational machines is also important to our understanding of *rhetorical machines*. Chilean philosopher Leonardo Flores and Terry Winograd—the influential computer scientist and graduate advisor to Google founders Sergey Brin and Larry Page—argued in the 1980s that "in order to become aware of the effects that computers have on society we must reveal the implicit understanding of human language, thought and

work that serves as a background for developments in computer technology."[8] Basic ideas about how computers work are influenced by the language we use to describe them—perhaps most famously by John von Neumann's designation of the computer's storage as "memory," which opened the doors to considering computers as thinking machines. Wendy Hui Kyong Chun's *Programmed Visions* points out this and other influential genetic and biological metaphors for the computer.[9] David Nofre, Mark Priestley, and Gerard Alberts describe the way that programming began to be thought of as a linguistic activity in the early 1960s and claim that "the language metaphor in programming [is] one of the most essential metaphors around which computer science has been built."[10] My own work on the laws governing software extends legal discussions of patent, first amendment, and copyright law into the realm of rhetoric through an assertion that each legal regime implies certain uses and audiences for computer code.[11] Similarly, in this collection, Hammond uses a detailed history of automated essay scoring to show "technology's rhetorical priority of definition: definition precedes (i.e., is prior to) rhetorical engagement with-or-through technologies, and questions of definition and essence cut to the rhetorical core of technologies, revealing what they are imagined to do (or not) and how they are assessed as working (or not)." As Lisa Gitelman and Janet Abbate have both argued, the metaphors we choose when we discuss technology influence the ways the technology is used, and by whom.[12]

In the introduction to a special issue of *Computational Culture* on computational rhetoric, Brown and I took the connection of rhetoric and computation one step further: we argue that computational machines are rhetorical not only by virtue of their relationship to language but also in their machinic operations in our everyday lives. We wrote, "even the most mundane computational technologies can be seen as rhetorical—from the grocery store checkout scanner to the high school graphing calculator—because any computational machine shapes and constrains behavior." Work in that edited collection sets the stage for what appears here: on the ways that error is defined machinicly and linguistically; on rhetorical justifications for algorithms; on the influence of style on coding decisions; and on the ways that artificial intelligence is also artificial rhetoric.[13] Through all of these explorations, we are learning how amenable to rhetoric computers can be. As Hammond puts it in this collection, "far from being the (technical) opposite of (social) rhetoric, computation is inextricable from the social and rhetorical."

Moreover, the stakes for work in computational rhetoric are high: Elizabeth Losh asserts here that "it may actually be the conflation of gender and technology at work in the popular imagination that was to blame for Clinton's stunning defeat" in the 2016 US Presidential election. In other words, the rhetorical configuration of technology not only shapes whom we think should

be using it but may also be powerful enough to sway elections. Closer to the metal, Ryan Omizo suggests that one reason rhetoricians should be creating computational objects that engage with rhetoric is so that we can ethically shape the ways they are built. Along the same lines, writing computational processes while ignoring the real work of rhetoric operating within them is downright dangerous, Jennifer Maher, Helen Burgess, and Tim Menzies powerfully state. Big data is a rhetorical enterprise, they argue, and without rhetorical attention to its collection, processing, and implications, we risk a literally whitewashed perspective on the world—one that also has real effects. As Anthony Stagliano memorably puts it here, code is powerful, but it's also much *wilier* than we often give it credit for. A takeaway from this collection is that this computational wiliness can be traced the whole way down the stack.

Indeed, its "full stack" coverage is one of the most compeling things about this book's collection of chapters: from the math that drives the algorithms that influence human decisions about roadway repair (Juszkiewicz and Warfel) or the collection of big data as a rhetorical enterprise (Maher, Burgess, and Menzies) to the cutscenes that create "ambient rhetoric" reflecting and shaping cultural assumptions about race in videogames (Daniel-Wariya and Sanchez). Hammond focuses on the rhetorical framing of automated essay scoring, noting that the ways teachers and machines interacted in its early history shapes what we think of both teachers and the "collections of machines" that comprise writing assessment. Stagliano asserts that software itself can respond to rhetorical situations in complex ways; in his chosen case of CV Dazzle, the software responds to ubiquitous surveillance and intervenes in human relations. Jennifer Juszkiewicz and Joseph Warfel go deeper into the workings of computation to argue that even the math driving algorithms is rhetorical. Using the field of Operations Research as a guide, they point out that algorithms are really "a set of mathematical statements that, when considered simultaneously, describe a human's perception of a system." Jonathan Buehl and Kevin Brock both examine the extant text and the design of computational systems in tandem, indicating how intertwined language, culture, and the engineering or programming constraints of the system can be. Buehl charts a fascinating path through Charles Babbage's writings to reveal the ways that his appeals for funding shaped his influential protocomputer, ultimately claiming that "Charles Babbage's Difference Engine was an engine of rhetoric. The project was enabled by clever rhetorical work; the engine in turn produced both new rhetorical situations and innovative responses." Through his close reading of both code and its explanations in the open source project Ruby on Rails, Brock argues, "How these various actors are compelled to act in response to these influences depends heavily not just on who the author of a given code text is but how that author argues, explicitly or implic-

itly, in and through his or her coded procedures and relevant style toward particular actions and activities." Maher and Burgess take a close look at a study of GitHub pull requests to emphasize the many layers of the stack that those researchers had to consider to do justice to their subject, and they conclude that big data is an enterprise of "big rhetoric." Joshua Daniel-Wariya and James Chase Sanchez are close to the computer's surface in their analysis of racialized depictions of characters in videogames, but their attention to the design constraints in the software engines that power these games sets their work apart from a more traditional literary approach to games. Losh provides a rhetorical perspective on the social layers of the computer as they play out across big questions of gender and politics on the world stage. Beyond an informed analysis of the full stack, Ryan Omizo goes so far as to *create* a software interface for socially complex rhetorical interventions, which he discusses in his contribution to this collection. Following Maher and Burgess's move from "big data" to "big rhetoric," I might say that we see here in this collection "full stack rhetoric."

While editing the first Software Studies definitional collection, *Software Studies: A Lexicon*, Matthew Fuller was adamant that each of the writers be well versed in both the academic/critical angle to software *and also* the mechanics of software itself.[14] This is clearly the case with the writers in this collection as well. Is this the new standard for work in computational rhetoric? Should it be? For rhetoricians aspiring to work in this area of research, this is a tall order. Already, digital rhetoric demands much from its practitioners: extensive cross-disciplinary knowledge of both rhetoric and technology; keeping up with new versions of software; constant retooling in the face of changing trends of programs and digital design. Must we add to this a facility with computer programming, or even the electrical or mathematical underpinnings of the computer? The term "full stack" comes from the profession of web development and encompasses everything from the user interface of websites to the backend databases and code libraries that support websites. The profession of web development was enthusiastic about the prospect of "full-stack developers" in the mid-2000s—developers who could do both front end and back end work—because these developers could minimize miscommunications across design levels; from the pragmatic point of view, companies could hire just one person to make a complex website. But as the stack has become more complex, expertise in all its levels has become nearly impossible, and the profession of web development now generally concedes that "full-stack developers" are as common as unicorns. Following this trend in web development, I wonder: is full-stack rhetoric possible, or desirable?

I think it is desirable, even necessary. But full-stack rhetoric is really only possible as a collaboration, in collections such as this one that draw from a

variety of expertise. Fuller notes that his lexicon "aims to make available some of the mixed intelligences thinking through these conditions [of contemporary software]. The authors are artists, computer scientists, designers, philosophers, cultural theorists, programmers, historians, media archaeologists, mathematicians, curators, feminists, musicians, educators, radio hams, and other fine things, and most straddle more than one discipline. The voices collected here bring more than one kind of intelligence to software because software makes more sense understood transversally."[15] Similarly, a project with the intellectual and technical range of the present collection is only manageable with a range of expertise such as these contributors have. What is essential to good work in computational rhetoric is that we know the full stack exists—that we know it is rhetoric all the way up and down, from data to math to images in games. It is not *purely* rhetoric though, which is the tricky part. There are non-negotiable parts of engineering and math, or pre-existing designs (whatever their contemporary motivations or biases) that constrain our choices and make some aspects of the computer less susceptible and less accessible to rhetoric than others. As rhetoricians who might want our work to travel across disciplines, we must know the difference.

Once we are aware of the stack's existence, once we have a basic understanding of how the layers create a complex object of analysis and rhetoric, we can hone in on specific layers, conscious of their relationship to others. Collaborative projects, such as this entire collection and some of its individual pieces, will be essential for examining the intersections between particular layers. The result can be fascinating analyses of the ways rhetoric makes its way into the ubiquitous computational machine that influences our lives so significantly.

Our rhetorical interventions into computation have, by nature, ethical underpinnings, as Brown has previously asserted.[16] As many of the pieces in this collection argue, rhetoricians have a responsibility to do this work. If we don't, others will, and they may not have the training in language and persuasion that rhetoric affords us. This is clearly articulated in Omizo's contribution, where a rhetorical machine is constructed to help humans intervene in online arguments. The alternative is, perhaps, Tay, the disastrous chatbot released by Microsoft in 2016—or other rhetorical machines constructed without the sensitivity to how language works in social situations.[17] An understanding of natural language processing might be critical to these endeavors, as Omizo indicates. But just as necessary is understanding and attention to how language functions affectively and rhetorically, how it persuades, insults, flatters, or praises. Also important to these interventions is attention to pedagogy, as Hammond notes. What are our computational and rhetorical machines teaching us, society at large, or our children? Elsewhere, Noble's study of the racial

bias of algorithms, the influential collection *Race after the Internet* edited by Lisa Nakamura and Peter Chow-White, and the work here by Daniel-Wariya and Sanchez all point to the lessons of racial representation in computational spaces. Maher and Burgess also emphasize the importance of ethics here: "data science is too often limited to notions of goodness as either social consequences or matters of technique, the effect of which is the displacement of human conceptions of goodness to those of the machine." As they note, the ethical basis of computing was a worry even for Norbert Weiner, who asserted that only those who knew about the workings of the computer understood the moral positions it implied. Since Weiner, many computer scientists have made the point that ethical approaches to computing are critical. Programmer Daniel Kohanski warned that computers do not have inherent ethics; they simply magnify human thoughts and instructions, and so we must be quite careful what we relegate to the machine.[18] In the same spirit, Terry Winograd founded the influential organization Computer Professionals for Social Responsibility, which emphasizes the necessity of ethical approaches to computing and even traces its philosophy back to Weiner through its Norbert Wiener Award for Social and Professional Responsibility to notable computer scientists such as Doug Englebart, Kristen Nygaard, and Mitch Kapor. It is important for rhetoricians to acknowledge that computational rhetoric is not alone in providing ethical approaches to computing.

With collaboration from computer science and other fields, computational rhetoric has a critical role to play in both the analysis and construction of rhetorical machines in the future. Fuller's lexicon, the special issue on computational rhetoric in *Computational Culture*, and the present collection are all ways of asserting our role. I look forward to seeing more great work to come in computational rhetoric—that is, explorations that acknowledge the computer's full stack and ethically wield the tools of rhetoric to illuminate what is still too often thought of as an inflexible black box. The computer is a *rhetorical machine* that calls for *full-stack rhetoric*.

Notes

1. Tara McPherson, "US Operating Systems at Mid-Century," in *Race after the Internet*, ed. Lisa Nakamura and Peter Chow-White (New York: Routledge, 2012), 36.

2. Jennifer Light, "When Computers Were Women," *Technology and Culture* 40, no. 3 (July 1999): 455–83; Nathan Ensmenger, *The Computer Boys Take Over* (Cambridge: MIT Press, 2010); Margot Lee Shetterly, *Hidden Figures* (New York: Harper Collins/William Morrow and Company, 2016); Marie Hicks, *Programmed Inequality* (Cambridge: MIT Press, 2017); Janet Abbate, *Recoding Gender* (Cambridge: MIT Press, 2012).

3. Tarleton Gillespie, "The Relevance of Algorithms," in *Media Technologies*, ed.

Tarleton Gillespie, Pablo Boczkowski, and Kristen Foot (Cambridge: MIT Press, 2014), 167–94; Nicholas Diakopoulos, "Algorithmic Accountability," *Digital Journalism* 3, no. 3 (2015): 398–415; Safiya Umoja Noble, *Algorithms of Oppression* (New York: NYU Press, 2018).

4. R. Stuart Geiger, "The Lives of Bots," in *Wikipedia: A Critical Point of View*, ed. Geert Lovink and Nathaniel Tkacz (Amsterdam: Institute of Network Cultures, 2011), 87.

5. Ibid., 92.

6. Douglas Guilbeault, "Growing Bot Security: An Ecological View of Bot Agency," *International Journal of Communication* 10 (2016): 5009–10.

7. James J. Brown Jr., *Ethical Programs: Hospitality and the Rhetorics of Software* (Ann Arbor: University of Michigan Press, 2015).

8. Terry Winograd and Fernando Flores, *Understanding Computers and Cognition: A New Foundation for Design* (Norwood, NJ: Ablex Publishing Corporation, 1986), 7.

9. Wendy Hui Kyong Chun, *Programmed Visions: Software and Memory* (Cambridge: MIT Press, 2011).

10. David Nofre, Mark Priestley, and Gerard Alberts, "When Technology Became Language: The Origins of the Linguistic Conception of Computer Programming, 1950–1960," *Technology and Culture* 55, no. 1 (January 2014): 43.

11. Annette Vee, "Text, Speech, Machine: Metaphors for Computer Code in the Law," *Computational Culture: A Journal of Software Studies* 2 (2012), http://computationalculture.net/article/text-speech-machine-metaphors-for-computer-code-in-the-law.

12. Abbate, *Recoding Gender*; Lisa Gitelman, *Scripts, Grooves and Writing Machines* (Stanford, CA: Stanford University Press, 2000).

13. Annette Vee and James J. Brown Jr., eds., Special Issue on Rhetoric and Computation, *Computational Culture: A Journal of Software Studies* 5 (2016), http://computationalculture.net/issue-five/.

14. MIT Press's overview of Matthew Fuller's *Software Studies* notes that "the contributors to Software Studies are both literate in computing (and involved in some way in the production of software) and active in making and theorizing culture." MIT Press, "Software Studies," MIT Press, n.d. https://mitpress.mit.edu/books/software-studies.

15. Matthew Fuller, "Introduction, The Stuff of Software," in *Software Studies: A Lexicon*, ed. Matthew Fuller (Cambridge: MIT Press, 2008), 10.

16. Brown Jr., *Ethical Programs*.

17. James Vincent, "Twitter Taught Microsoft's AI Chatbot to Be a Racist Asshole in Less than a Day," The Verge, last updated Mar 24, 2016, https://www.theverge.com/2016/3/24/11297050/tay-microsoft-chatbot-racist.

18. Daniel Kohanski, *The Philosophical Programmer: Reflections on the Moth in the Machine* (New York: St. Martin's Press, 1998), 21.

Bibliography

Abbate, Janet. *Recoding Gender*. Cambridge: MIT Press, 2012.

"About Us." Girls Who Code. Accessed January 23, 2017. https://girlswhocode.com/about-us/.

Aikat, Debashis. "Big Data Dilemmas: The Theory and Practice of Ethical Big Data Mining for Socio-Economic Development." In *Ethical Data Mining Applications for Socio-Economic Development*, edited by Hakikur Rahman and Isabel Ramos, 106–30. Hershey, PA: IGI Global, 2013.

Anderson, Karrin Vasby, and Kristina Horn Sheeler. "Texts (and Tweets) from Hillary: Meta-Meming and Postfeminist Political Culture." *Presidential Studies Quarterly* 44, no. 2 (June 2014): 224–43.

Anderson, Lark. "*Resident Evil 5* Is a Fun and Frantic Evolution." GameSpot. March 13, 2009. https://www.gamespot.com/reviews/resident-evil-5-review/1900–6228853/.

Anderson, Logan. "So Many Monsters: But Never a Man!" *The Phi Delta Kappan* 35, no. 9 (June 1954): 369, 392.

"Anybody Can Learn." Code.org. Accessed January 23, 2017. https://code.org/.

apotonick et al. "Simplify Finding Default Layout #15050." GitHub. Last modified May 9, 2014. https://github.com/rails/rails/pull/15050.

Aristotle. "Nichomachean Ethics." In *The Complete Works of Aristotle*, vol. 2, edited by Jonathan Barnes, 1729–1867. Princeton: Princeton University Press, 1984.

———. "Rhetoric." In *Complete Works of Aristotle*, 2:2152–69.

"Attention Newbies to Incubating." Backyardchickens.com. 2014. Accessed June 4, 2014. http://www.backyardchickens.com/t/878773/attention-newbies-to-incubating.

Babbage, Benjamin Herschel. "Recreations of a Philosopher." *Harper's New Monthly Magazine* 30, no. 175 (1864): 34–39.

———.*Reflections on the Decline of Science in England and Some of its Causes*. New York: Augustus M. Kelly, 1970.

———."On the Theoretical Principles of the Machinery for Calculating Tables." *The Works of Charles Babbage*, vol. 2, edited by Martin Campbell-Kelly, 38–43. New York: NYU Press, 1989.

Babbage, Charles. "A Letter to Sir Humphry Davy on the Application of the Ma-