DHQ: Digital Humanities Quarterly

Preview
2018
Volume 12 Number 1

SpotiBot — Turing Testing Spotify

Pelle Snickars <pelle_dot_snickars_at_umu_dot_se>, Umeå University
Roger Mähler <roger_dot_mahler_at_umu_dot_se>, Umeå University

Abstract

Even if digitized and born-digital audiovisual material today amounts to a steadily increasing body of data to work with and research, such media modalities are still relatively poorly represented in the field of DH. Streaming media is a case in point, and the purpose of this article is to provide some findings from an ongoing audio (and music) research project, that deals with experiments, interventions and the reverse engineering of Spotify's algorithms, aggregation procedures, and valuation strategies. One such research experiment, the SpotiBot intervention, was set up at Humlab, Umeå University. Via multiple bots running in parallel our idea was to examine if it is possible to provoke — or even undermine — the Spotify business model (based on the so called "30 second royalty rule"). Essentially, the experiment resembled a Turing test, where we asked ourselves what happens when — not if — streaming bots approximate human listener behavior in such a way that it becomes impossible to distinguish between a human and a machine. Implemented in the Python programming language, and using a web UI testing frameworks, our so called SpotiBot engine automated the Spotify web client by simulating user interaction within the web interface. The SpotiBot engine was instructed to play a single track repeatedly (both self-produced music and Abba's "Dancing Queen"), during less and more than 30 seconds, and with a fixed repetition scheme running from 100 to *n* times (simultaneously with different Spotify Free 'bot accounts'). Our bots also logged all results. In short, our bots demonstrated the ability (at least sometimes) to continuously play tracks, indicating that the Spotify business model can be tampered with. Using a single virtual machine — hidden behind only one proxy IP — the results of the intervention hence stipulate that it is possible to automatically play tracks for thousands of repetitions that exceeds the royalty rule.

Introduction

Digitized and born-digital audiovisual material today amounts to a steadily increasing body of data to research — and do research with [Brügger 2016]. Yet, other media modalities than text is still relatively poorly represented in the field of DH. As is well known, the focus on the modality of text is (and has remained) strong; textual scholarship has been the discipline's core concern — from Busa's concordances, to various text encoding initiatives and distant *reading*. Naturally, there are a number of exceptions, especially regarding visual culture, and the way that images as datasets have been researched within major frameworks as cultural analytics, or the abilities to algorithmically use computers to read and analyze a film's formal elements. Then again, most analytical DH-frameworks are derived from text, and it remains illustrative that as late as 2014, a special interest group focusing on "audiovisual material in digital humanities" was set up within ADHO. Its purpose is to expand the role of audiovisual media within DH, and serve as "a venue for exchanging knowledge, expertise, methods and tools by scholars who make use of audiovisual data" [AVinDH 2016]. 1

Following John F. Barber's claim, sound and music are especially "overlooked" within DH. The "tilization of sound as a methodology for information representation" has remained odd, or at best intriguing for digital humanities research [Barber 2016]. As of lately, however, there seems to be an increased scholarly DH-interest in the media modality of sound. As Erica Robles-Anderson and Patrik Svensson has argued in their excellent piece on Powerpoint in *Computational Culture*, oral presentation culture "is a powerful reminder that before Gutenberg much reading was listening". [Robles-Anderson and Svensson 2016]. Situated within a wider cultural analysis of audio collections, the first issue of the, *Journal of Cultural Analytics*, featured an intriguing article by Tanya Clement and Stephen McLaughlin on how to measure applause [Clement and McLaughlin 2016]. Clement has also published in this journal on the infrastructural analyses of sound collections within 2

the so called, HiPSTAS-project [Clement 2016]. In addition, two of our colleagues at Humlab are also pursuing research around sound, particularly regarding representational strategies of oral antiquity [Foka and Arvidsson 2016].

3 The purpose of this article is hence to make another contribution towards a greater variety of media modalities within DH-research. Our article provide some initial findings from an ongoing research project that deals with various experiments, interventions and the reverse engineering of Spotify's algorithms, aggregation procedures, and valuation strategies. The article focuses an experiment which we have called, the *SpotiBot intervention*. Via multiple bots running in parallel our idea was, in short, to examine if it is possible to provoke — or even undermine — the Spotify business model (based on the so called "30 second royalty rule"). The intervention resembled a Turing test, where we asked ourselves what happens when — not if — streaming bots approximate human listener behavior in such a way that it becomes impossible to distinguish between a human and a machine. As is well known, in the original Turing test the question was whether a *human* could distinguish between a person and a machine [Turing 1950], whereas in our case *Spotify's algorithms* were implicitly "asked" to make this distinction. Basically, the SpotiBot interventions we conducted hence approximated a repetitive Turing test — i.e. our bots interacted with the Spotify system, which (in one way or the other) tried to decide (via various unknown fraud detection tools) if communication was human or machine based.

4 While most previous scholarship on Spotify has primarily focused the service role within the music industry, its alterations of the digital music economy, or causing the eventual end of piracy [Wikström 2013] [Wikström and DeFilippi 2016] [Anderson 2015][Galuszka 2015] [Schwarz 2013], our research project on Spotify takes a software studies and DH approach towards streaming media. In general, the key idea is to "follow music files" — rather than the people making, using, collecting or listening to them — on their distributive journey through the Spotify streaming ecosystem [Fleischer and Snickars 2017].[1] Basically, our scholarly purpose is thus to draw a more holistic picture by using Spotify as a lens to explore social, technical, and economic processes associated with digital media distribution. Building on the tradition of "breaching experiments" in ethnomethodology [Garfinkel 1967] — where "reactions" are caused by disturbing or even violating commonly accepted rules or norms — our project has tried (in different ways) via repeated and modified interventions, to "break into" the hidden infrastructures of digital music distribution. The interventionist approach in many ways resembles the simple way a postman would follow the route of a parcel — from packaging to delivery.

5 The project also engages in reverse engineering Spotify's algorithms and aggregation procedures, all in order to study platform logics, including underlying norms and structures. As is well known, reverse engineering starts with the final product (the music service Spotify in our case) and tries to take it apart — backwards, step by step "seeking clues as to why it was put together in the way it was and how it fits into an overall architecture" [Gheel 2014, 10]. As an attempt to reveal the procedures of culture and technology at work, reverse engineering can be linked to various forms of hacking practices. Within studies of media reverse engineering has been used both by academic scholars [Friesinger and Herwig 2014] as well as by tech journalist wanting to understand and analyze, for example, how Netflix's sorting algorithms, vocabulary and grammar work [Madrigal 2014].

6 On the one hand, we have within our project been interested in broadly studying different data patterns and media processes at Spotify. On the other hand, we have also been keen on producing and obtaining research data, for example by documenting (and tracing) Spotify's history through constantly changing interfaces, or by tracking and archiving advertisement flows (through debugging software as Fiddler or Ghostery). One point of departure is that Spotify resembles a black boxed service, metaphorically as well as practically (at least from an academic media studies perspective). Another, is that Spotify does not — to put it bluntly — share any data. Lack of access to data today confronts both media scholars, (digital) humanists and social science researchers working within media studies. As a consequence, since Spotify user data is not available, it has had to be acquired and compiled through other means in order to perform research — for example by deploying bots as research informants [Eriksson 2018 (forthcoming)].

Bots as Informants

7 Bots *appear* to be human — which is why they are interesting. Bots give an impression of being able to act as a normal user and/or person. The computer scientist Joseph Weizenbaum programmed what is often regarded as the first bot, ELIZA in the 1960s, which could then (almost) pass for human. Today, the possibilities of such intelligent machines (or rather *software robots*) have naturally increased [Boshmaf et al. 2011]. The most sophisticated contemporary bots react instantly to public information, like the advanced algorithmic bots on the stock option market. They seems almost like disembodied cyborgs, part human and part automaton.

The bots we have programmed in our research project, however, are far from cyborgs. Our so called “BOT step-by-step behavior-scheme” rather testifies to rudimentary tasks like: “BOT requests access to play.spotify.com; BOT enters credentials; BOT submits credentials; Read current play position; BOT waits until search link is clickable; Enter search string that identifies track to play”, etcetera. Still, our bots have proven to be *very* useful in order to explore, investigate, mimic, and (even) subvert Spotify’s notions of usage and listening. 8

Under the computational hood of streaming music services all streams are equal, yet unlike at Apple Music every stream at Spotify Free means (potentially) increased revenue from advertisers. The ad supported streaming model — giving away music for free — has been controversial (especially in the U.S.). Yet, it has enabled Spotify to acquire a lot of customers, and the company’s conversion rate (to Premium subscription) lies around 25 percent. Established in Sweden in 2006, Spotify is today the dominant player in the streaming music market. With a user base now officially reaching more than 100 million — including 50 million paying subscribers — the music streaming service is today widely recognized as the solution to problems caused by recent decades of digital disruption within the music and media industries. Spotify resembles Netflix, YouTube and Apple Music as an epitome of streaming’s digital zeitgeist envisioned to shape our future, and has during the last years emerged as *the giant* within the streaming music business. Even outcompeting Apple Music, Spotify’s market share lies above 40 percent. 9

One success factor is arguably the ease by which a listening account can be set up at Spotify. For a number of years it was, for example, extremely simple to sign in to Spotify — both for humans and bots. In endless discussions with record labels (around rights management) Spotify took the stance that the continuous offering of a zero-price version with recurrent advertisement (Spotify Free) would in the long run be the best solution as well as incentive to *scale* businesses and attract global listeners. The importance of scaling and constantly adding new listeners might hence be one reason for Spotify’s low security thresholds. There is no way of knowing how many fake accounts are registered among the service’s 100 million ‘users’. No CAPTCHA was for example needed (for our bots) when we performed the interventions and data capture for this article (during late spring 2016), and obviously, one should in this context remember what the CAPTCHA abbreviation originally stands for: “Completely Automated Public Turing test to tell Computers and Humans Apart”. 10

As is well known, a CAPTCHA is a program that protects websites against bots by generating a simple test that humans can pass but computer programs cannot. Given that different forms of *click fraud* — with automated bots pretending to be consumers — has been an increasing problem within the online advertising industry, the lack of CAPTCHA’s at Spotify did trigger discussions on the community blog, for example as a way to stop bot fake listenings. In March 2016 user Jdwhicker asked if there was an “issue with fraudulent plays going on. What if non-premium accounts had to pass a captcha instead of listen to an advertisement every so often?” [Jdwhicker 2016]. As a possible consequence, during late summer of 2016, Spotify started using CAPTCHAs as well as reCAPTCHAs (with image identification) — allegedly to better protect their system. 11

Prior to these altered security measures, however, within our research project we used and deployed multiple bots to study the explicit and implicit logics of the Spotify web client. We were even been able to automate account registration (for fast bot setups). In one intervention, for instance, we deployed 288 bots, within 48 parallel Spotify sessions. Departing from the idea that software is normative and regulatory — and, hence, that the streaming architecture of Spotify promotes and materializes certain world views (and not others) — our bots have been programmed to explore (and to some extent disrupt) the socio-technical protocols that endow music files with cultural meaning. 12

While setting up our different experiments and interventions, we have asked ourselves questions like: How are music files repeatedly recontextualized and rearranged by Spotify? How many steps does ‘raw’ audio data have to take before it can become a streamed listening experience? How is the social life of a file imagined (and fostered) by this system? Another fundamental question we have asked ourselves is what sounds are actually perceived as music (or not) according to Spotify, and at various adjacent music aggregating services that regulate content appearing on streaming platforms [Morris and Powers 2015]. Bots, in short cannot only “listen” to music; they can also produce content (in the form of “music”). Our interventionist methods and explorations with uploading (more or less artificial) sounds and music have, for example, resulted in different responses. The same music (or sounds) pass some aggregators — while others define it not to be music content at all. When principles as to what is considered music vary, and when rejection criteria at music aggregators turn more or less arbitrary — usually depending on whether users pay an aggregation fee or not — the line between music and non-music, artist and machine, human and bot becomes increasingly blurred. 13

14

In general, and arguably due to the fact that more music is usually better music at Spotify, the service seems likely to include — rather than reject — various forms of (semi-)automated music [Snickars 2016]. Even though the service differs from open platforms such as SoundCloud or YouTube, it is definitively ajar to (un)intentional and/or calculated musical t(h)inkering, whether brought about by humans or bots. In short, what is labeled as *music* on Spotify is quite arbitrary — machines can for example produce sounds and aggregate these. Then again, what is labeled as a *listener* on Spotify is also random — our bots have as a matter of fact "listened" to quite a lot of music.

15 Another major research issue we have struggled with is the type of knowledge that can be gained (and gleaned) from working with bots as informants. Are they to be trusted? Can they produce valid empirical data to be used in, for example, scholarly publications? We do think so. Yet, as a kind of virtual informants our bots do not interactively and explicitly *collect* information, rather they have been designed and set up to acquire and log certain data via the "actions" and different functions they have been programmed to perform. Nearly all of our bots have been Spotify Free users. The setup has involved a few similar steps. Firstly, bots are named (selenium44, selenium45, radon17, radion18 etcetera) and each given certain specified (or random) characteristics (age, nationality, gender etcetera). Secondly, the bots — in the form of virtual users — have been programmed to do specific tasks and hence act as research informants within the Spotify web client. Within our research project these have varied depending on what scholarly issues or tasks we have been interested in. Some bots have been used to massively play a single track repeatedly (as the many SpotiBots), others have been programmed to test music recommendations based on gender (or age), and a third category have been deployed to research so called "radio looping", i.e the algorithmic song recommendations that follow automatically when (in our case) a bot picks a song and starts a "radio channel" within the Spotify web client [Snickars 2017].

16 A typical "bot experiment description" (from the Spotify Radio intervention above) gives a hint as to the ways we have been trying to design our bots as *active research informants*. Usually our bot experiments have been iterative. Importantly, all bots have been instructed to *log* all their programmable actions and outcomes. Sometimes we have also recorded their actions on video.

```
All radio looping bots sign into the web client and starts Spotify's radio
function – based on the following specifications:
Bot1 (the obedient listener): Starts a radio station based on Abba's "Dancing Queen"
and passively listens to the full loop. Run time 12 hours.
If the radio loop stops playing, the bot should be prepared to restart.
Bot2 (the skipper): Starts a radio station based on Abba's "Dancing Queen"
and skips every fifth song. Run time 12 hours.
If the radio loop stops playing, the bot should be prepared to restart.
Bot3 (the liker): Starts a radio station based on Abba's "Dancing Queen"
and likes every fifth song. Run time 12 hours.
If the radio loop stops playing, the bot should be prepared to restart.
Bot4 (the disliker): Starts a radio station based on Abba's "Dancing Queen"
and dislikes every fifth song. Run time 12 hours.
If the radio loop stops playing, the bot should be prepared to restart.
```

17 It is sometimes said that when bots can pass for humans in a conversation — it will be a milestone in artificial intelligence. "We live in a world of bots", a recently published "botifesto" furthermore stated. Bots — in the form of sets of algorithms — this botifesto argues, are both responsible for much of what happens at "the backend of the internet", as well "as playing a more active role in our everyday lives" [Woolley et al. 2016]. Automated systems' increasing (in)ability to understand humans, in short, point to the fact that bots are in vogue. It has even been suggested that the obsession with bots is driven by a perceived fatigue with apps. Developers are hence looking for new ways to reach consumers — Facebook has, for example, decided that various businesses are allowed to deliver automated customer support through chatbots within their messaging application Messenger. Given that Spotify's data exchanges with Facebook go back to 2011, it should hence come as no surprise that in April 2017 Spotify announced that it was now "easier than ever to share and discover music within Facebook Messenger with the new Spotify bot for Messenger with the all-new Chat Extensions feature" [Spotify Team 2017].

18

Acting as informants our research bots can be perceived as a similar kind of virtual assistants. In a way they resemble the digital assistants that Silicon Valley currently seems to favor — with the hope that AI-powered bot assistants as Siri, Alexa, Cortana etcetera "will manage more and more of our digital activities", as the hype goes [Newton 2016]. We are, naturally, aware of that bots are today programmed for a variety of reasons. They send news articles to people (and write some of these too); bots can be used to manipulate likes and followers; the bot-friendly design of Twitter differs from the regulated one on Facebook (even if that is likely to change), and *public service bots* as for example @earthquakeBot — "I am a robot that tweets about any earthquakes 5.0 or greater as they happen" — informs about important geological circumstances.

19 Even if some Twitter bots can search the Web for information (as well as post collected material at predetermined times), bots are not usually programmed to act as informants *per se* — and especially not in a music environment. Naturally, there are exceptions: so called *political bots* are for example increasingly used for ideological purposes to gather information, change opinion, and enhance specific political objectives (for good, ill, or in-between). It has even been argued that bots could become the "go-to mode for negative campaigning in the age of social media" [Woolley and Howard 2016]. In addition, *undercover bots* have for years been deployed to act as "insiders" when security firms have tried to outsmart hackers, literally working as semi-automatic informants. Bots have also increasingly started to get integrated into various forms of cloud based team collaboration tools, as for example Slack. Such *collaborative bots* facilitate and assist conversation. Still, the difference between bot users and regular users is of course that instead of interacting with a team via one of Slack's apps, bot users are controlled programmatically via a bot user token.

20 The bots described above are all sophisticated ones. Yet, given the increasing interest in bots and research around them [Woolley 2016] [Davis et al. 2016] [Abokhodair et al. 2015], we have refrained from simulating and building *really* human-like bots. If a "social bot" is a computer algorithm that automatically "produces content and interacts with humans on social media, trying to emulate and possibly alter their behavior" [Ferrara et al. 2016] — then our bots run on simpler versions of code. Then again, it remains to be stressed that even if our bots are not the most sophisticated on the market (or in the academy), experiments and interventions with them have been informed by profound considerations around bot culture and its subsequent implications.

21 Essentially, the bots we have programmed are scripted algorithms that exhibit human-like behavior (in one way or the other) when "listening" to music. Implemented in the Python programming language, and using a web UI testing frameworks, our SpotiBot engine has been able to automate the Spotify web client by simulating user interaction within the web interface. In the implementation to conduct experiments with the Spotify web client we have used a system framework originally designed for automated tests of web pages. Normally its purpose has been to validate correct behavior of software. Consequently, our bots have been designed to program user activities and — importantly — log and record output of these (i.e. customize scheduling, actions and logging of outcomes). Initially, our research project designed and developed a rudimentary virtual machine (or run time engine) with the capability to execute about 15 high level machine instructions such as "register_account", "login", "logout", "play_media", "follow_artist" that targeted a streaming media service provider (Spotify). The selected instruction set hence corresponded to the most common user interactions (in a web interface), and the set has also included primitives for data capture (screenshots, playlists, video), as well as loops and conditional execution. Using these instructions as buildings blocks, we have been able to design various user scenarios (case studies). In short, given a sequence of instructions, the virtual machine can execute them in turn, until an end condition is reached. The selected instruction set has thus been generic enough to, in theory, target basically any streaming service provider; that is, the machine uses a streaming service driver and abstracts, as well as implements each instruction for targeted services (Spotify, YouTube etcetera). Importantly, the machine also includes features for management of timed and repeated executions (of a case study), a reporting system to view results of executions, as well as features for monitoring ongoing executions. Finally, our virtual machine has used the popular Web testing framework, Selenium for access and control of various web browsers.

SpotiBot — Results and Anomalies

22 One of the major controversies regarding the transition to streaming music platforms involves payouts to artists. The sometimes heated discussion has, in short, been centered around the issue *if* streaming music will be able to generate a sustainable income for musicians — or not. Statistics vary (and are often very hard to find), but estimations usually state that revenue per played track range from $0,0003 to $0,0013 at streaming services as Spotify, Apple Music or Deezer. Within the music industry this has led to considerable debate; Taylor Swift decided to remove her entire back catalogue from Spotify, and lesser known artists have experimented with different music-hacks or pranks. These have ranged from the funk

band Vulfpeck, and their conceptual album, “Sleepify” — containing five minutes and 16 seconds of pure silence; asking fans to stream the album on repeat (while sleeping) — and hacks by the band Ohm & Sport and their application Eternify, were for a (very) short time one could enter the name of a favorite artist and play songs on repeat for economic support in 31-second intervals — to the music spammer Matt Farley, who has personally released over 15,000 songs.

23 At Humlab (Umeå University) we therefore set up an intervention — SpotiBot — with the purpose to determine if it was possible to provoke, or to some extent undermine, the Spotify business model in a similar manner. Confidential agreements and record label contracts with Spotify vary, but royalties are usually disbursed to artists (or more precisely, record labels) once a song or track is *registered* as a play, which happens after 30 seconds. The SpotiBot engine — with the ability to run a multiple of pre-programmed user bots — was hence instructed to play a single track repeatedly for more and less than 30 seconds, and sometimes simultaneously with different accounts. Tracks consisted both of self-produced music (from our research project; the artist Fru Kost with the song “Avplock”) and Abba’s “Dancing Queen”. The fixed repetition scheme ran from 100 to *n* times. The first round of bot experiments played Fru Kost, the second round played Abba, and the third round played Abba again — but with increasing multiple bots at the same time. The SpotiBot engine was, in short, programmed to automate the Spotify web client by simulating bot interaction within the interface in the following way:

```
1.1     SpotiBots plays Fru Kost “Avplock” for 25 seconds – repeat 100 times.
1.2     SpotiBots plays Fru Kost “Avplock” for 35 seconds – repeat 100 times.
1.3     SpotiBots plays Fru Kost “Avplock” for 35 seconds – repeat 1,000 times.
1.4     10 SpotiBots plays Fru Kost “Avplock” for 35 seconds – repeat n times ...

2.1     SpotiBots plays Abba “Dancing Queen” for 25 seconds – repeat 100 times.
2.2     SpotiBots plays Abba “Dancing Queen” for 35 seconds – repeat 100 times.
2.3     SpotiBots plays Abba “Dancing Queen” for 35 seconds – repeat 1,000 times.

3.1     10 SpotiBots plays Abba “Dancing Queen” for 25 seconds – repeat 100 times.
3.2     10 SpotiBots plays Abba “Dancing Queen” for 35 seconds – repeat 100 times.
3.3     100 SpotiBots plays Abba “Dancing Queen” for 35 seconds – repeat 100 times.
3.4     1,000 SpotiBots plays Abba “Dancing Queen” for 35 seconds – repeat n times ...
```

24 From a computational perspective the Spotify web client appeared as a black box; the logics that the Spotify application was governed by was, for example, not known in advance, and the web page structure (in HTML) and client side scripting quite complex. It was not doable within the experiment to gain a fuller understanding of the dialogue between the client and the server. As a consequence, the development of the SpotiBot-experiments was (to some extent) based on ‘trial and error’ to find out how the client behaved, what kind of data was sent from the server for different user actions etcetera. In addition, since our bot experiments violated (some of) Spotify’s user agreements, a VPN connection was used that hid the running clients behind a public proxy IP outside of the university network. The selected VPN proxy service was very cheap (used by private consumers) and the stability of the connection was not entirely predictable.

25 Figure 1 and Figure 2 give a graphic estimation of some results from our experiments. In the first one, the SpotiBot engine was able to play Fru Kost’s track, repeatedly for 25 and 35 seconds. The bot “selenium57” for example played the track 229 times, and the “selenium_bot” as many as 1,141 times repeatedly — that is, after 35 seconds of “Avplock”, the bot started the song again, and again, and again (i. e. more than eleven hundred times in a row). Similarly, in the second experiment, “selenium_bot37” was able to repeatedly play Abba’s “Dancing Queen” for 35 seconds, at repeated intervals of 16 times, 208, 30, 1,141, 19, 20 times etcetera — to repeat: after 35 seconds of “Dancing Queen”, the bot started the song again, and again, and again.

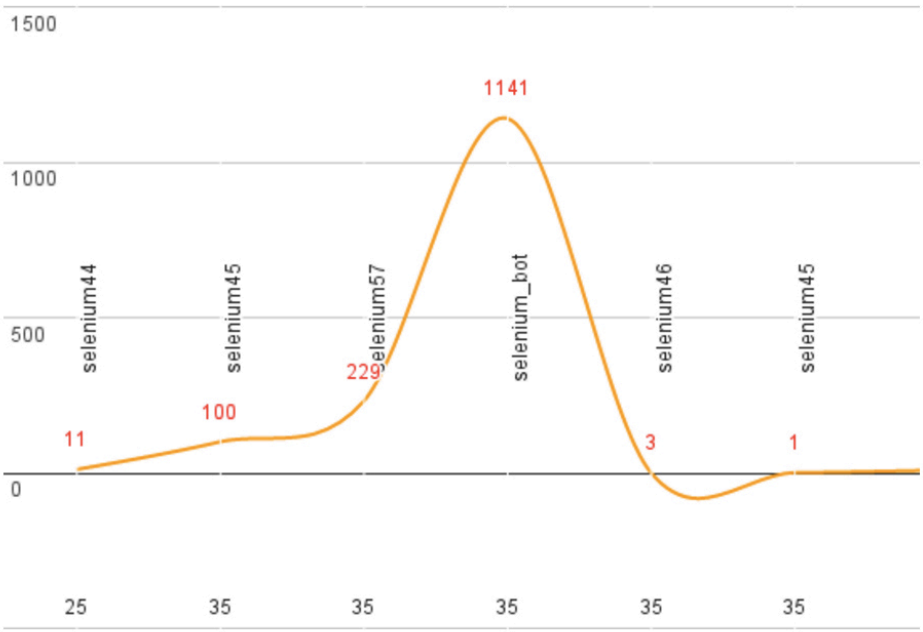

Figure 1. Different SpotiBots playing Fru Kost “Avplock”, repeatedly for 25 or 35 seconds (never mind negative results).

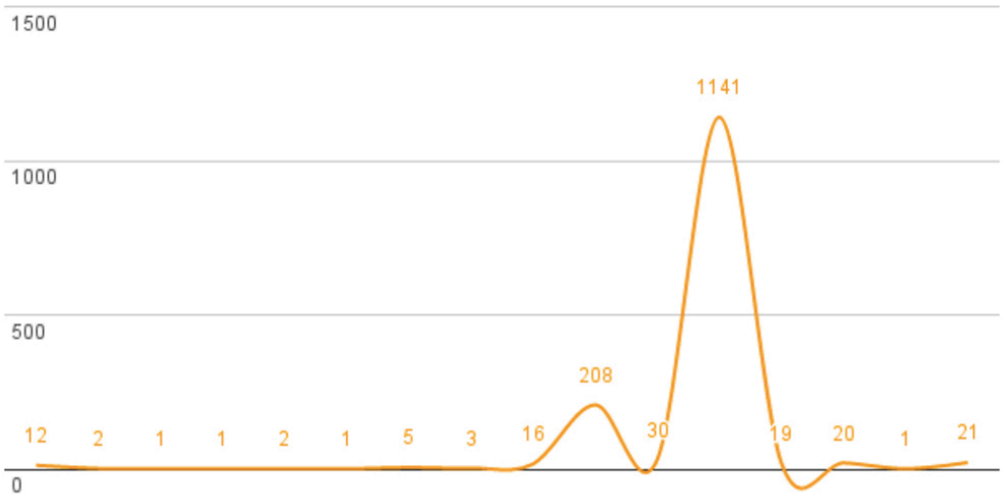

Figure 2. “selenium_bot37” repeatedly plays Abba’s “Dancing Queen”” for 35 seconds — with recurrent and (often) unknown Chrome browser errors (never mind negative results).

Apart from the possibility of *actually* being able to automatically (and repeatedly) play tracks on Spotify via bots, one preliminary result indicate that there was no major difference between our bots playing artist like Fru Kost *or* Abba for 25 *or* 35 seconds. For the Spotify system, both artists were simply content. Our hypothesis was that playing tracks repeatedly for 25 seconds would (at least in theory) not be a problem, since such plays are not regarded as a registered play by the Spotify system. This was also true. On one occasion the bot “selenium51” played Abba 550 repeated times for 25 seconds, and on another the “selenium_bot” managed to play Fru Kost 100 repeated times for 25 seconds. Then again, we did not discover *any* statistical difference if the same songs were played repeatedly during 35 seconds — even if they were then registered as a play (and subsequent royalties were registered). 26

A second preliminary result was that our third experiment (using a large number of parallel bots to play Abba’s “Dancing Queen”) could not really be executed on available hardware since it would have required investment in a lot of new 27

machines. In a lighter setup, however, we used more than 20 bots running in parallel on two computers (virtual machines) each interacting with a Spotify web client, repeatedly playing the same Abba track hundreds of times. Chart 3. gives a graphic estimation of some of the results. In theory (if we had the financial abilities) we definitively believe it would be possible to perform a massive "bot setup" with hundreds of clients running in parallel on a larger number of (possibly cloud based) servers. Our simple experiments, in fact, clearly indicates the ease in which hundreds of bots can be setup in parallel. In another experiment at Humlab we have, for example, set up 50 parallel bots using five virtual machines (Windows 10 clients) on two computers running Microsoft Hyper-V server (in total ten virtual machines running five bots each). It is easy to extrapolate this to hundreds, perhaps even thousands, of bots running in parallel with additional hardware, or the use of leaner clients, cloud services or distributed computing efforts.

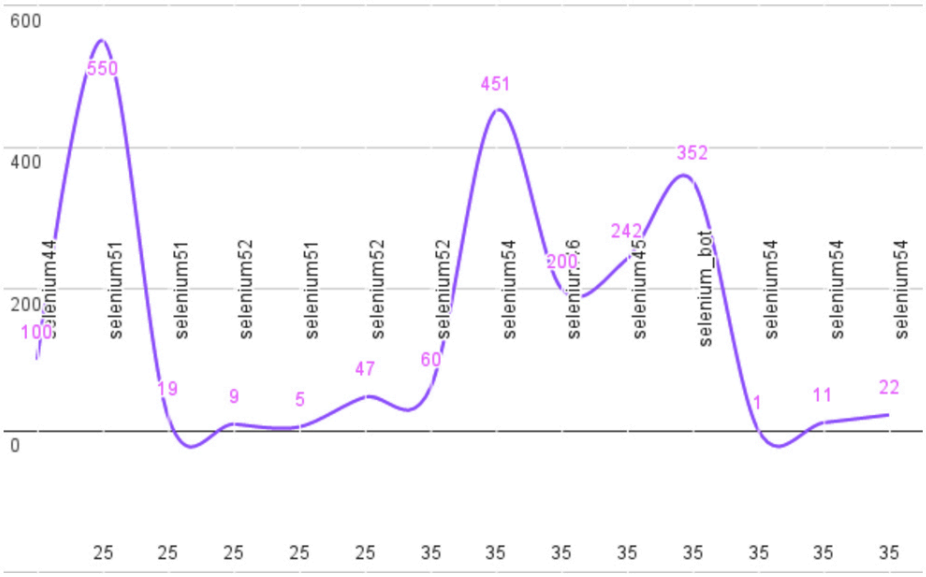

Figure 3. Multiple SpotiBots playing ABBA's "Dancing Queen", simultaneously and repeatedly for 25 or 35 seconds (never mind negative results)

In fact, this type of massive music hack (seems to) have been executed previously. In an article on Motherboard.com, the music journalist William Bedell, for example describes how he decided to "prototype a robot with an endless appetite for music to see if Spotify could detect what it was doing". His aim was to program a botnet on his "old laptop to generate $30 a day in fake Spotify listens"" — and he claims to have succeeded (even if there is no documentation of how he actually coded his bots). Automated streaming, following Bedell, could thus be seen as "a lucrative heist involving robots emulating humans". Performing his hack he did not "encounter many Turing tests … There wasn't even a CAPTCHA or email verification when creating accounts". His conclusion was hence similar to ours: "The barriers to entry are clearly minimal." [Bedell 2015] 28

Bedell's hack resembles a similar one made in 2013 by Peter Filmore. As a payments security expert, Filmore wanted to test the robustness of music-streaming services, and particularly if they had any fraud detection systems in place — which it turned out, they hadn't. Apparently, Fillmore made around a thousand dollar (he claims he was repeatedly playing his own "music") on the streaming service Rdio. One of his conclusions was that many music streaming services lacked automated analysis regarding suspected fraudulent plays [Fiveash 2013]. 29

Basically, the results from our experiments with the SpotiBot engine, confirm Bedell's and Filmore's hacks. The defense mechanisms used by Spotify to prevent our experiments have been insufficient — or remained unknown. In fact, it has proven more or less impossible in advance to predict how, or when, different kind of fraud detection systems have been activated (or not). It is worth stressing, however, that our bots are more advanced than the ones Bedell and Filmore programmed. They appear to have been "fixed programmed" with the purpose to *only* play songs — not register and log any outcomes, nor "interact" with a web client or be able to perform different tasks. Filmore's bots were programmed in Bash, a command-line interface in Linux — indeed scalable, yet without any form of web interface interaction. 30

31 One of the core assumptions in our research project has been to use bots as informants in multiple ways. As a consequence, they have been designed as “programmable bots” with the ability to receive instructions to perform different task (depending on the purpose of the intervention or experiment). Then again, even if our programmable bots were arguably more sophisticated than Bedell’s and Filmore’s, an important result from our SpotiBot interventions is still that a huge number of deviations did interrupt the Spotify web client, causing a number of our bots to stop playing. The SpotiBot setup was based on an *ideal* bot usage flow, and *all* deviations would in practice interrupt the client execution. This is the main reason (we believe) why so many of our bots did not perform the exact amount of repeated plays they were programmed to perform. If truth be told, nearly *all* of our bots — within the three rounds of experiments — stopped at random occasions. Most frequently deviations were caused by wrong behavior by the bot, due to lack of knowledge of client logic. A lot of the interrupts were, in addition, caused by synchronization problems, where a bot tried to access parts of the user interface not yet loaded (or not yet visible). A number of fixes for synchronization issues were, consequently, added to various actions in order to compensate for waits or components/features to become available. Still, even though the used framework, Selenium, had a good support for these kind of errors, the software didn’t always behave as we expected. Bot ‘clicking’ on a component not visible on the screen (that is, not within the scroll region) caused a number of errors with the Chrome browser, for instance — something that would almost never happen for a *real* user since it would be impossible to click outside a visible window area. Other problems were even harder to counteract, as for example network problems, performance problems related to used hardware, or unexpected resource usage of the Spotify web clients.

32 Our setup used the same IP for all clients which also made it difficult to determine if interruptions were caused by the actual setup, or by high volume originating from the same IP. Given more resources, it would however (in theory) be simple to use multiple VPN proxies to distribute the net load over several IP’s. Some Spotify web client interrupts were also (most likely) caused by network problems at the VPN service provider. The client VPN, in short, sometimes refused to reconnect automatically after disconnects, which caused the SpotiBots to halt execution. Furthermore, a single virtual machine (under Microsoft-V system) was used for all of our SpotiBot experiments. The VM ran in a shared environment with several other virtual machine on a host machine running Microsoft Server 2012. Some interrupts of the experiments were caused by inappropriate system configuration (automated software updates on both host server and client), system maintenance that unexpectedly disconnected the VPN, a couple of power outages (badly configured UPS), as well as limited resources (physical memory) assigned to the client virtual machine.

33 To sum up our problems, the bots we programmed to listen to the Spotify web client were not only disobedient — they were inserted into a non-compliant system full of latent errors. As a consequence, our SpotiBots stopped “listening” on (in)numerous occasions. This led to a situation where much more supervision and program correction than anticipated was needed. In addition, lots of interrupts in our experiments occurred off hour, when the SpotiBot engine was without supervision at Humlab. It caused repeated restarts of interventions as well as delays in the execution. All in all, our bots weren’t really ‘battle-tested’ before our experiments began, and a lot of bugs were constantly found that needed fixing. Certain aspects of the bots logic also needed fine-tuning, most often related to waits for element to be present, visible, clickable etcetera. Bots focused on the single task of playing music could have been designed in much simpler way like Bedell’s and Filmore’s bots. But we kept our design — mainly due to the data capture features that was a crucial part of the “bot tasks”. In later experiments and interventions within our project, however, our bots have shown a high level of stability.

Conclusion

34 It has sometimes been argued that digitization “drives botification”, where the use of technology in “a realm of human activity enables the creation of software to act in lieu of humans”. When bots become sufficiently sophisticated and numerous, as well as embedded within the systems within which they operate, “these automated scripts can significantly shape” human systems [Hwang et al. 2012]. These are considerations and reflections around the relationship between man and machine worth thinking twice about. Within our project we have, for example, repeatedly asked ourselves if aggregated “music” or “listeners” at Spotify “drive botification” as some have assumed — and if so: how do these automated scripts feedback and reshape listening behaviors? We have basically, deployed our bots in a *single* manner — yet, naturally, a more intricate situation would occur if one started looking at our bots *collectively*, and in a more systematic manner. That is, not as isolated units of code, but as a kind of *complex of bots*, a botnet “acting” together within the Spotify environment, shaping and tuning the system, evolving and transforming over time. The relation between listener and system in a streaming music environment as Spotify is hence complicated — not the least due to the vast amount of available music. Estimations vary, but roughly one-fifth of Spotify’s catalogue of some 30 million songs haven’t once been listened to by

anyone (neither man, nor machine). In fact, the main reason for purchasing manipulated *bot promotion* in the form of fake likes, followers or listeners are due to Spotify's swelling back catalogue — of unheard music.

35

Working with bots as research informants, this article has suggested a set of concrete methodologies for performing humanist inquiry on black-boxed media services (as Spotify) that today increasingly serve as key delivery mechanisms for cultural materials. Essentially, we used uncomplicated bots in order to test a streaming music environment — again, one of the used system frameworks (Selenium) originally deals with automations of web pages. Yet, even if we encountered a number of problems within our SpotiBot intervention, our bots demonstrated an ability to continuously play tracks, indicating that the Spotify business model can indeed be tampered with. One finding from our intervention is hence that that Spotify's algorithms (at least at the time of our experiment) could not tell human and bot listening apart. Later experiments within our research project — for example around measurements of loop patterns and repetitiveness on Spotify Radio [Snickars 2017] — has also gained a lot of insights from the ironing out of bugs within the SpotiBot intervention, which consequently has made our bot engine much more stable. Even if we encountered a number of problems and random deviations that interrupted client execution, the general results from our SpotiBot intervention do indicate that it is possible to automatically play tracks for thousands of repetitions that exceeds the royalty rule at Spotify. Admittedly, a more robust setup, with frequent and repeated testing — based on increased knowledge around client logic — would have made our bots less disobedient, and more successful in their listening habits. Still, since our experiments do raise a number of research ethical issues [Eriksson 2018 (forthcoming)], such a resilient intervention infrastructure would all likely have increased our hesitations around musical manipulation. In the end, Abba become a tiny fraction wealthier — and we also made (a very, very small amount of) money ourselves. Research can be rewarding.

Notes

[1] The research project, "Streaming Heritage. Following Files in Digital Music Distribution" is funded by the Swedish Research Council between 2014 and 2018. It involves system developers Roger Mähler and Johan von Boer (at Humlab, Umeå University), as well as researchers Pelle Snickars, Maria Eriksson, Anna Johansson and Rasmus Fleischer (at Umeå University), and Patrick Vonderau (at Stockholm University). For more information: http://streamingheritage.se/.

Works Cited

AVinDH 2016 "AVinDH SIG | Special Interest Group AudioVisual material in Digital Humanities". Available at: https://avindhsig.wordpress.com/ [Accessed 20 April 2017].

Abokhodair et al. 2015 N. Abokhodair, D. Yoo, & D.W. McDonald. Dissecting a social botnet: Growth, content and influence in Twitter. *CSCW '15: Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, 839–851.

Anderson 2015 Anderson, P.A. "Neo-Muzak and the Business of Mood". *Critical Inquiry* 41(4), 811-840.

Barber 2016 Barber, J.F."Sound and Digital Humanities: reflecting on a DHSI course." *Digital Humanities Quarterly* 10.1. Available at: http://digitalhumanities.org/dhq/vol/10/1/000239/000239.html [Accessed 20 April 2017].

Bedell 2015 Bedell, W. "I Built a Botnet that Could Destroy Spotify with Fake Listens". *The Motherboard* October 16. Available at: http://motherboard.vice.com/read/i-built-a-botnet-that-could-destroy-spotify-with-fake-listens [Accessed 20 April 2017].

Boshmaf et al. 2011 Boshmaf, Y., Muslukhov, I., Beznosov, K. & Ripeanu, M. "The socialbot network: When bots socialize for fame and money". *ACSAC '11: Proceedings of the 27th Annual Computer Security Applications Conference*, 93–102.

Brügger 2016 Brügger, N. "Digital Humanities in the 21st Century: Digital Material as a Driving Force". *Digital Humanities Quarterly* 10.2. Available at: http://www.digitalhumanities.org/dhq/vol/10/3/000256/000256.html [Accessed 20 April 2017].

Clement 2016 Clement, T. "Towards a Rationale of Audio-Text". *Digital Humanities Quarterly* 10.2. Available at: http://www.digitalhumanities.org/dhq/vol/10/2/000254/000254.html [Accessed 20 April 2017].

Clement and McLaughlin 2016 Clement, T. & McLauglin, S. "Measured Applause: Toward a Cultural Analysis of Audio Collections" *Journal of Cultural Analytics* 1.1. Available at: http://culturalanalytics.org/2016/05/measured-applause-toward-a-cultural-analysis-of-audio-collections/ [Accessed 20 April 2017].

Davis et al. 2016 Davis, C. A., Varol O. & Ferrara, E. BotOrNot. "A System to Evaluate Social Bots". *WWW '16 Companion Proceedings of the 25th International Conference Companion on World Wide Web*, 273-274. Available at: http://dl.acm.org/citation.cfm?id=2889302 [Accessed 20 April 2017].

Eriksson 2018 (forthcoming) M. Eriksson, R. Fleischer, A. Johansson, P. Snickars and P. Vonderau. *Spotify Teardown. Inside the Black Box of Streaming Music*. Cambridge, Mass. MIT Press (forthcoming).

Ferrara et al. 2016 E. Ferrara, O. Varol, C.B. Davis, F. Menczer and A. Flammini. “The Rise of Social Bots”. *Communications of the ACM* 59 (7), 96-104.

Fiveash 2013 “Aussie bloke hacks way to top of music charts with MIDI-based tunes”.*The Register*. 11 May. Available at: http://www.theregister.co.uk/2013/11/05/peter_fillmore_hacks_into_online_charts/ [Accessed 20 April 2017].

Fleischer and Snickars 2017 Snickars, P. “Discovering Spotify”. *Culture Unbound* 9.2, 130-221 (thematic issue)

Foka and Arvidsson 2016 Foka, A. & Arvidsson, V. “Experiential Analogies: A Sonic Digital Ekphrasis as a Digital Humanities Project”. *Digital Humanities Quarterly* 10.2. Available at: http://www.digitalhumanities.org/dhq/vol/10/2/000246/000246.html [Accessed 20 April 2017].

Friesinger and Herwig 2014 Friesinger, G. & Herwig, J. (eds.).- *The Art of Reverse Engineering.* Bielefeld: Transcript.

Galuszka 2015 Galuszka, P. “Music Aggregators and Intermediation of the Digital Music Market”. *International Journal of Communication* 9, 254–273.

Garfinkel 1967 Garfinkel, H. *Studies in Ethnomethodology*. Englewood Cliffs, Prentice Hall.

Gheel 2014 Gheel, R. W. *Reverse Engineering Social Media. Software, Culture, and Political Economy in New Media Capitalism* Philadelphia: Temple University Press.

Hwang et al. 2012 Hwang, T, Pearce, I. & Nanis, M. “Socialbots: voices from the fronts”. *Interactions* 19.2. Available at: http://dl.acm.org/citation.cfm?id=2090161 [Accessed 20 April 2017].

Jdwhicker 2016 “[All Platforms][Other] Stop the spam bots”. *Spotify Community Blog*. 16 March 2016. Available at: https://community.spotify.com/t5/Closed-Ideas/All-Platforms-Other-Stop-the-spam-bots/idi-p/1308638 [Accessed 20 April 2017].

Madrigal 2014 Madrigal, A. C. “How Netflix Reverse Engineered Hollywood”. *The Atlantic* January 2. https://www.theatlantic.com/technology/archive/2014/01/how-netflix-reverse-engineered-hollywood/282679/ [Accessed 20 April 2017].

Morris and Powers 2015 Morris, JW. J and D. Powers. “Control, Curation and Musical Experience in Streaming Music Services”. *Creative Industries Journal* 8.2., 106-22.

Newton 2016 Newton, C. “The Search for the Killer Bot”. *The Verge* 6 January. Available at: http://www.theverge.com/2016/1/6/10718282/internet-bots-messaging-slack-facebook-m [Accessed 20 April 2017].

Robles-Anderson and Svensson 2016 Robles-Anderson, E. & Svensson, P. “One Damn Slide After Another: PowerPoint at Every Occasion for Speech”. *Computational Culture*. 15 January. Available at: http://computationalculture.net/article/one-damn-slide-after-another-powerpoint-at-every-occasion-for-speech [Accessed 20 April 2017].

Schwarz 2013 Schwarz, J. A. *Online File Sharing: Innovations in Media Consumption*. London: Routledge.

Snickars 2016 Snickars, P. “More music is better music”. In P. Wikström & R. DeFillippi (eds), *Business Innovation and Disruption in the Music Industry*. London, Edgar Elgar.

Snickars 2017 Snickars, P. “More of the Same — On Spotify Radio”. *Culture Unbound* 9.2, 184-211.

Spotify Team 2017 Spotify Team. “Find, Share and Listen to Spotify Song Clips with Friends Directly within Messenger”. *Spotify Blog*. 18 April. Available at: https://news.spotify.com/us/2017/04/18/find-share-and-listen-to-spotify-song-clips-with-friends-directly-within-messenger/ [Accessed 20 April 2017].

Turing 1950 Turing, A. “Computing Machinery and Intelligence”. *Mind* 49, 433–460. Available at: http://www.csee.umbc.edu/courses/471/papers/turing.pdf [Accessed 20 April 2017].

Wikström 2013 Wikström, P. *The Music Industry*. Cambridge: Polity Press.

Wikström and DeFilippi 2016 Wikström, P. & DeFillippi, R. (eds). *Business Innovation and Disruption in the Music Industry*. London, Edgar Elgar.

Woolley 2016 Woolley, S. “Automating Power: Social Bot Interference in Global Politics”. *First Monday* (21) 4. Available at: http://firstmonday.org/ojs/index.php/fm/article/view/6161/5300#author [Accessed 20 April 2017].

Woolley and Howard 2016 Woolley, S. & Howard, P. “Bots Unite to Automate the Presidential Election”. *Wired*. 15 May 2016. Available at: http://www.wired.com/2016/05/twitterbots-2/ [Accessed 20 April 2017].

Woolley et al. 2016 Woolley, S. et al. “A Botifesto”. “How to Think About Bots”. Available at: http://motherboard.vice.com/read/how-to-think-about-bots [Accessed 20 April 2017].