10-2017

Why We Should Have Seen That Coming: Comments on Microsoft's Tay "Experiment," and Wider Implications

K. W. Miller
University of Missouri-St. Louis

Marty J. Wolf
Bemidji State University

Frances S. Grodzinsky
Sacred Heart University

ORBIT

Why We Should Have Seen That Coming

Comments on Microsoft's Tay "Experiment," and Wider Implications

Wolf, M.J.
Bemidji State University

Miller, K.W.
University of Missouri-St. Louis

Grodzinsky, F.S.
Sacred Heart University

Corresponding Author: Marty J. Wolf, mjwolf@bemidjistate.edu

Abstract

In this paper we examine the case of Tay, the Microsoft AI chatbot that was launched in March, 2016. After less than 24 hours, Microsoft shut down the experiment because the chatbot was generating tweets that were judged to be inappropriate since they included racist, sexist, and anti-Semitic language. We contend that the case of Tay illustrates a problem with the very nature of learning software (LS is a term that describes any software that changes its program in response to its interactions) that interacts directly with the public, and the developer's role and responsibility associated with it. We make the case that when LS interacts directly with people or indirectly via social media, the developer has additional ethical responsibilities beyond those of standard software. There is an additional burden of care.

Keywords: learning software development, responsibility, AI, technologies of humility, software profession

Introduction

On March 23, 2016, Microsoft Corporation unveiled a new chatbot they named "Tay." Tay was to interact with human users on the Internet via Twitter and pick up human habits of speech (Larson, 2016; Victor, 2016). After less than 24 hours, Microsoft shut down the experiment because the chatbot was generating tweets that were judged to be inappropriate since they included racist, sexist, and anti-Semitic language.

Soon afterward, Microsoft was widely criticized for deploying Tay in the way that they had. Selena Larson wrote, "…Microsoft and Twitter suffer from the same problem: a lack of awareness or understanding as to what potential harm these technologies can do, and how to prevent it in the first place" (Larson, 2016). This claim is corroborated by the statement that Microsoft emailed that included: "Unfortunately, within the first 24 hours of coming online, we became aware of a coordinated effort by some users to abuse Tay's commenting skills to have Tay respond in inappropriate ways" (Victor, 2016).

This was not the first time a major U.S. tech company launched web-based software that ended up embarrassing the developers. In May of 2015, Google released "Google Photos," an online bot that, among other things, "learned" from users how to label photos. Unfortunately, the software was found to be labeling photos of black people as "gorillas" (Dougherty, 2015).

In this paper, we will explore the idea that these two incidents are more than isolated cases of technical programming errors. We contend that these incidents are symptoms of a deeper problem with the very nature of learning software (LS-a term that we will use to describe any software that changes its program in response to its interactions) that interacts directly with the public, the developer's relationship with it, and the responsibility associated with it. We make the case that when LS interacts directly with people or indirectly via social media, the developer has additional ethical responsibilities beyond those of standard software. There is an additional burden of care.

The differences in the ethical implications between learning and non-learning software have been known for some time. In the next section, we will identify some previous scholarship that warned of the potential ethical problems that arise with the development of learning software. These concerns are exacerbated today due to the prevalence of social media. We use the case of Tay to illustrate these concerns. In the third section, we argue that there are additional normative responsibilities for those who develop learning software that interacts with the general public. We conclude the paper with a short summary.

Existing Literature and Its Application to Tay

In 2008, we identified some of the responsibilities surrounding the development of learning software. We wrote:

> Our focus in this paper has been on the designers of artificial agents. We have argued that these designers need to take great care particularly in developing artificial agents that exhibit learning* and intentionality*. However, we are not arguing that designers are alone in having responsibilities associated with such agents. The buyers of artificial agents, the trainers of neural nets, and anyone who deploys an artificial agent (for example, a bot on the Internet or an agent controlling a physical robot) all share the responsibility of avoiding harmful consequences that might arise from the deployment of the artificial agent. It is not sufficient

> to claim that ignorance of \an agent's eventual behaviors insulates these stakeholders from ethical claims regarding the agent's behaviors. We assert that this ignorance, willful or unintentional, is itself an ethical lapse, a lapse that is shared (Grodzinsky, Miller, & Wolf, 2008).

It should come as no surprise that the problems associated with Tay surfaced. The application of our 2008 concerns to the case of Tay is obvious, but no less relevant. Based on Peter Lee's statement on behalf of Microsoft in response to Tay's bad behavior (Lee, 2016), it seems that they were well aware that harmful consequences might occur. They "planned and implemented a lot of filtering and conducted extensive user studies with diverse user groups. We stress-tested Tay under a variety of conditions, ..." (Lee, 2016). While Lee claims that this was "a coordinated attack by a subset of people [who] exploited a vulnerability in Tay" (Lee, 2016), it is unclear whether this vulnerability was apparent to the developers prior to or only after the release of Tay. An important point of our article quoted above is that LS *always* has this sort of vulnerability, and therefore, a developer of LS should adopt a position of expecting this behavior. The developer cannot be confident about knowing *how* the system will behave because of the *nature* of software that learns.

Another important observation is that LS developers need to be more keenly aware of their ethical responsibilities. The ACM Code of Ethics says that computer professionals must give "comprehensive and thorough evaluations of computer systems and their impacts, including analysis of possible risks" (Anderson et al., 1992). And, while the developers may not have anticipated this particular risk, they should have anticipated that Tay might behave in a way they did not anticipate. Such a risk should have been mitigated prior to release. Taking Tay offline when it became abusive was reactive, not proactive.

LS that is designed to behave in human-like ways raises additional ethical concerns. In 2011, we wrote (AA here stands for artificial agent):

> Some AA developers are attempting to make AAs more human-like by programming them to be more adaptable to their environment by allowing them to self-modify their programs. We contend that the potential gains of this strategy are not sufficient to justify the enormous risks, especially when the adaptation process is poorly understood by the developer and not easily recognized by humans who have e-trust relationships with the AAs. We prefer that AAs be boringly predictable. We are far more concerned about the trustworthiness of AAs and far less concerned that they mimic human adaptability. In almost all situations, we think that AA developers have an obligation to the safety of the public. That duty should restrict their use of self-modifying code to implement AAs and place limitations on the use of neural nets in AAs (Grodzinsky, Miller, & Wolf, 2011).

One of the reasons that Tay was deployed on Twitter was to "experiment" with a learning algorithm designed to acquire human conversational skills (Victor, 2016). The fact that

the software ended up behaving in a way that its developers had not anticipated demonstrates that the Microsoft team released the software without having a clear sense of the breadth of possible ways the bot would develop after deployment. As we explained in 2011, we consider that ethically unacceptable and reckless. The Latin proverb "*ignorantia juris non excusat*" is loosely translated "ignorance of the law is no excuse." In the case of LS, we adapt that saying to "ignorance of a program's future behavior is no excuse" for dismissing responsibility for software that one releases.

More recently, Ibo van de Poel set about developing an ethical framework for evaluating new technology (van de Poel, 2016). In this work, he identifies new technologies as experimental "if there is only limited operational experience with them, so that social benefits and risks cannot, or at least not straightforwardly, be assessed on basis of experience" (van de Poel, 2016). Tay clearly meets this definition of experimental, especially since the engineers at Microsoft envisioned Tay as an experiment. Thus, van de Poel's more general moral principles for responsible experimentation with new technology, non-maleficence, beneficence, respect for autonomy, and justice, also apply here. While a thorough application of the conditions of his ethical framework to this case is beyond the scope of this paper, it is clear that while Microsoft was consistent with the some of the principles, there are demonstrable points of inconsistency.

In the remainder of this section we expand on communication styles, deception, and the practice of the computing profession as applied to Tay.

Communication styles

According to Daniel Victor, "Microsoft had a fairly reasonable goal here: They wanted to develop better 'conversational understanding' for their products. Part of the reason computers and humans don't interact well is that humans tend to communicate obliquely while robots think literally"(Seitz, 2016). Was this goal realistic?

Alvidrez and Rodriguez have shown that people tend to use different communication styles that rely on the social context in which they communicate (Alvídrez & Franco-Rodríguez, 2016). In order to understand social mobilization facilitated by Twitter, Tay would have had to have the power to interpret these styles. Twitter is good for social mobilization: "Twitter's speed and reach have made it a communication tool used widely by public figures to attract the attention of users, creating emotional bonds with their followers and ultimately, mobilizing people to undertake a concrete action" (Alvídrez & Franco-Rodríguez, 2016). Thus, we can understand Microsoft's desire to harness this power in an automatic way. Conversational understanding would be a move in that direction, albeit an unrealistic one given the limited ability of the bot. Alvidrez and Rodriguez indicate that people who use Twitter are trying to ascertain the credibility and the persuasive effect of the source by examining communication style, gender, and congruency of style (Alvídrez & Franco-Rodríguez, 2016). Tay had no way to ascertain the credibility of those who manipulated it and, therefore, was easily led to take on their antisocial rhetoric. It is beyond the scope of our paper to examine socio-linguistic theories in depth. However, if Microsoft were truly vested in conversational

understanding, on one hand, it might enquire how a bot emulating a human would convince people that it is a credible source of information, as "…credibility is a basic condition for persuading users of marketing web pages or information sources in social media" according to Shi, Messaris and Capella (2014). On the other hand, Tay might have avoided being "taught" objectionable speech if it were programmed to evaluate the credibility of its senders as well.

This is the approach taken by Candid (becandid.com) a social media platform that, rather than generate "speech", uses LS to classify posts as negative or positive statements and then give them a score. Posts that are beyond a cut-off are not posted (McEvers, 2016). This sort of approach seems to achieve the same sort of goals as Tay, without the breadth of risk. Nothing the LS generates is subjected to the public. It still learns a conversational understanding. Furthermore, since Candid is designed to read conversations from people, there is a chance that this LS will develop a conversational understanding of constructive discussion. Essentially, the people that use Candid will be aware of the chance that their posts might not meet muster and self-moderate to ensure that their posts are not culled. While this may be seen as a threat to free speech, the Candid LS can quite simply inform a poster of the apparent offensive nature of the post. The poster can either modify the language or, better, Candid could have an appeal process in place that allows developers to guide the learning of the LS based on borderline posts. In this way, Candid developers have a LS that has minimized its potential for harm. In contrast with Tay, large swaths of the public are not subject to the impact of its actions. Any harm is directed at a single individual and the broader societal implications are balanced.

Deception

Deception is commonplace, and ultimately software made for human interaction will have to include sophisticated approaches to recognizing deception in others, and perhaps deploying deceptions itself. Nonetheless, it is a tricky feature to get right ethically. In 2015, we wrote about deceptions in software:

> Our default position is that deception is unacceptable and that benign or beneficial deceptions are exceptions. Responsible developers should be required to make a strong case-by-case analysis of any deceptions they plan to implement and should justify why a particular deception should be an exception to the default prohibition. We believe that an appropriate policy framework should be developed and implemented now. We anticipate significant practical, ethical, and legal problems in the foreseeable future when AAs become increasingly human-like. Users "enchanted" by deceptive machines are likely to make inappropriate decisions based on these deceptions. Therefore, we recommend that developers acknowledge their responsibility to justify any deceptions they program into their artifacts (Grodzinsky, Miller, & Wolf, 2015).

On the one hand it is clear that Tay is not a human because Tay's Twitter account is clearly identified as belonging to a bot. Yet, Microsoft seemed to promote the idea that Tay was acting as a person in subtle, yet powerful ways. Tay's profile image was that of a young, white woman. Tay's header photo, although distorted and abstract, also contained hints of people's faces. Clearly, the developers were trying to have Tay act as if it were human. It is noteworthy that it requires special effort on the part of the user to see the text that says Tay is a bot. The suggestive profile image is by default attached to every tweet in a Twitter feed. At the very least, this is a case where an implicit deception was attempted: trying to make Tay behave in a way that was sufficiently human-like.

The Practice of the Computing Profession

Above we suggested that Microsoft did not practice appropriate professional diligence with Tay. A counter-argument to that claim might be that they shut Tay down very quickly. However, based on Lee's response, it is unclear how closely they were monitoring Tay. Human monitoring of Tay, especially in the first 24 hours it was online, is a minimum level of appropriate professional behavior, especially since this was where they "expected to learn more" (Lee, 2016). Such monitoring would have resulted in none of the offensive behavior being seen by the public. The attackers would not have known whether they had been successful in corrupting Tay. They likely would have thought Tay was impervious to such an attack. This sort of monitoring would have removed a large public stage on which people could misbehave.

A second concern that stems from Microsoft's reactive taking down of Tay is that it is unclear whether the takedown was a reaction to Tay's offensive tweets or a reaction to the Twittersphere objecting to the offensive tweets. Even though Lee says that Microsoft takes "full responsibility for not seeing this possibility ahead of time" (Lee, 2016), it is unclear whether they were the first to see Tay's tweets as being offensive. This raises the question of whether Microsoft was so committed to "learning more" that they left Tay up for as long as could and only decided to take Tay down when the external pressure made it apparent that it was necessary. This identifies important considerations for the development of appropriate professional best practice for internal processes when "releasing" LS to the general public. A web page to report a problem with the software is not sufficient. Closer monitoring by the developer is an ethical imperative for LS software that has contact with the public.

Traditionally, the in-house part of the development of software typically culminates in the testing of that code against some specification. This testing and subsequent fixing is done without interaction with the public. There are established best practices, and reputable firms adopt those practices, although with some inconsistency from firm to firm as to which protocols are in place and how software is deemed suitable for release to the public. When the software meets some level of closeness to the specification, it is released to the public. The important point here is that at the point of release, the code is fixed. It does not change. Because of that, at a certain level, there is a group of people

who understand the code well enough so that when an error occurs, they can at least offer some explanation that connects the error to the code.

This is not to say that the public are not involved with the development of this sort of software. In beta testing, people who use the software identify errors in the software that had escaped notice by the developers and testers. They report these problems back to the company and eventually many errors are fixed, and then the developer pushes patches back out to the users. Fixes are possible, due to the understanding that the developers have of code and the error in behavior that the code is exhibiting. While such a connection may not always be immediately apparent to the developers, they eventually determine such connections and can offer the sort of explanation mentioned above that leads to a thoughtful modification to the code that fixes the error and does not introduce new errors.

Simply migrating this model of interaction with the public to LS is not justifiable. We note that LS is different from traditional software in that the code that underlies the system is not static while the system is running. A typical implementation for LS includes a neural network. Learning systems dynamically make changes to their underlying code as they gain more input and take in responses to the output that they produce. It is unlikely that anyone on the development team has the same sort of understanding of the underlying code that is commonplace for developers of non-learning software. This lack of understanding is not due to the competence or professionalism of the developers. Rather, it is due to the fundamental nature of learning software that effectively precludes such knowledge. On the other hand, there are scholars who are beginning to address this problem by designing more sophisticated LSs that do have this capability, at least in a rudimentary sense (Lei, Barzilay, & Jaakkola, 2016). Another important and immediate ethical difference between LS and traditional software is that LS, and especially LS that is in contact with the general public, is not in widespread use, and we are in the early stages of developing our understanding of best practices. Thus, LS requires different practices that incorporate a more proactive ethical stance.

Further justification for a more proactive ethical stance in the development of LS is the research theme that underlies Microsoft's desire to learn more about Tay. Most software developers are not researchers in the traditional sense. Even those software developers with PhDs are rarely involved with research that involves people. Learning software puts developers at the forefront of experimentation and the use of social media (and other technologies such as robotics) makes people an integral part of the research aspect of the software development process. Clearly this sort of research calls for an expert in human subjects experimentation to be part of the development team. An additional difference is that historically, this sort of “pure” research was the purview of universities and government labs. While there were clear violations of ethical boundaries over the years, university and governmental researchers now abide by protocols that are in place to protect the research subjects and the public from potential ill effects of the research and the research methodology. Currently, at least in the U.S., there is little in the way of law and little public pressure for corporations to adopt similar protocols and practices in their

experimentation with technology, which increasingly interacts with people in significant ways.

A final concern with respect to the profession of computing that we reiterate here is that fixing errors in LS is no simple task. Neural networks are notoriously opaque. Their developers may not have a deep understanding of relationships between behavior and what in the underlying network manifested those behaviors. Even if those relationships were understood, it is unclear that we have the expertise needed to make instant modifications to neural networks to mitigate errors and change behaviors of the system.

Extraordinary Responsibilities of Learning Software Developers

In this section we suggest four imperatives for the developers of LS as they develop software that learns through directly engaging with the public. These imperatives are undergirded by Floridi's distributed morality (DM) in the context of multi-agent systems (MAS) in which actions "are assessed on the basis of their impact on the environment and its inhabitants" (Floridi, 2013). DM is called for in the cases under study here. The developers, the LS, and the trainers constitute the minimal set of agents that share some responsibility in the development of learning systems. Floridi's analysis includes the observation that most actions are morally neutral. However many of them have a potential bias toward being morally good or morally evil. He calls for the establishment of infra-ethics, "a first-order framework of implicit expectations, attitudes, and practices that *can* facilitate and promote morally good decisions and actions" (Floridi, 2013). His idea is to foster the development of ethical infrastructures that include moral aggregators, which tend to harness actions with a morally good bias, and moral fragmenters, which tend to isolate and neutralize actions with morally evil bias. The first imperative is that the initial learning environment must be controlled in some way so that it works to "aggregate good actions" and "fragment evil actions."

One might argue that the kind of experiment that Microsoft was attempting is a step in the right direction, a way to advance automated understanding of human language. However, like all good science, an experiment whose effects we are trying to understand needs a controlled environment with subjects who opt in as volunteers. Microsoft would have been better served if it had regulated the conversational input to their bot and then experimented with different variables and recorded the perceived changes and responses of Tay. That is responsible science, and while we may not like to think that social media experimentation as "science," it, at the very least, deserves the same checks and controls as any legitimate scientific experiment because it potentially affects a large demographic. If this "experiment" had been proposed in a university setting, it would likely have run afoul of an institutional review board, a body likely to object to the unpredictable responses of Tay. We support this objection. But our objection is not a reflection of a Luddite position; it is rather a rational insistence on responsible experimentation. Using the language of Floridi, we are calling for a system with a moral aggregator in order to promote morally good actions. We note that the case of Candid demonstrates a much

more responsible approach to developing the same sort of understanding of human language. It has an ethical infrastructure that isolates clearly morally evil statements. As its LS becomes more refined, it will isolate more morally neutral statements that have a bias toward moral evil.

The second imperative is that there needs to be better law and regulation to protect the public when a LS uses public involvement. While the Tay experiment is instructive, there can be more dire consequences when an LS interacts with the public. The recent case of Tesla Motors in which someone was killed due to the autopilot, which is not an LS, in a self-driving car failed to distinguish between the sky and a white tractor-trailer brings up an important consideration. In this case, the "driver" of the vehicle on autopilot who was killed had opted into the experiment (Rushkoff, 2016). On the other hand, the driver of the tractor-trailer had not. Even though the autopilot system was not an LS, Tesla was conducting an experiment on U.S. highways without the same sort of oversight that comes with an institutional review. While the National Highway Transportation Safety Administration had approved the use Tesla's autopilot on the highways, the level of scrutiny was likely not at the same level of concern for the general public as one finds in an institutional review. As Douglas Rushkoff reminds us, "As autonomous vehicle proponents like to point out, these problems would be solved if robotic cars weren't required to share the road with humans. We people are the problem" (Rushkoff, 2016). The concern here, as applied to LS development, is that the developers and the enthusiasts are too close to the decision making process about what constitutes suitable risk for the general public who do not know that "they are the problem" for an LS with which they may or may not be interacting. The development of something akin to institutional review boards for all LSs that interact with the general public is in order.

Our third imperative for LS developers is one of exceptional transparency and humility about what can be known about the LS. We have observed this imperative in academic literature throughout the years with emerging technology. In the 1980s, Joseph Weizenbaum called for caution (Weizenbaum, 1985) as did Bill Joy in 2000 (Joy, 2000). In 2003 Sheila Jasanoff called for "technologies of humility" in which shortcomings regarding the uncertainties and ambiguities about new technology are made plain (Jasanoff, 2003). It is now 2017. We have not made enough progress in the area of transparency. In order for a review board to do its work, it must have access to all the details about the system, but especially about what is not known about the system. While a certain level of confidence about the behavior of non-learning software is intellectually justified, the same cannot be said about LS. The wide range of possible behaviors makes a similar level of confidence ethically unjustified.

Finally, LS developers must put in place additional safeguards and testing procedures that are beyond those used for non-learning software. There must be more testing, more safety features, more filtering, and longer lead times before the impact of LS is experienced by anyone beyond the development process. As a technology, LS is less well understood and more unpredictable than other software. It demands an entirely new set of best practices for its development. These new best practices must become the infra-ethics for LS developers as quickly as possible.

Conclusion

Potential problems with the development and deployment of artificially intelligent machines have been foreseen for many years in both computer ethics literature and science fiction; yet, despite the warnings, best practices for the creation of the LS artefacts that interact directly with the public have yet to become prevalent. Our intent in this paper was to have a more important message than "we told you so." Collectively, developers and stakeholders alike need to learn from these incidents, and behave differently in the future. Floridi has a given us one possible mechanism to think through the implications of learning software and to structure the environments in which LS will operate in order to increase the likelihood of good ethical outcomes. Software developers must recognize software that is unpredictable is dangerous by design and take steps to limit its interaction with the public until it has been thoroughly tested in a controlled environment. Then, upon limited release, they should inform their customers, their users, and the general public not only of the advantages of software that evolve during its use, but also of the vulnerabilities introduced by unpredictable consequences for changed behavior. Then and only then can LS move forward in an ethically responsible manner.

References

Anderson, R. E., Engel, G., Gotterbarn, D., Hertlein, G. C., Hoffman, A., Jawer, B., … Rosenberg, R. S. (1992, October 16). ACM Code of Ethics and Professional Conduct. Retrieved July 26, 2016, from https://ethics.acm.org/code-of-ethics/

Alvídrez, S., & Franco-Rodríguez, O. (2016). Powerful Communication Style on Twitter: Effects on Credibility and Civic Participation. *Comunicar*, *24*(47). https://doi.org/10.3916/C47-2016-09

Floridi, L. (2013). Distributed Morality in an Information Society. *Science and Engineering Ethics*, *19*(3), 727–743. https://doi.org/10.1007/s11948-012-9413-4

Grodzinsky, F. S., Miller, K. W., & Wolf, M. J. (2008). *Ethics and Information Technology*, *10*(2–3), 115–121.

Grodzinsky, F. S., Miller, K. W., & Wolf, M. J. (2015). Developing Automated Deceptions and the Impact on Trust. *Philosophy & Technology*, *28*(1), 91–105. https://doi.org/10.1007/s13347-014-0158-7

Jasanoff, S. (2003). Technologies of Humility: Citizen Participation in Governing Science. *Minerva*, *41*(3), 223–244. https://doi.org/10.1023/A:1025557512320

Joy, B. (2000, April 1). Why the Future Doesn't Need Us. Retrieved July 13, 2017, from https://www.wired.com/2000/04/joy-2/

Dougherty, C. (2015, July 1). Google Photos Mistakenly Labels Black People "Gorillas." Retrieved October 7, 2016, from https://bits.blogs.nytimes.com/2015/07/01/google-photos-mistakenly-labels-black-people-gorillas/

Larson, S. (2016, March 25). Microsoft's racist robot and the problem with AI development. Retrieved March 27, 2016, from https://www.dailydot.com/debug/tay-racist-microsoft-twitter/

Lee, P. (2016, March 25). Learning from Tay's introduction. Retrieved July 27, 2016, from https://blogs.microsoft.com/blog/2016/03/25/learning-tays-introduction/

Lei, T., Barzilay, R., & Jaakkola, T. (2016). Rationalizing Neural Predictions. *arXiv:1606.04155 [Cs]*. Retrieved from http://arxiv.org/abs/1606.04155

McEvers, K. (2016, August 1). Can Candid Conversations Happen Online Without The Trolls? Retrieved February 8, 2016, from http://www.npr.org/sections/alltechconsidered/2016/08/01/488256587/can-candid-conversations-happen-online-without-the-trolls

Rushkoff, D. (2016, July 2). Tesla crash highlights real problem behind self-driving cars. Retrieved February 8, 2016, from http://edition.cnn.com/2016/07/01/opinions/tesla-self-driving-car-fatality-rushkoff

Seitz, D. (2016, March 24). How Microsoft's Twitter Experiment Became A Racist Nightmare. Retrieved March 27, 2016, from http://uproxx.com/technology/microsoft-tay/

Shi, R., Messaris, P., & Cappella, J. N. (2014). Effects of Online Comments on Smokers' Perception of Antismoking Public Service Announcements. *Journal of Computer-Mediated Communication*, *19*(4), 975–990. https://doi.org/10.1111/jcc4.12057

Victor, D. (2016, March 24). Microsoft Created a Twitter Bot to Learn From Users. It Quickly Became a Racist Jerk. - The New York Times. Retrieved August 27, 2016, from https://www.nytimes.com/2016/03/25/technology/microsoft-created-a-twitter-bot-to-learn-from-users-it-quickly-became-a-racist-jerk.html

Weizenbaum, J. (1985). *Computer power and human reason: from judgment to calculation*. San Francisco: Penguin Books.