Live coding machine learning?
Iván Paz & David McAndrews

Writing source code in real time to produce sound mixes both technology and artistic practice. Analysing live coding tools, instruments and technologies provides insight into how they shape the way we structure sound and musical ideas. For example, the source code becomes our notation language. It is the score of the piece, as the algorithms, functions, etc. describe sound unfolding in time. Different languages provide different conceptions of time. For example cyclical, in which a declared pattern repeats every cycle, provides a different conception than say, linear, in which events are conceived with a start, a duration and an end. A deep reflection on how technology shapes our music making can be found in (Magnusson, 2019), which looks at the technologies of material instruments, the symbolic notations of music and the signal inscriptions or recording mediums.

The digital nature of live coding brings new algorithmic possibilities to the core of the new instruments that are emerging alongside new technologies. These are artifacts that change by interacting with the performer, e.g Knotts, 2019, or that are able to perceive and adapt to their external reality by means of machine listening.

At the time of writing these lines, machine learning is being explored within live coding. While most early live coding performances used simple sound generators and processors, such as sine waves, filters, etc., today it is increasingly common to hear performances, even those starting *from scratch* (Villaseñor and Paz, 2020), using machine learning to perform specific tasks. For instance, creating clusters in a music database so that the performer can navigate an ordered space.

The use of machine learning algorithms has implications in live coding practice. For example, real time training v.s offline training implications can be viewed through the lens of the live coding ManifestoDraft, which positioned live coding within the digital arts back in 2004 and has been a reference for live coding practice. Here is a fragment:

We demand:

- Give us access to the performer's mind, to the whole human instrument.
- Obscurantism is dangerous. Show us your screens.
- Live coding is not about tools. Algorithms are thoughts. Chainsaws

are tools. That's why algorithms are sometimes harder to notice than chainsaws.

We recognise continuums of interaction and profundity, but prefer:

- Insight into algorithms
- The skillful extemporisation of algorithm as an expressive/impressive display of mental dexterity

Integrating machine learning into live coding raises such questions as how can machine learning algorithms be visualized within live coding? How do training times and/or size and nature of databases influence the design and use of algorithms? To what extent does on-the-fly machine learning exist with the current technology?

It is possible to use algorithms trained over large datasets, such as in melody generation. Sema, for example, is a playground for prototyping live coding mini-languages that integrates signal synthesis, machine learning and machine listening. It allows the use of models from TensorFlow (the end-to-end open source platform for machine learning). The training time of some models can take minutes, hours or days depending on the algorithm, the dataset, and the hardware. Once trained, the models can be fairly accurate, rarely surpassed by other systems. This approach reaches philosophical limits like the one suggested by Collins, 2016 in his paper entitled: "Towards Machine Musicians Who Have Listened to More Music Than Us". Indeed an algorithm can be trained over corpora of music that would take a human years to listen through.

Some artists have eschewed such large-scale datasets. Niklas Reppel opened his presentation at the fifth International Conference on Live Coding (Limerick 2020) with the question: Why small data? Stating the ideas behind the design of *megra*, a mini-language to make music with variable-order Markov chains (Reppel, 2020) he answered in the following way: Small data is a defined response to the current algorithms, which are mostly focussed on big data sets and specialized hardware, and therefore have long training times that won't fit into the live coding performance. *megra* is designed considering everything you can do with tiny datasets having real time feedback.

These approaches define extreme possibilities. Each has to embrace its necessary consequences. However, just as technologies condition our way of making music, live coding, being a well defined practice, is shaping technological developments, designing systems that learn, exploring the limits

and possibilities of machine learning algorithms from a creative perspective, analysing how they change through different training datasets and writing new ones that provide desired affordances.

This is an emergent field in which we hope these lines help to conceptualize the algorithms to come.

Bibliography

[1] Nick Collins. "Towards machine musicians who have listened to more music than us: Audio database-led algorithmic criticism for automatic composition and live concert systems". In: *Computers in Entertainment (CIE)* 14.3 (2016), pp. 1–14.

[2] Shelly Knotts. "Interactively evolving compositional sound synthesis networks". In: *Proceedings of the Live Coding Music Seminar*. IMPA. 2019, pp. 29–31.

[3] Thor Magnusson. *Sonic writing: technologies of material, symbolic, and signal inscriptions*. Bloomsbury Academic, 2019.

[4] Niklas Reppel. *Megra International Conference on Live Coding*. 2020. URL: `https://youtu.be/6dhvNrwQTRU?t=3839`.

[5] Hernani Villaseñor and Iván Paz. ""Live coding from scratch: the cases of practice in Mexico City and Barcelona"". In: *V International Conference on Live Coding*. 2020.