Published as: Roger B. Dannenberg, "Artificial Intelligence, Machine Learning, and Music Understanding," in *Proceedings of the Brazilian Symposium on Computer Music (SBCM2000)*, Curitiba, Brazil, 2000.

Artificial Intelligence, Machine Learning, and Music Understanding

Roger B. Dannenberg
School of Computer Science and College of Art
Carnegie Mellon University
Pittsburgh, PA 15213 (USA)
rbd@cs.cmu.edu, http://www.cs.cmu.edu/~rbd

Abstract. Artificial Intelligence and Machine Learning are enabling many advances in the area of music. Three computer music problems are described in which Machine Learning promises to solve problems and advance the state of the art. These are: computer accompaniment, music understanding in interactive compositions, and music synthesis. Machine Learning plays an important role in dealing with poorly defined problems where data is subject to noise and other variation, and where complexity rules out direct, handcrafted solutions. These characteristics are typical in sophisticated computer music systems. Machine learning promises to enable more natural communication between machines and musicians.

1. Introduction

Computers are a fantastic tool for composers, performers, theorists, engineers, and scientists. All of these bring a different perspective to music, but all can profit from the precision and automation afforded by computers. Computers have even more to offer than just a better calculator, printer, audio recorder, digital effects box, or whatever useful immediate function the computer serves. The real significance of the computer lies in the new paradigms of art-making and scientific thought that are engendered by computing.

The concept of algorithms and automated processing was not new when computers first came along. The famous example of Mozart's musical dice game is often cited as an early example of algorithmic composition. However, the computer allows composers to specify elaborate composition systems and to realize them quickly without error. With the advent of computers, what was previously a vague concept can now be studied experimentally. Algorithmic composition has become an important and influential compositional technique. Notice that the role of computers here is not simply to automate existing practice but to enable the development of an entirely new conceptual approach to music composition. In this case, the notion of algorithms has translated into an important line of musical thought and practice.

I believe that there are many similar conceptual advances in store for virtually all fields of intellectual endeavor, including music. One of the major areas of new development, experimentation, and change in computer science is that of Machine Learning. To make a long story very short (and probably overly simple), researchers in Artificial Intelligence (AI) started by writing programs to solve difficult problems that had no direct algorithmic solution. Early AI programs used a variety of techniques, especially sophisticated forms of search. It became apparent that humans solve many problems using prior knowledge to guide their search, so AI researchers began to study knowledge acquisition and representation. "Expert Systems" emerged from this and found many applications. As researchers continued to tackle hard problems, some began to look for ways that computers could acquire their own knowledge, thus "learning" about problems and their solutions. In many cases, machine learning systems have been found to surpass systems based on manually coded knowledge.

What does Machine Learning have to do with music? The most important thing is that Machine Learning is creating a new way of thinking about our world and about us. This philosophical shift will undoubtedly change the way we think about and create music. It is too early to say with any certainty what will happen, but I would like to describe some of the work my students and I have been doing that suggests some directions music might take. My work has origins in Artificial Intelligence, especially with the goal of Music Understanding, and only lately has this work begun to come under the influence of Machine Learning.

I will describe several projects where Machine Learning is beginning to show promise. The first of these is c*omputer accompaniment*, in which a computer listens to, follows a human performer, and performs in synchrony with the performer. The second is *interactive composition* in which computers compose music that is responsive to the performance of a live musician. Finally, I will describe some work in *music synthesis*, in which synthesis depends upon learning to perform and shape notes the way humans do.

2. Computer Accompaniment

Computer accompaniment systems were first introduced in 1984 (Dannenberg, 1985). One would think by now this application of computer music would be well known, but it still surprises many visitors to my lab that this is even possible. Computer accompaniment systems are modeled more or less on human accompaniment: The accompanist begins with a composed score containing both solo and accompaniment parts. The accompanist listens to the solo and follows the solo in the score, matching performed sounds to notated symbols. As the accompanist follows the score, the location and tempo of the soloist become apparent. The accompanist then uses this information to synchronize a simultaneous performance of the accompaniment part from the score. The accompanist is also a performer and is expected to play the accompaniment part expressively and musically. Looked at more generally, computer accompaniment takes place whenever musicians play together, when a conductor conducts an orchestra to synchronize with a soloist or small group, and even when a lighting crew synchronizes lights to a musical performance.

2.1 Is this Artificial Intelligence?

Computer accompaniment in the systems I have built relies upon many techniques developed under the general rubric of “artificial intelligence.” As with most AI programs, computer accompaniment does not seem very intelligent once you understand all the details. Nevertheless, computer accompaniment systems must deal with the recognition of imprecise and sometimes erroneous data (the performance). I have described the use of dynamic programming to perform real-time matching. This approach is related to and derives from earlier work by others in speech recognition. Accompaniment systems must also embody knowledge about musical ways to change tempo and synchronize with a soloist. This knowledge is encoded in my systems as a set of conditions that can be tested, and a corresponding set of solutions that can be executed, to achieve musical behavior.

My initial work in computer accompaniment has led in many directions. One of these was the construction, with my student Lorin Grubb, of an ensemble accompaniment system (Grubb and Dannenberg 1998), that is, one that can play along with an ensemble rather than a soloist. Aside from the obvious difference that an ensemble has more than one performer, there are two key differences between an ensemble and a soloist. First, the members of an ensemble do not necessarily play in synchrony, as one performer can get behind or lost. Soloists have a much harder time playing in two places at once (although my piano playing might offer an

exception)! Second, ensembles raise the question of who to follow. Usually, the accompanist must listen to different ensemble members at different times because not everyone plays all the time. In our system, we listen to all performers all the time, but then we have to decide moment-by-moment what information is the most appropriate and most reliable.

One of the interesting results of our investigations of ensemble performance relates to the idea of self-awareness. Early on, Lorin and I were playing trios with the computer, making intentional errors to test the system. We found that if we deliberately diverged so as to be playing in two different places, the computer could not decide who to follow. Even if one of us played normally and the other made an abrupt departure from the normal tempo, the computer would *not* always follow the "normal" player. In a moment of inspiration, we realized that the computer did not consider itself to be a member of the ensemble. We changed that, and then the computer performed much more reasonably. Here is why this worked: When the computer became a first-class member of the ensemble and one of us diverged, there were still two members playing together normally, e.g. Lorin and the computer. The computer, hearing two members performing together, would ignore the third.

Some people consider self-awareness to be a mystical concept or at least a higher form of intelligence. I think this example shows that self-awareness might be much simpler. In my opinion, self-awareness is a natural consequence of an organism (or computer program!) trying to build a successful model of the world in which it interacts. I think there is plenty of opportunity for computer music systems to become more self-aware. Our little experiment shows that if you do not pay attention to your own musical actions, you may not even be able to play trios!

2.2 Computer Accompaniment and Machine Learning

My early work in computer accompaniment views the problem in algorithmic terms. The score and the performance are viewed as sequences of symbols (simply pitch names, without rhythmic attributes). When the input is acoustic, signal-processing techniques are used to segment the signal into notes and to estimate the pitch of each note. The problem is thereby reduced to the question of comparing strings of symbols.

With the discrete symbol approach, notes either match or they do not match. There is no concept of "almost matching" or matching with degrees of certainty. If scores are accurately performed and pitches are accurately estimated, then this may not be a problem. With vocal performances, however, it is difficult to segment a performance into individual notes. Vibrato and other variations make pitch estimation difficult even if the signal is segmented properly. The result is that vocal accompaniment using my symbol-matching techniques does not work well at all.

My student, Lorin Grubb, invented an entirely new approach. (Grubb and Dannenberg 1998) The key is to represent the score position not as a discrete position but as a probability density function. The function represents the likelihood that the performer is located at a given position. The beauty of this representation is that it can be updated with partial information. Even if two notes do not match, a close match increases the probability that the performer is at a given location. The closer the match, the more likely is the position. This representation is also good for integrating various sources of information. Position can be estimated based on a prior estimate (estimates are made about 30 times per second), the current pitch, the current spectrum, and the current amplitude. Each of these information sources contributes some information to the overall position estimate.

This statistical approach works better than the symbolic approach because it uses more information. The disadvantage, however, is that statistical information about the system and about performers must be collected. This is not a programming task, nor is the information readily available from existing studies. The only practical way to make the system work is to collect information by "training" the system using real performances.

Grubb used a set of 20 performances and hand-labeled thousands of notes to characterize the typical tempo variation, pitch deviation, and other data distributions. These are then used by the system during a performance to assess the likelihood of any particular location in the score. Thus, machine learning has become critical to the success of advanced computer accompaniment systems.

3. Interactive Compositions

After working on computer accompaniment and developing systems that perform well enough for public concerts, I expected to begin writing some music to take advantage of this new technology. It would be interesting to create pieces that are not limited by the synchronization problems of tape. I also thought about dynamic structures where the performer could make various choices. A modified accompaniment system could quickly determine by listening and pattern matching what choice the performer has made.

Ultimately though, I did not find the traditional model of composition and performance as interesting as other possibilities. What interests me now is the idea of using the computer to compose the details of the piece during the performance. This allows the computer to respond in many ways to the performer, and it allows the performer the freedom to improvise and participate in the composition process.

This is a new form of notation. It is less precise than standard notation, because it does not specify notes. Instead, a computer program captures more abstract aspects of the work that are most important to the composer. These may be scales, rhythms, harmonies, overall structure and development, ways of responding to the performer, sonorities, or textures, to name just some of the possibilities.

How is this a notation? It decidedly *is not* a notation in the sense of visual material that instructs how to play the piece. But it *is* notation in the sense that it guides the performer with the intentions of the composer. A sensitive performer will be guided by the computer-generated music. In addition, the performer will discover that certain ways of playing create more interesting music than others.

For example, if the composer wants sustained tones with variations in loudness, this could be "notated" by generating interesting counterpoint only when tones are held and by giving the performer some control over the counterpoint using variations in loudness. This may seem to be much more difficult than simply writing whole notes and dynamic markings, but there are some benefits. The performer is free to explore the musical territory created by the composer, perhaps discovering new and beautiful possibilities. In this example, the performer might exploit rotary breathing to create extra-long tones. With loudness as an interactive control parameter, the performer might be motivated to discover new ways to control and articulate his or her instrument.

Of course, the composer can (and should) always communicate musical intentions verbally and even with traditional notation. However, I find something very compelling about the idea of the music itself being the notation. Performers do respond to what they hear, so interactive

compositions offer a new way for composers to communicate with performers using the computer as an intermediary.

3.1 Music Understanding

Interactive compositions are like robots. They perceive information from the "real" world. They contain a model of the world, their position, their goals, and other internal state information. They generate output intended to change the robot and the robot's world. In musical terms, an interactive composition listens to music. The interactive composition has an internal model of the performance, where it has been, where it is going, what is happening now. It generates music that the performer is listening to. Although it may not be explicitly represented, the generated music manipulates "the world" because the performer is listening and responding. Ultimately, the interactive composition has a goal, perhaps to move through a particular of musical terrain, realizing musical sounds by visiting certain locations in a certain prescribed time.

There are really just two key technical requirements identified in the preceding paragraph. Interactive compositions must "listen" and "generate." Since music generation is widely studied and written about under the topics of algorithmic composition, musical grammars, Markov-models, rule-based systems, and many others, I will confine my discussion to the requirement of listening.

Many interactive music systems have been created, and they have explored many interactive techniques. MIDI has given composers a terrific boost by solving many of the hardware and interfacing problems associated with sensing music performers. More recently, software-based digital audio signal processing has enabled direct audio capture and processing in interactive systems. In most cases, the level of "listening" has been very rudimentary. Typically, systems use MIDI notes to trigger events. Different pitches or simple pitch patterns may give the performer some additional control over the music. Additional parameters have been derived from simple statistical measures such as note onset density, average pitch, and rhythmic and pitch interval sizes.

Unfortunately, simple parameters do not do a very good job of capturing musical intent. This is not necessarily a problem in absolute musical terms: Great music is often produced by simple means, and the most sophisticated technology does not guarantee musical quality. Nevertheless, if the goal is to construct compositions that truly communicate in musical terms with a live performer, it is frustrating to be limited to a simple syntactical vocabulary of note onsets and quantized pitches. Instead, the computer and performer should be communicating in much more abstract musical concepts such as lyrical, freely, tense, angry, sustained, bright, tonal, linear, heavy, and so on.

3.2 Style Classification

Many composers have tried to create software that captures these abstract concepts. Usually, input processing is developed on an ad-hoc basis depending upon the requirements of the piece, so there are few if any scientific studies with objective evaluations. My personal experience is that it is fairly simple to create interactive systems that respond in interesting ways to music performances, but capturing abstract musical notions can be quite difficult.

For example, my piece "Nitely News" listens to a trumpet while playing Miles Davis jazz-rock influenced music consisting of synthesized electric bass and drums. I wanted an animated dancer and the drummer to do something wild when the trumpet playing becomes "frantic." What does "frantic" mean? For me, it is the sense of playing very rapidly and

energetically, with a variety of intervals and articulations. Can a computer program capture this concept? At first, I thought it would be simple. Just count the number of notes over a time interval. When the number gets to be high, and perhaps remains high for a few successive intervals, the playing must be "frantic." Unfortunately, and surprisingly, this approach did not work. Nor did a series of reasonable refinements. It seems that when playing frantically, there is silence to take a breath, and there are longer notes to create contrast. Perhaps the pitch detection apparatus breaks down when the notes are not cleanly articulated or held for a normal duration, leading to lower note counts. In any case, my hand-coded "frantic" recognizers often fail.

This is the point at which my students, Belinda Thom and David Watson, and I began to explore machine learning techniques for what we call *style classification*. The idea is simple: If recognizers based on one or two parameters such as note counts are failing, why not try to consider *many* low-level features? Furthermore, rather than try to guess what features are important and what range of values might be appropriate, why not *train* recognizers on actual performance data?

We used a set of about a dozen low-level features, including such things as note counts, pitch, duration, duty factor, and standard deviations of these parameters, all measured over 5-second intervals of time. Features were extracted from real performances in various styles. This is a standard supervised learning problem. The analyzed features represent the input, and the performed style is the output we are looking for. The goal is to train a classifier that, when given new input in the form of low-level features, will output the correct style.

3.2 Results

We created naive Bayesian classifiers, linear classifiers, and neural networks using our training data. The classifiers had recognition rates in excess of 98% for simple problems with 4 different styles. (Dannenberg, Thom, and Watson 1997) This is much better than anything previously coded by hand. More recently, Brad Beattie (1998) and my student Jonathan Coleman (2000) have applied these techniques to polyphonic keyboard music with some success.

There is much more work to be done. For example, Coleman's study looked at the effect of varying the length of the time interval during which features are collected. As expected, longer intervals lead to more accurate classification, but this was on data that was not changing in style. What we really want for interactive compositions is to detect when the style has changed, and this requires shorter time intervals. How is it that humans seem to form stable impressions of style, but are able to revise that impression quickly when the style changes? It seems clear that some on-line learning is taking place. We characterize what we are listening to as we listen, so that when something new comes along, we can evaluate it as new relative to what we were just hearing in addition to evaluating it in absolute terms. This area is ripe for further research.

4. Music Synthesis

Another area where machine learning shows great promise is that of music synthesis. This may seem like an unlikely area for learning applications, so consider the following approach to synthesis. The combined spectral interpolation synthesis technique has been developed in my lab by a number of students and researchers over the years. The main contributors have been Marie-Helen Serra, Dean Rubine, and Istvan Derenyi. The basic idea is quite simple, and

to simplify things even further, we will consider only the synthesis of trumpet tones. (The technique has also been applied to other brass and woodwind instruments.)

A trumpet in the hands (and on the lips) of a given player is very consistent in producing tones. Given a dynamic level and a pitch to play, the player will produce very nearly the same spectrum every time. Even though the trumpet is a complicated dynamical system, it appears that the system behavior is almost entirely characterized by the amplitude and fundamental frequency at any moment. We have verified, for example, that the spectrum at a given amplitude level is very nearly the same whether the signal is examined during a held note, a rapid attack, or a release. (Dannenberg and Derenyi 1998) This means that the trumpet sound can be described by capturing spectra at many different points in a two-dimensional space of amplitude and pitch. To reproduce any sound, you describe the sound as a point moving through pitch and amplitude space. That moving point indexes a spectrum (interpolation can be used to estimate the spectrum from the captured data). The resulting time-varying spectrum can be generated using additive synthesis or wavetable interpolation.

Using this approach and a few enhancements that are described elsewhere (Serra, Rubine, and Dannenberg 1990), we can create very natural-sounding trumpet tones. However, it turns out to be very important to drive the synthesis model with appropriate control functions. Traditional computer music techniques such as ADSR envelopes or even envelope tables derived from actual performances do not sound natural when used to synthesize phrases. This synthesis approach relies upon good control functions for pitch and amplitude.

4.1 The Importance of Control Functions

My student, Hank Pellerin, made a study of trumpet envelopes by measuring envelopes from actual performances. (Dannenberg, Pellerin, and Derenyi 1998) Instead of taking the usual approach seen in the literature where isolated tones are recorded and analyzed, we looked at tones performed in context. What we discovered is that there are dramatic and systematic changes to envelopes that depend upon how the note is articulated, whether the note is part of an ascending or descending line, and other factors relating to the context of the note. A quarter note followed by a rest is played differently from one followed by another note.

To deal with this variation, I studied Pellerin's results and I looked at a number of other examples. I created a function with many parameters that could approximate any trumpet envelope in the set of examples. I then determined parameters manually for a collection of envelopes and studied how these parameters varied with context. Because the envelope function was created by hand, most of the parameters have an intuitive meaning. It was not too hard to find reasonable ways to compute these parameters from information available in the score, including context, slurs, and dynamic markings.

With my new envelope function, it is possible to create good envelopes from a machine-readable encoding of music notation. It is interesting that ordinary MIDI is not sufficient because it lacks articulation information that is typically found in music notation. The results are some of the best synthesized trumpet sounds currently available.

4.2 Machine Learning and Music Synthesis

A limitation of this approach is that it does not handle non-classical playing techniques where inharmonic and transient sounds are significant. Half-valve, flutter-tongue, smears, falls, and doits are examples of effects that we cannot produce well. More importantly, there are limitations with my handcrafted envelope function. It works reasonably well for a small range of notation, but there are many places where it does not work well at all. For example, we

"cheated" on our sound examples by eliminating notated trills. The spectral interpolation technique can generate beautiful trills using control functions derived from real performances, but my envelope function does not generate good control functions for trills. Improving the envelope function is difficult and tedious because one must consider more and more factors and relate them to a fairly large number of output parameters.

The solution, we believe, is to apply machine learning to create the envelope automatically. More precisely, we need to obtain a set of features from music notation and convert these into a dozen or so numerical parameters that drive the envelope function. This can be set up as a supervised training problem. Features are derived from the notes in a set of scores. A performer plays the scores and each note is analyzed to obtain its envelope. Further analysis can obtain the dozen or so parameters that describe the envelope. Finally, we have input data from the score and output data in the form of a dozen or so parameters. Machine learning is used to "learn" the function from input to output.

I believe this approach shows great promise. Already, machine learning has been used to determine how musicians express different emotions in their performances. (De Poli, Roda, and Vidolin 1998) MIDI files and other representations of music can be altered to express emotion according to what was learned. This is quite similar to learning how notation and context affect performance parameters to make a music synthesizer sound natural, musical, and therefore realistic. Ultimately, I expect that synthesizers will learn to perform from machine-readable notation augmented with annotations containing stylistic and emotional directions.

5. Summary and Conclusion

I have outlined several areas where machine learning can make significant contributions to computer music. In computer accompaniment, the problem is to integrate unreliable and noisy information from a variety of sensors in order to track the progress of a performance in a score. Symbolic matching fails when the information is too unreliable, but statistical techniques allow us to extract and integrate whatever information is available. Machine learning helps to characterize the information sources and the expected behavior of performers.

Interactive compositions are a class of programs that rely heavily upon music understanding or listening. It is important to understand not just the syntactic elements of music such as notes, pitches, and duration, but also the higher level concepts at which real musical communication takes place. Machine learning has been applied to the creation of style classifiers, a first step at getting machines to deal with the same abstract musical concepts as the human performer. By training classifiers with data from the performer, the machine can learn exactly what the performer means by, say, "frantic" or "lyrical." Very reliable classification has been achieved, at least in simple examples.

The third application area is music synthesis. Synthesis is not really decoupled from performance as previous computer music research has assumed. In order to make a trumpet sound like a trumpet, you have to play it like a trumpet player. This adds great complexity to the synthesis problem, and this particular complexity is not addressed by the electrical engineering and systems theory that has dominated synthesis research in the past. It seems that machine learning offers a viable approach to solving some of these problems.

Machine learning will find application in these and other areas of computer music. More importantly, machine learning will alter the way we think about music and music

composition. The possibilities of training musical systems to behave the way we like are fascinating. The loss of direct detailed control over the behavior of complex learning systems will add a new element of interest to live performance (although many would say we have already achieved this level of complexity in computer music performance even without machine learning). The machine learning paradigm will undoubtedly affect the way we think about music as composers, performers, and listeners.

Taken together, this research all points toward higher levels of communication between machines and musicians. Accompanists that listen to performers, interactive compositions that converse in a sophisticated musical language, and synthesizers that model musicians as well as their instruments all demand new levels of music understanding by machine. Machine learning and artificial intelligence are the technologies to deliver music understanding. As this newly forming technology advances, many new applications and possibilities will emerge.

Acknowledgments

I am most grateful to my students, who inspired, enabled, and conducted so much of this work. The School of Computer Science at Carnegie Mellon has provided a good home for this work, along with a completely open mind toward interdisciplinary research that simply could not exist at many other institutions. Connecticut College and Stanford's CCRMA invited earlier versions of this presentation in lecture form. Finally, I would like to thank the organizers of this conference for inviting this paper in written form.

References

Beattie, B. 1998. *Improvements in a Machine Learning Appoach to Musical Style Recognition*. The School of Engineering, Tulane University. (Masters Thesis)

Coleman, J. 2000. *Musical Style Classification through Machine Learning*. Carnegie Mellon University. (unpublished senior thesis).

Dannenberg, R. B. 1985. "An On-Line Algorithm for Real-Time Accompaniment," in *Proceedings of the 1984 International Computer Music Conference*, Computer Music Association, (June), 193-198.

Dannenberg, R. B. and I. Derenyi. 1998. "Combining Instrument and Performance Models for High-Quality Music Synthesis," *Journal of New Music Research*, 27(3), (September).

Dannenberg, R.B., H. Pellerin, and I. Derenyi.1998. "A Study of Trumpet Envelopes." In *Proceedings of the International Computer Music Conference*. ICMA. pp. 57-61.

Dannenberg, R. B., B. Thom, and D. Watson. 1997. "A Machine Learning Approach to Musical Style Recognition" in *1997 International Computer Music Conference*, International Computer Music Association (September), pp. 344-347.

De Poli, G. A. Roda, and A. Vidolin. 1998. "Note-by-Note Analysis of the Influence of Expressive Intentions and Musical Structure in Violin Performance." *Journal of New Music Research*, **27**(3) (September), pp. 293-321.

Derenyi, I. and R. B. Dannenberg. 1998. "Synthesizing Trumpet Performances." In *Proceedings of the International Computer Music Conference*. San Francisco: ICMA, pp.490-496.

Grubb, L. and R. B. Dannenberg. 1994. "Automated Accompaniment of Musical Ensembles," in *Proceedings of the Twelfth National Conference on Artificial Intelligence*, AAAI, pp. 94-99.

Grubb, L. and R. B. Dannenberg. 1998. “Enhanced Vocal Performance Tracking Using Multiple Information Sources,” in *Proceedings of the International Computer Music Conference,* San Francisco: International Computer Music Association, pp. 37-44.

Serra, M.-H., D. Rubine, and R. B. Dannenberg. 1990. “Analysis and Synthesis of Tones by Spectral Interpolation,” *Journal of the Audio Engineering Society*, 38(3) (March), pp. 111–128.