

Shane Hoversten

Papadopoulos and Wiggins, AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects

This paper discusses five categories of computer science techniques for music composition, although I should mention at the outset that I find the divisions between the categories somewhat odd.

Mathematical models

The first category of techniques is “mathematical models.” I would venture that the preponderance of these are actually statistical models, like HMMs and Markov Chains. These methods have shown promising results in a number of learning applications, both in music composition and, more famously, NLP domains. Other approaches, like “chaotic nonlinear systems” and “iterative approaches” I’ve never heard of, and the paper doesn’t explain.

The paper points out that these techniques require the probabilistic features of a lot of corpora to be collected, and that they do not model well the departure from structure, which is often a critical component of skillful composition. About the former point I can only agree; data sparsity is the monkey on the back of statistical machine learning. The latter is an interesting point; much current work in this area addresses this exact issue. Hierarchical HMMs are an example.

Knowledge-based systems

The authors claim that “in one sense, most AI systems are knowledge-based systems.” That might have been true at one time, but I suspect most AI that’s actually doing anything is of the statistical variety, mostly because of internet-based natural-language applications like search. At any rate, KBS have shown themselves to be good at a number of problem domains, which is not really surprising: if you can get a bunch of experts to precisely formulate what they know about a domain, you can probably come up with a pretty decent solution. This was true even twenty years ago, with medical diagnosis expert systems. When KBS are good, they’re often very good.

The chief problem with KBS, as the paper points out, is that harvesting the knowledge is expensive. With statistical techniques you can often give the computer a huge pile of data and let it crank. KBS require painstaking effort by experts to formulate what they know, and programmers (who may or may not be the experts) to come up with a representation for this knowledge, and then encode it. Both the “coming up with the representation” and the “encoding it” are significant undertakings. The paper also points out that these type of systems are prone to getting crufty, as more and more bits of knowledge are inserted. The knowledge base can become inconsistent, or it can require a series of specialized rules to account for exceptional behavior. This complexity quickly becomes unwieldy.

Grammars

Grammars were once upon a time the darling of cognitive scientists and computational linguists, and are useful tools for a variety of problems, particularly

problems that require one to represent structure and substructure. The paper mentions several problems with grammars, most of which I contest. First, the paper claims that while grammars are inherently hierarchical, not all music is, and so the grammars are ill-equipped to describe the non-hierarchical music. Improvisation is an example of this. I have my doubts about this statement; while improv. can certainly be thought of as less rigidly hierarchical than more formal structured music, it does maintain some basic hierarchical principles, doesn't it? Even if it doesn't, a modified grammar, with some kind of markup, could probably handle these "digressions" from larger structure. Still, it's true that the less hierarchical the domain is, the less suitable a grammatical approach would be.

The paper also points out that a grammar could generate a "large number [of] musical strings of questionable quality." This is undoubtedly true, but I hesitate to attribute this weakness explicitly to the grammar. One could probably do away with much of the bad musical strings by tuning the grammar more precisely, and by constraining the grammar using extra-grammatical techniques, which I'll address below.

Evolutionary Methods

Evolutionary methods are an intuitively clean way of manipulating musical "genomes." There are two main problems; the first is of representation: how is the music codified into a genome, exactly? Once again, deciding how to represent the problem will go a long way toward determining the quality of the solution. The more fundamentally difficult problem, however, is coming up with an appropriate fitness function. As the paper points out, one must either formulate some heuristic to make the quality judgment (which is, in a sense, the original problem all over again: if you could formalize what made a piece of music good you wouldn't need the genetic algorithm in the first place.) or else have a human in the loop, sitting in front of the computer screen listening to batches of genetically modified music. Considering how many iterations are probably required to produce anything of interest such a task would be mind-numbing indeed. Genetic algorithms are probably best utilized as part of a larger system.

Systems Which Learn

This is an odd category; in a number of non-trivial ways, all of the earlier classifications can be considered "systems which learn." The authors use this category mainly as a discussion of neural nets. Neural nets have enjoyed at least two golden eras; these eras end when people realize once again just how difficult it is to get them to solve anything besides toy problems. The authors mention various ANN critiques: representing time, requiring overly-simple problems, a difficulty in reproducing the data they're trained on, and a high degree of human involvement to create. Against this first criticism I can say that much recent work has been addressed, with mixed results. The fact is that it's just pretty damn hard to represent time in a neural net in any clean way. The toy-problem critique I mentioned already, and the high-degree of human involvement is even more true than the author lets on: it takes an incredible amount of craft to structure a network to perform even moderately-sophisticated tasks, more craft than science, even.

However, the critique that ANNs can't even reproduce the training set seems an unfair one. There is a degree of "muddiness" to learning with ANNs that practitioners are assumed to understand. ANNs are supposed to generalize to other classes of

problems. A variety of techniques can learn to spit back the training set, as indeed can ANNs, although they may not then be useful for much else besides identifying those particular instances. Criticizing them on this basis is like criticizing a pit bull for chewing off its owner's face: you get a dog bred specifically to be violent and aggressive, you aren't allowed to be surprised when it behaves the way it was engineered to behave.

Hybrid methods

The last class of systems is really no class at all; it's a chimera composed of already-mentioned techniques. The authors eventually propose that hybrids are the best hope for advances in musical composition, which seems obvious. When is a rigidly restricted solution domain ever better? The problem, of course, is that it's hard to figure out how to combine different techniques in a problem as general and demanding as music composition. Perhaps, given time, some particular combinations of techniques to address particular aspects of the problem will emerge.

After highlighting these various categories the paper goes on to describe design issues and evaluation criteria. Knowledge representation has already been discussed. Creativity is problematic for a variety of reasons: first, how, precisely, should we define it? This seemingly simple question has spawned an entire field of study and a huge number of wordy tomes as well as a number of interesting conundrums: how desirable is it to be creative, even if we had a good definition for what creativity meant? How aesthetically pleasing is creativity, anyway? How can we adapt the aforementioned techniques to be creative?

The authors close by suggesting that composition systems take performance into account, and incorporate musical ideas like "musical tension" and "expectation" and "musical closure." This seems like a fine idea, but it strikes me as re-stating a theme that's come up in the paper twice already: a system that composes must be able to "see" the piece it's composing at a reasonable level of abstraction; meaning, it should represent the piece in terms of elements appropriate to the nature of the piece. More briefly: if you represent the problem right, it's already half-solved.