David Mash

16 December 2021

Crucial Problems with Machine Learning for Generative Music

Abstract: *Both machine learning and computer-generative music share the common goal of creating a system that autonomously produces content. However, when one accounts for the fact that the expressive power of computer-generative music derives from the process of creating the system rather than the simple presence of a generative system, an issue arises at the prospect of using machine learning to do the necessary analytical and implementational work. In this paper, I analyze generative music as a kind of meta-compositional practice. As part of this effort, I examine the inner workings of generative systems to explain why this medium is seen as highly expressive for the artists behind them. Once identified, I use the fundamental elements of these systems to evaluate the prospect of machine learning being used to create generative music. Ultimately, this paper explains that given its current usage, machine learning does not improve the expressive art of generative music, but one should recognize and appreciate its potential for the arts.*

Introduction

The concepts of machine learning and generative music can appear as a near-perfect match. After all, the main goal for generative artists is to create a system that makes music on its own, which seems to imply some level of computerized autonomy like machine learning. Although there is certainly plenty of untapped potential for the use of machine learning in the arts, I argue that it ultimately does not fit with what generative music has evolved into. In this paper, I explain the state of generative music, starting with the scholarship on the art and moving into concrete examples. I then, by referencing scholarship in the field of artificial intelligence (AI) as well as the existing applications of machine learning to music, demonstrate the differences between what has been achieved in the field and what is possible with machine learning. In the end, this paper identifies the machine learning approach to generative music as a negative divergence for the art form; however, its potential uses in other aspects of art and music are recognized and appreciated.

Generative art has been defined by AI and philosophy scholar Margaret Boden as "work that has been produced by the activation of a set of rules and where the artist lets a computer system take over at least some of the decision-making (although, of course, the artist determines the rules)" (Boden et al. 2009, p.4). Applied to music, this definition is further expanded when we consider the fact that the "computer" behind the music does not necessarily have to be the kind of operating system running machines we typically call "computers," but it can also include physical and conceptual music generating systems. With this expansion, it is not hard to see process music, like the works of Steve Reich, or even the fugues of Bach, as examples of generative music. Another generative art scholar, Philip Galanter, has gone so far as to say that "while it shouldn't be terribly surprising that the earliest forms of generative art used simple

systems, some will find it surprising and perhaps even controversial that generative art is as old as art itself" (Galanter 2003, p.12). In fact, even a set of wind chimes can be considered a generative system, with the craftsperson as the composer or artist behind the music (Brown 2005 p.217). As represented in figure 1 below, the artist behind a piece of generative music is the composer and creator of the system, which is what produces the final music.

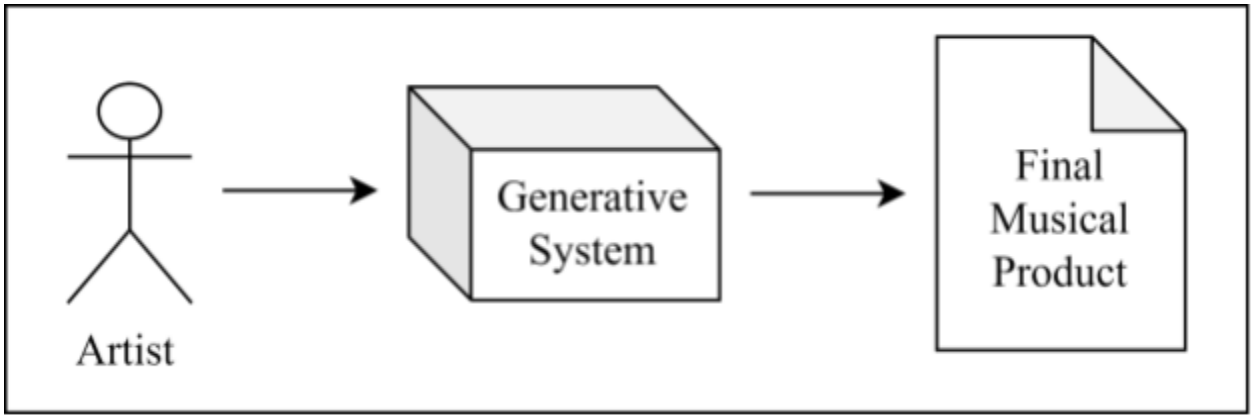

Figure 1 Representing the artist's relation to the final product in generative music.

The Implications Computer-Generative Music

Computers (as in the machines with CPUs) are now the most commonly used mediums for generative music. They provide artists with a closed environment in which they can efficiently explore much more complex generative systems than ever before. Artists utilize large sets of highly refined moving parts in the form of data structures and algorithms. Generative music artist and scholar, Nick Collins, has described the expressive abilities of algorithmic music in his 2008 paper, *The Analysis of Generative Music Programs*.

> "Algorithmic music is compositional design at a metalevel, human creativity in musical representations, examination of particular rule sets in a space of multiple music theories, with the composer–designer–musician becoming a 'composer-pilot' through musical modeling space. Composers model composition

itself, and such systems give us valuable insight into the relations of music theory, musical design and aural instantiation" (Collins 2008, p.239).

Collins attributes his usage of the term "composer-pilot" to Iannis Xenakis's paper *Formalized Music*, in which Xenakis explains that in quantifying the rules of music in a system, an artist must determine the "certain uncertainties" that constitute music composition itself (Collins 1992, p.144). This advanced level of refinement and complexity has created an expressive medium for generative artists that makes it possible to produce music based solely on the creator's musical identity without having to worry about the limitations of their system because it is no longer physical and does not even have to be human-readable. This kind of algorithmic generative music is something I refer to as CG-music, as in "computer-generative music," and it is inspired by Boden's use of the term "CG-art" in her paper (Boden et al. 2009, p.10).

CG-music has been explored in many projects using a variety of creative methods. Gustavo Díaz-Jerez's generative system, Nodal, produces music by performing user-created graphs, in which the "nodes," or vertices of the graph, hold note information, like pitch, and the connections between nodes, or edges of the graph, express other musical information, like timing (McCormack 2007, p.5). Peter Chilvers and Brian Eno worked together to develop Bloom, which is a generative music phone app, that responds to user gestures to create musical patterns, but can also make music completely on its own (Siegal, 2018). In the 1980s, George Lewis created a groundbreaking generative system that he called Voyager. In his paper, "Too Many Notes: Computers, Complexity and Culture in 'Voyager,'" Lewis notes that his system is "not asking whether machines exhibit personality or identity, but how personalities and identities become articulated through sonic behavior" (Lewis 2000, 38). He argues that Voyager's sonic

behavior has embodied the “African-American aesthetic and musical practices” (Lewis 2000, 33). Though Lewis holds the code behind the system rather tight to the chest, it is clear that the AI he created for his system is based on a specific preset musical grammar.

In his 1981 paper, *Using Generative Grammars for Music Composition*, S. R. Holtzman explains that similarly to linguistics, the processes that musicians use for music composition can often be expressed as a set of grammar rules. Holtzmann explained how one could use a Generative Grammar Definition Language (GGDL) to express rules for music composition on both a macro and micro level. A GGDL represents generative processes by expressing rules for transforming inputs (Holtzmann 1981, p.52). These transformations can be simple replacements, transitional matrices, Markov chains, or any applicable high-level function. The output can represent any kind of musical object or structure, like an entire section of music, a note’s length, a sound envelope, etc. Once an artist has defined all of the macro and micro level rules that make up their generative grammar, they need to apply those rules to code, which creates a form of artificial intelligence for CG-music.

Good Old Fashioned Artificial Intelligence vs. Machine Learning

There is a widespread misconception that the term artificial intelligence (AI) implies the presence of a neural network or machine learning (ML) algorithm. ML is a subcategory of AI, but it is not present in any of the CG-music systems mentioned earlier or in the process of turning a musical grammar into a generative system that I just detailed. The kind of AI used in those examples is referred to as symbolic AI, or Good Old-Fashioned AI (GOFAI) because it is a classical approach to AI that does not use newer concepts like neural networks and machine learning (Boden 2014 p.89). In a typical GOFAI, probabilistic selections are made from symbolic data structures and tables that represent solutions to fractional problems that, together, result in a

final solution. Programmers often use expert opinions to transform these symbolic data structures to make their probabilities favor the accepted correct answer. In her essay included in *The Cambridge Handbook of Artificial Intelligence*, Margaret Boden talks about the fundamental concepts behind GOFAIs:

> "The program is provided before it starts with a number of possible differences, a list of actions (operators) that can eliminate the various differences, lists of prerequisites that must hold if a certain operator is to be used, and heuristics for ordering the actions when more than one action is possible in the current circumstances. Indeed, most of the 'intelligence' involved lies in the choices of actions, operators, and heuristics specified by the programmer." (Boden 2014, p.90).

An important part of Boden's description is the recognition that the programmer and creator of the system is the one that determined the choices, operators, and heuristics available to the generative process. This relationship is illustrated below in figure 2, which shows that the use of GOFAIs to create CG-music may increase the complexity of the system but does not interfere with the artist's relation to the final product.

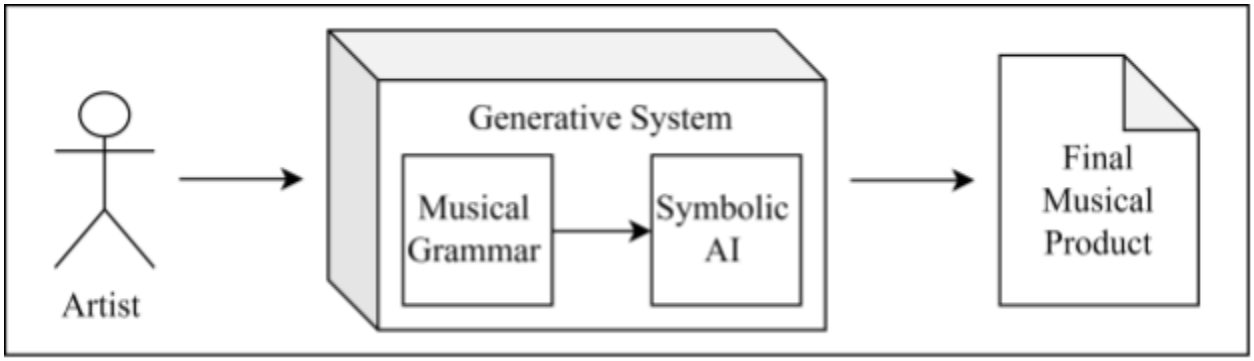

Figure 2 Representing the artist's relation to the final product in generative music with the use of musical grammar analysis and symbolic AI.

Symbolic AI and machine learning (ML) are often blurred as synonymous concepts when discussed by people outside of the scientific field, but they have always, since their conception, remained two distinct categories of AI that differ in philosophy and implementation (Franklin 2014 p.15). In contrast to GOFAIs, machine learning (ML) approaches do not have the programmer decide what choices, operators, and heuristics the process should use. In David Danks's essay included in the previously mentioned handbook, he explains the fundamental concepts behind machine learning algorithms.

> "The algorithm works by extracting – and exploiting – structural relationships among the variables without regard to the meaning or domain of the variables. For example, if doing classification using an artificial neural network, one might be provided with a dataset containing measurements of various features of widgets, as well as some target category. The neural-net learning algorithm then uses only the statistical regularities in the dataset to learn the relevant inter-variable structure, which can then be used to predict the target category for future widgets. The precise 'meaning' of the variables is irrelevant to the learning algorithm" (Danks 2014, p.152).

There are different kinds of ML algorithms, but in general, ML requires a training dataset, which in our case would be a collection of raw audio clips or, more likely, MIDI files of existing songs of which we want our algorithm to be able to recreate the style. These samples are processed by the ML algorithm to create a trained model that will hopefully be able to identify common features among the samples if they are present (Danks 2014, p.152). If used correctly, a trained model could be used to recognize these features in new inputs or imitate those features in their own synthesis. Thus instead of having the artist and creator consider what features of music composition are important to them and apply those features as grammar rules to a system of their own design, the ML algorithm performs the analysis and creates the model on its own. This ML approach is clearly divergent from the process discussed earlier, and for the purposes of creating CG-music it is divergent in a negative way. The problem with using ML to create CG-music is that it takes away those decisions regarding the generative grammar and implementation that made the GOFAI and other approaches expressive mediums. In its place, ML offers other artistic opportunities that I will discuss later, but in terms of CG-music, it is certainly a negative divergence with respect to the expressive ability of the artist.

There are several ML projects that have been created to work with art and music. One such project, called Magenta, was launched by Google in 2016 as an art-focused model based on TensorFlow, which was their original ML algorithm. There are many examples of computer scientists using Magenta to do all sorts of interesting things with sound. One of my personal favorites is the NSynth project, which dynamically mixes sounds together and has created a new way for music producers to synthesize unique timbres (NSynth Team 2017). Projects like NSynth are meant to inspire artists or just to act as proofs of concept. Other projects use Magenta to try to make generative music. With varying success, these projects often achieve what they set

out to do: training an ML algorithm on a massive dataset of music in order to have it output strings of notes. In fact, there have been blogs written about Magenta with titles like “The Beginning Of The End For Composers? Google Magenta Just Wrote Its First Song” (Sethi 2016). Despite the excitement around ML being used for music composition, the melodies produced from Magenta are short and obviously just imitations of the music it has analyzed (Sethi 2016). This is because Magenta’s understanding of music came exclusively from the set of premade music that it was trained on and, as Danks explained, ML algorithms do not care about the “meaning” of the variables they process (Danks 2014, p.152). This ability to imitate could be an interesting attribute to explore, but as a CG-music system, it definitely falls short of the expressive, perspective-capturing ability of systems that utilize GOFAIs and other techniques.

In general, audiences perceive art far differently than the creators do in the process of creating the work. This can be largely attributed to the fact that viewers often do not have a complete picture of the self-imposed rules and contexts with which creators implement the objects of their work (Lefford 2021, p.129). When it comes to physical generative systems, the rules are never very difficult to understand. For example, in Steve Reich’s “It’s Gonna Rain” the rules are clear: there are two recordings on tapes of slightly different lengths that are played at the same time which results in their contents slipping in and out of sync with each other, creating the final piece. When the processes get more complex with CG-music, audiences can still be grounded in the fact that the creator of these systems have meticulously installed their own perspective into the music by prescribing how the musical decisions are made. However, when ML replaces that fragile connection between the artist and the final product, the transparency of the process is completely broken for both the artist and the audience, which disembodies the music and contributes to the loss of expressive ability. This disembodiment of the final product is

represented below in figure 3, where the question mark (?) shows the unknown or debatable relationship between the artist and the system.

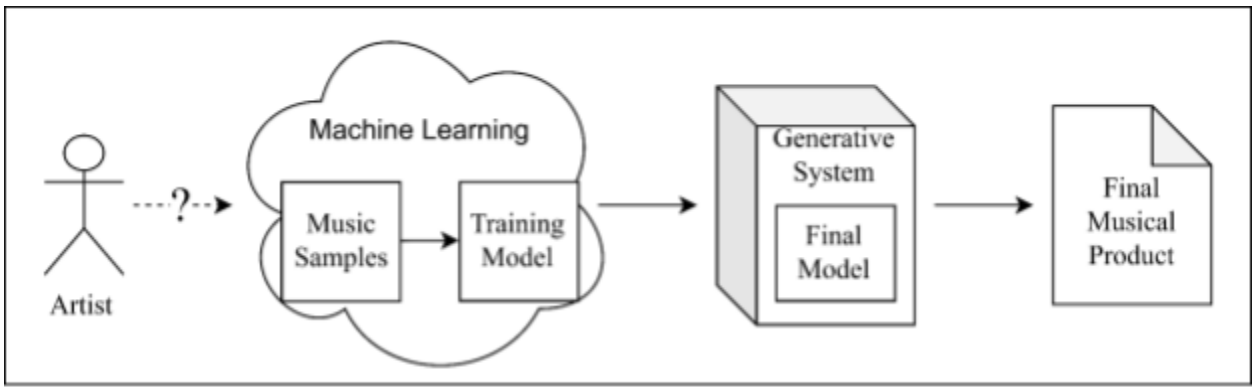

Figure 3 Representing the artist's debatable relation to the final product in music generated using machine learning.

The beauty of CG-music comes from "compositional design at a metalevel" (Collins 2008 p.239) and not simply from the presence of a machine that can imitate compositional features. ML has undoubtedly been used to advance many fields of study (Franklin 2014 p.16), but that does not mean that it can be applied everywhere with the same effectiveness. Because ML works by identifying "statistical regularities" in a dataset (Danks 2014, p.152), when used for art or music, it can only create imitative work. It also takes away the heightened expressive power of CG-music by removing the control that the artist had over how the final product would perform.

Conclusion

The state of generative art using machine learning is still primitive and there is high potential for it, but as it exists today, it is not a groundbreaking technique for CG-music and it is not advancing the art form. Though it isn't as expressive for artists of generative music, ML can still be used as a powerful tool in the conception and creation of art and music; its product just

needs to be curated or abstracted in some way. For example, Magenta was used by the band, YACHT, to create unique sounds that they used in their music. They fed the learning algorithm with a dataset of samples from their released music and Magenta outputted melodies and lyrics that they were able to pick and choose from to compose their new music (Friedman 2019). Magenta has also enabled the creation of NSynth, as mentioned earlier, whose team has made it clear that rather than trying to use ML to replace some human function in music, their goal is "to aid the creative process" (NSynth Team 2017). These projects are not related to generative music, but they demonstrate how ML can be used by musicians in the process of producing music.

It was smart for the NSynth team to avoid trying to replace human roles in music because Magenta, as it exists now, cannot process music like humans in the first place. This is caused by the fact that ML algorithms always try to fit data to the curve they were trained on, which makes their outcomes biased in highly unnatural ways (Kant 2019 p.7). In fact, these issues have been exploited to create art that plays on the inabilities of ML. A project known as the "Happy Valley Band" created by David Kant uses machine listening to transcribe recordings of music. The produced notation is consistently riddled with errors that wouldn't have been a challenge for human ears to get right. These errors are exactly what Kant is looking for and he puts them on display by giving these computer-generated transcriptions to a band of talented musicians to perform. In relation to the music that is originally fed into the system, the final performances are mystifyingly wrong and often almost completely incoherent, but in a strange way, they quite eloquently demonstrate the discontinuity between human comprehension of music and an algorithm's processing of sound files.

Though there are these places and more for machine learning in art and music, I do not believe that the future for CG-music is likely to be intertwined with these algorithms as they appear today. Despite this, the genre has found a way to constantly evolve. The “live coding” community has become an active place where generative music continues to develop. Live coding is the practice of actively programming and modifying a generative system while music is being produced. During performances of live coded music, the artist typically shares their code with the audience as they write it by projecting it on a large screen (Brown et al. 2009 p.18). This extreme level of transparency lets the audience know exactly what is happening as the artist builds the process right before their eyes, which gives this form of CG-music an ideal amount of shared artist-audience context (Lefford 2021 p.130) - the type that made physical generative systems so approachable. In this way, live coding addresses all of the problems I have found with machine learning approaches to CG-music. For the future of CG-music, I believe that artists in the live coding community will have more success than any machine learning algorithm.