The Moving Target in Creative Interactive Machine Learning

Mark Cartwright
Interactive Audio Lab
Department of EECS
Northwestern University
Evanston, IL
mcartwright@u.northwestern.edu

Bryan Pardo
Interactive Audio Lab
Department of EECS
Northwestern University
Evanston, IL
pardo@northwestern.edu

Abstract
It is common for the teacher's understanding of a concept to evolve and change as they teach. This is especially common for creative tasks—preferred goals and methods can and should shift during the creative process. This can be problematic for interactively training a machine learning system to assist a creative task. Algorithms typically presume a constant goal and treat inconsistency in training data as unwanted noise. "Creative types" typically don't understand the internals of learning algorithms and cannot compensate for the weakness of the algorithms. We must develop methods better able to handle training data that represents a shifting goal or concept. Ideally, these approaches should incorporate a training paradigm that even novice, non-technical users can use effectively.

Author Keywords
interactive machine learning; concept evolution

ACM Classification Keywords
H.5.m [Information interfaces and presentation (e.g., HCI)]: Interaction styles; I.2.6 [Artificial Intelligence]: Learning

Introduction
Interactive machine learning (IML) is a machine learning approach that elicits feedback and examples from the user throughout the learning process. Researchers have shown

that IML can be an effective approach for enabling users with limited experience, technical knowledge, and physical abilities to accomplish creative goals that would typically be beyond their abilities. Examples include gesture and language controlled synthesizers for novices and musicians with disabilities [5, 7, 6]. By providing and/or rating examples, users of IML tools are able to interactively teach target concepts to the machine and overcome what were previously significant interaction barriers.

Over the past few years, we have developed several interactive systems that employ machine learning to make audio production tasks accessible to novices [2, 4, 3]. In that time, we have noticed a common trend with these systems—often the user's desired concept evolves during training, either becoming more refined or shifting to another concept altogether. This concept evolution is problematic for existing methods because it introduces inconsistent training data, which can cause learning algorithms to produce undesirable results. The learning objective is a moving target. This phenomenon also makes model evaluation difficult—if a system fails to model the desired concept, is it due to a bias or deficiency in the learning algorithm or is it due to inconsistent training data?

Other researchers have noted the issue of concept evolution as well. Kulesza et al. [8], who coined the term "concept evolution", found that expert users of an interactive binary classifier for web-pages often defined/refined their concept while interacting with the tool—even while most of them had clear initial concepts. Their data suggested that these concepts may have evolved as a result of seeing new training data (i.e. web-pages). Fiebrink et al. [5] found that users of their interactive gesture classifier also evolved the desired concept. Some of their users evolved their concept because they encountered limitations of the learning algorithm, and others evolved their concept because they encountered unexpected results that were preferred over their initial concept. This explains why we prefer the term concept evolution over concept drift, because in a creative task, the movement of the concept is often part of intentional rethinking rather than an unconscious or accidental process. While observed in non-creative tasks, we believe this problem is especially common in IML for creative tasks, since goals may not be as clearly defined and more subject to revision.

Examples of concept evolution in our research

In this section we review our interactive systems for audio production and describe instances of concept evolution.

SocialEQ is a system for audio equalization to which users teach descriptive adjectives of audio-equalization concepts (e.g. "warm") by listening to and rating examples on a continuous scale [2]. After teaching the system an adjective, a user commented:

> "I realized that I had two interpretations of 'hollow'. . . . After hearing the equalizer, those I designated as not 'hollow' ended up sounding more 'hollow'. . . "

This quote indicates that because their initial concept wasn't initially defined well, they contributed incorrect training data until the concept evolved and became more refined.

Another user commented:

> "I felt the software was training ME [*sic*] to give it a word that it could work with."

This sentiment is reminiscent of feedback given by users in [5]. Here, the user's desired concept evolved as they became acquainted with the potential limitations of the software.

Mixploration is a system for audio mixing in which users rate mixes and the system uses those ratings to learn a refinement controller. In our user study of the system, 48% of the participants indicated that their mixing objectives (i.e. desired concept) had changed during the interaction. In this system, users didn't interact with the learned model until they had finished rating. Therefore, their concept evolution was likely due to either the exposure to new training examples, the act of evaluating examples, or both.

SynthAssist is a system for audio synthesizer programming in which users provide an initial audio example or vocal imitation of their desired concept and then refine their concept by evaluating rounds of suggestions given by the system. While we have not conducted a thorough user study of this system, we have demoed it on several occasions in which users have commented that their concept evolved during the interaction. Several users commented this was due to one of the following: becoming acquainted with the sound/limitations of the synthesizer, hearing another sound which caught their interest, or hearing their desired concept realized but concluding that another concept was more appropriate.

In all of these systems, the goal is for novice users (without technical expertise) to train the system by evaluating many audio examples. How can we correct concept evolution in these and similar regression scenarios?

Discussion

We predict that existing solutions for similar problems will not work well in such systems. For example, many of the users in the Fiebrink et al. study directly evaluated their models and improved them by adding or subtracting from their training data [5]. However, this approach is likely too tedious for systems with many (e.g. 50) audio training examples. Kulesza et al. addressed the problem of concept, evolution by introducing a labeling technique called "structured labeling" which allows users to create groups within 'yes','could be', and 'no' labels. The approach of Kulesza et al. [8] is for binary, not continuous labels. While this could potentially be adapted for continuous labels, such an approach would likely be less effective for audio since audio segments cannot be re-evaluated as quickly as glancing at images.

One potential solution could be to utilize active learning techniques—not to pick new potentially informative *unlabeled* examples but rather to pick potentially informative *already labeled* examples for re-evaluation (additionally filtering or weighting the criteria by the time elapsed since labeling to avoid re-evaluating examples belonging to the new concept). This could limit the number of items a user has to evaluate—an important criteria for audio and other complex or temporal examples. However, [1] summarizes that users want to do more than simply be "oracles" and prefer more engaged feedback mechanisms.

Therefore, a solution to the problem presented in this paper should:

- Clarify the user's desired concept and its related training data
- Support ML algorithms that use continuous data labels (e.g. regression)
- Be accessible to novice users without technical knowledge

- Minimize the number of additional examples to be evaluated
- Engage the user

We look forward to discussing this problem and potential solutions at the workshop as well as related problems in human-centered machine learning.

Acknowledgments

This work was supported by NSF Grant No. CHS-1420971.

References

[1] Saleema Amershi, Maya Cakmak, W Bradley Knox, and Todd Kulesza. 2014. Power to the people: The role of humans in interactive machine learning. *AI Magazine* 35, 4 (2014), 105–120.

[2] Mark Cartwright and Bryan Pardo. 2013. Social-EQ: Crowdsourcing an Equalization Descriptor Map. In *Proceedings of the 14th International Society for Music Information Retrieval Conference*.

[3] M. Cartwright and B Pardo. 2014. SynthAssist: Querying an Audio Synthesizer by Vocal Imitation. In *Proceedings of the Conference on New Interfaces for Musical Expression*.

[4] M. Cartwright, B Pardo, and J Reiss. 2014. Mixploration: Rethinking the Audio Mixer Interface. In *Proceedings of the International Conference on Intelligent User Interfaces*. ACM.

[5] Rebecca Fiebrink, Perry R Cook, and Dan Trueman. 2011. Human model evaluation in interactive supervised learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 147–156.

[6] Cheng-Zhi Anna Huang, David Duvenaud, Kenneth C Arnold, Brenton Partridge, Josiah W Oberholtzer, and Krzysztof Z Gajos. 2014. Active Learning of Intuitive Control Knobs for Synthesizers Using Gaussian Processes. In *Proceedings of the International Conference on Intelligent User Interfaces*.

[7] Simon Katan, Mick Grierson, and Rebecca Fiebrink. 2015. Using Interactive Machine Learning to Support Interface Development Through Workshops with Disabled People. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 251–254.

[8] Todd Kulesza, Saleema Amershi, Rich Caruana, Danyel Fisher, and Denis Charles. 2014. Structured labeling for facilitating concept evolution in machine learning. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*. ACM, 2557238, 3075–3084. DOI: http://dx.doi.org/10.1145/2556288.2557238