Realität programmieren? Zum Einfluss von Algorithmen auf die Wirklichkeit

Pfeiffer, Jasmin
jasmin_pfeiffer@gmx.de
FAU Erlangen, Deutschland

Im Mai erregte ein Artikel der Bloggerin Lisa Ringen Aufsehen, der auf die Benachteiligung weiblicher Personen durch den Suchalgorithmus des Netzwerks Xing hinwies: Gibt ein potentieller Auftraggeber bei seiner Suche nach Freiberuflern nicht explizit die weibliche Form der entsprechenden Berufsbezeichnung ein, so werden ihm ausschließlich Profile von Männern in der Ergebnisliste angezeigt. Möchte man eine Freiberuflerin finden, so muss man etwa nach einer „Fotografin“ oder einer „Entwicklerin“ suchen – eine Idee, auf die vermutlich die wenigsten User kommen werden. Wie Ringen richtig schreibt, werden Frauen hierdurch nicht nur benachteiligt, sondern es entsteht auch „der unterschwellige Eindruck, Männer seien die erfolgreicheren, kompetenteren Fotografie-, Beratungs- und Grafik-Spezialisten“ (Ringen 2017). Bezeichnend ist das Statement des Xing-Sprechers Kopka, der, wie die SZ berichtete, betonte, dass ihm das Problem nicht bewusst gewesen sei (Holzki 2017).

Die Affäre um Xing legt ein Problem offen, das bisher noch unzureichend erforscht und thematisiert worden ist: Computerprogramme stellen keine neutralen mathematischen Gebilde dar, sondern sind, wie Lemire betont, zu wirkungsmächtigen Agenten im sozialen Raum geworden, die die uns umgebende Wirklichkeit beeinflussen und verändern können: „In any case, we have to accept software as an active agent that helps shape our views and our consumption rather than a mere passive tool.“ (Lemire 2016) Die Annäherung der Informationswissenschaften an die Disziplinen der Mathematik und der Naturwissenschaften, die u. a. im Begriff der MINT-Fächer evident wird, hat den Blick auf diese Formen der Einflussnahme verstellt – Tech-Größen wie Mark Zuckerberg und Alexander Nix preisen die angebliche Vorurteilslosigkeit und Neutralität der Algorithmen, und Unternehmen verkaufen die von ihnen entwickelten Computerprogramme als Meilensteine auf dem Weg zu einer neuen Objektivität der durch sie übernommenen Prozesse (vgl. Maschewski / Nosthoff 2017). Auch die Klassifikation von Programmiersprachen als formale Sprachen, die im Vergleich zu natürlichen Sprachen als präziser und eindeutiger betrachtet werden, verstärkt diesen Eindruck.

In Abgrenzung hierzu möchte ich in meinem Vortrag Programmiercode als deklarativen Sprechakt beschreiben und auf diese Weise den Blick auf die von Lemire diagnostizierte realitätskonstituierende Dimension von Computerprogrammen lenken. Ich werde mich dabei auf objektorientierte Sprachen konzentrieren, da der Bezug zwischen Code und Wirklichkeit bei diesen besonders evident ist.

In einem ersten Teil werde ich aufzeigen, inwiefern in objektorientierten Sprachen verfasster Programmcode als deklarativer Sprechakt beschrieben werden kann. Searle definiert Deklarationen als Sprechakte, die eine Übereinstimmung von Wirklichkeit und Worten herstellen: „It is the defining characteristic of this class that the successful performance of one of its members brings about the correspondence between the propositional content and reality; successful performance guarantees that the propositional content corresponds to the world.“ (Searle 1975: 358) Deklarative Sprechakte beschreiben folglich etwas, was durch diese Beschreibung zur Wirklichkeit wird.

Dies trifft auch auf in objektorientierten Sprachen verfassten Programmiercode zu: Er beschreibt Klassen, Instanzen und deren Eigenschaften sowie Methoden, die dadurch zugleich innerhalb des Codes existieren und, wenn das Programm kompiliert wird, die Ausführung der von ihnen beschriebenen Operationen in der Hardware des Computers veranlassen.

Jedoch bleibt der Bezug zur Wirklichkeit im objektorientierten Paradigma nicht auf die Erzeugung von elektronischen Spannungen in den Bauteilen beschränkt. Vielmehr wird objektorientierter Code im Software Engineering als Repräsentation der Wirklichkeit konzipiert. Joachim Goll und Cornelia Heinisch beispielsweise beschreiben Klassen in ihrer Java-Einführung als Entsprechungen von Gegenständen der realen Welt: „Eine […] Klasse entspricht einem Typ eines Gegenstands der realen Welt.“ (Goll / Heinisch 2014: 36 f.) Die Idee der objektorientierten Programmierung besteht laut Goll und Heinisch darin, realweltliche Entitäten in Repräsentationen im Programmcode zu überführen: „Der Ansatz der Objektorientierung basiert darauf, Objekte der realen Welt mit Hilfe softwaretechnischer Mittel als Entity-Objekte abzubilden.“ (Goll / Heinisch 2014: 37) Bei dieser Überführung werden nur die für das Computerprogramm relevanten Aspekte der realen Entitäten im Code dargestellt: „ Bei der Abbildung einer Entität der Realität auf ein Objekt muss aber eine Abstraktion stattfinden, bei der das Unwesentliche weggelassen wird. Die typischen Eigenschaften bleiben übrig. Die unwesentlichen Eigenschaften werden ignoriert (Abstraktion).“ (Goll / Heinisch 2014: 37)

Ziel des objektorientierten Programmierens ist also, Objekte und Zusammenhänge der Wirklichkeit in Klassen, Instanzen und Methoden im Code darzustellen. Diese Überführung stellt einen selektiven und damit notwendigerweise reduktionistischen Akt der Zurichtung von Wirklichkeit dar. Obgleich in den verschiedenen

Ratgebern zum Software-Engineering versucht wird, objektivierbare Kriterien für die Abbildung von Realität in Code zu entwickeln, wird diese letztlich immer vom Erfahrungshorizont und den *world versions* des zuständigen Software-Entwicklers beeinflusst. Die Ergebnisse der Operationen, die der Algorithmus auf den Repräsentationen der realen Entitäten im Code vollzieht, werden von den Benutzern des Programms potenziell in die Realität rückgeführt und können als Ausgangspunkt realer Handlungen fungieren. Dies zeigt sich deutlich am Beispiel des in Washington D.C. implementierten Systems IMPACT zur Evaluierung der Leistungen der Lehrer von Washingtons Schulen: Die Ergebnisse der vom Algorithmus durchgeführten Berechnungen dienten hier als Grundlage für die Entscheidung, welche Lehrer entlassen und welche befördert wurden. Die deklarativen Sprechakte des Codes verändern folglich, so die These, die Wirklichkeit nicht nur auf der Ebene der Einsen und Nullen, sondern können auch Einfluss auf Diskurse nehmen und somit, wie Lemire konstatiert, zu aktiven, die Wirklichkeit verändernden Agenten werden. Ich möchte den Akt des Programmierens daher ähnlich wie natürliche Sprechakte als eine Art Einschreibung in den Diskurs und als wirklichkeitsschaffend beschreiben.

Möchte man diese Formen der Einflussnahme auf die Realität verstehen und adäquat beschreiben, so ist eine genaue Analyse des den Computerprogrammen zugrunde liegenden Codes unabdingbar. Hier setzt die in den frühen 2000er-Jahren unter anderem von Matthew Fuller und Lev Manovich propagierte Strömung der Software Studies an, in der sich dieser Vortrag situieren möchte. Daher werde ich im zweiten Teil meines Vortrags ein *Close Reading* eines objektorientierten Algorithmus vorstellen. Hierzu werde ich ein Beispiel aus dem Bereich des Machine Learning wählen. Einige in der jüngeren Vergangenheit erschienene Paper machen darauf aufmerksam, dass bestehende kulturelle und soziale Meinungen Systeme künstlicher Intelligenz in starkem Maß beeinflussen können. Für den Bereich der *Structured prediction* beispielsweise haben Zhao, Wang u. a. gezeigt, dass die in den zum Training verwendeten Datenmengen enthaltenen Geschlechterstereotypen nicht nur vom Model übernommen, sondern sogar verstärkt wurden.

Was die Maschine lernt und welche Konzeption der Realität sie hat, ist folglich offensichtlich in starkem Maße beeinflusst von den zum Training verwendeten Datensets, aber auch vom Wirklichkeitsverständnis des Programmierers, der die ML-Algorithmen erschafft. Dies möchte ich in meinem Vortrag am Beispiel der Entscheidungsbäume näher erläutern, da diese einerseits leicht zu verstehen sind und andererseits die oben umrissenen Probleme besonders deutlich demonstrieren. Entscheidungsbäume liefern zu Objekten, die durch Mengen von Attribut-Wert-Paaren beschrieben sind, jeweils eine Entscheidung darüber, welcher Klasse das betreffende Objekt zuzuordnen ist. Welche Eigenschaften eine Klasse hat, wird dabei vom Programmierer festgelegt und somit unmittelbar von dessen Wirklichkeitsverständnis beeinflusst.

In Hinblick auf das Tagungs-Thema, „Kritik der digitalen Vernunft“, verfolgt der Vortrag zweierlei Ziele: Einerseits soll ein möglicher methodischer Ansatzpunkt zur Analyse von Programmiersprachen und Algorithmen entwickelt werden und aufgezeigt werden, dass die genaue Analyse von Code und seiner Funktionsweise unabdingbar für das Verständnis digitaler Phänomene ist. Andererseits soll zu einer gewissen Vorsicht gegenüber digitalen Methoden in den Geisteswissenschaften aufgerufen werden: Die Benutzung digitaler Mittel ist eine große Bereicherung für die Geisteswissenschaft, erfordert aber auch eine eingehende Reflexion der Beziehung zwischen Realität und Programmcode und des Einflusses der Algorithmen auf den betrachteten Gegenstand.

Bibliographie

Alpaydin, Ethan (2016): *Machine Learning*, Cambridge, Massachusetts: MIT Press.

Berry, David M. (2011): „The Computational Turn: Thinking about the Digital Humanities“, in: *Culture Machine* 12: 1–22.

Berry, David M. (2011): *The Philosophy of Software. Code and Mediation in the Digital Age* , New York: Palgrave Macmillan.

Fuller, Matthew (2008): *Software Studies. A Lexicon* , Cambridge, Massachusetts: MIT Press.

Holzki, Larissa (2017): „Frauen sind im Netz schwieriger zu finden“, http://www.sueddeutsche.de/karriere/frauen-und-karriere-frauen-sind-im-netz-schwieriger-zu-finden-1.3492732 , Stand: 24.09.17.

Lemire, Daniel (2016): „Is software a neutral agent?“, https://lemire.me/blog/2016/04/29/is-software-a-neutral-agent/ , Stand: 24.09.17.

Manovich, Lev (2001): *The Language of New Media* , Cambridge, Massachusetts: MIT Press.

Maschewski, Felix / Nosthoff, Anna-Verena (2017): „Das Netz ist nie neutral“, https://www.nzz.ch/feuilleton/kuenstliche-intelligenz-digitale-technik-ist-nie-neutral-ld.1302959 , Stand: 24.09.17.

O’Neil, Cathy (2016): *Weapons of Math Destruction* , New York: Broadway Books.

Ringen, Lisa (2017): „Dummer Algorithmus: Als Frau bei Xing gefunden werden“, https://www.marketing-madam.de/2017/05/02/dummer-algorithmus-als-frau-bei-xing-gefunden-werden/92706211/ , Stand: 24.09.17.

Searle, John R. (1975): *Expression and Meaning. Studies in the Theory of Speech Acts* . Cambridge: Cambridge University Press.

Zhao, Jieyu / Wang, Tainlu / Yatskar, Mark / Ordonez, Vicente / Chang, Kai-Wei (2017): „Men Also Like Shopping: Reducing Gender Bias Amplification using Corpus-level Constraints“, https://arxiv.org/abs/1707.09457 , Stand: 01.01.18.