Motivated Learning in Human-machine Improvisation

Peter Beyls
The School of Arts, University College Ghent, Belgium
peter.beyls@hogent.be

ABSTRACT

This paper describes a machine learning approach in the context of non-idiomatic human-machine improvisation. In an attempt to avoid explicit mapping of user actions to machine responses, an experimental machine learning strategy is suggested where rewards are derived from the implied motivation of the human interactor – two motivations are at work: *integration* (aiming to connect with machine generated material) and *expression* (independent activity). By tracking consecutive changes in musical distance (i.e. melodic similarity) between human and machine, such motivations can be inferred. A variation of Q-learning is used featuring a self-optimizing variable length state-action-reward list. The system (called *Pock*) is tunable into particular behavioral niches by means of a limited number of parameters. Pock is designed as a recursive structure and behaves as a complex dynamical system. When tracking systems variables over time, emergent non-trivial patterns reveal experimental evidence of attractors demonstrating successful adaptation.

Author Keywords

Interactive music systems, motivation, machine learning

CCS Concepts

• Human-centered computing ~ Human computer interaction (HCI) • Computing methodologies ~ Reinforcement learning • Applied computing ~ Sound and music computing

1. INTRODUCTION

According to Rowe, "interactive music systems are those whose behavior changes in response to musical input" [1]. Rowe identifies three dimensions assisting classification of interactive music systems: (1) score driven vs. performance driven systems, (2) response methods as transformative, generative or sequenced and (3) the instrument vs. the player paradigm, i.e. aspire extended instrumental control/expression or aiming to construct an artificial player endowed with unambiguous musical intelligence and personality. Work described in this paper builds on the latter orientation and is motivated by a desire to build an artificial partner engaging in open improvisation with a human interactor. More specifically, our goal is to develop an interactive music system based on the seemingly conflicting requirements of affording at once coherent performance and unpredictable behavior. We expect the system to pursue a personal agenda and display a particular personality. In addition, it should blend external influence from a human performer in its independent real-time behavior. Human and machine improvise in a common ground and make for a larger complex dynamical system where human intensions interlock with activity in systems components. This orientation requests a comparative study of responsive vs. interactive systems. Responsive systems typically react to human actions by playing a response selected from a palette of predesigned responsive options – therefore the interaction is unbalanced and the machine assumes the role of a passive participant. Mapping [2] creates variable (possibly stochastic) relationships between features in human input and the complexity of system responses. However, at a higher level of abstraction, the psychological state of the performer remains unknown to the machine so it cannot develop a deeper understanding of human intentionality, no capacity to anticipate human behavior and to generate appropriate feedback – in short, it cannot adapt, for example, to the idiosyncratic language of a particular human performer.

In contrast, truly interactive systems typically display more life-like behavior with human and machine at equal levels of authority; communication is characterized as a form of common initiative and shared control. Human and machine engage in a more symbiotic relationship and turn into a complex macroscopic hybrid performance agency [3]. System complexity results from the expression of variable affinities between system components, adaptations of cognitive preferences for listening and playing in the human performer. Given a machine capable of acquiring a live image of the qualities of a human performer, it can learn as to provide more optimized musical arguments underpinning future interactions. Following adaptation, systems behavior becomes dynamic rather than simply procedural, with deeper reciprocal integration of human and machine actions thereby affording more mutually rewarding interactions.

This paper introduces an interactive computational framework (named *Pock)* that supports open human-machine improvisation by combining the concepts of learning and motivation. Pock uses a variation of temporal difference learning by (1) managing a variable length SAR-list (state-action-reward arrays) and (2) a two-step reward update procedure combining immediate activation/inhibition scaling of rewards and actual Q-learning [4]. We follow the behaviorist motivation theory advocated by Mook [5] viewing motivation as initiated by external influence rather than internal processes.

In terms of background, we take inspiration from the notion of interactive enaction since the enaction paradigm addresses cognition, as the history of structural couplings between an organism and its environment – autonomy and cognitive functionality is a consequence of exposure to a dynamic environment. As the environment impacts on the organism, the latter will adapt and, in turn, impact the environment so both co-evolve and co-exist as a macroscopic system of higher complexity. Some features of enaction-based systems [6] are exceptionally significant for the work described here:

a. The interaction starts from scratch, we avoid a priory representations of the environment i.e. the human performer – for example, a related rule-based

NIME'18, June 3-6, 2018, Blacksburg, Virginia, USA.

mapping scheme elaborated in view of exposure to a particular performer

b. Similar to plasticity in the brain, the organism absorbs external agitation and gracefully adapts through some form of structural adjustment

c. Organism and environment co-evolve within a recursive relationship: modifications effected by the organism on the environment will in return enforce modifications of that organism.

In terms of our implementation, enaction is viewed as an instructive metaphor to draw on principles of grounded biology in artificial systems while recognizing the irreversible nature of interaction in organism-environment coupling and interaction based on recursive interdependence of processes to maintain themselves.

2. LEARNING AND ADAPTATION

2.1 Machine learning background

Informally speaking, in the process of learning, we acquire knowledge and skills while trying to solve a particular problem so we will be in a better position to address the same problem at future occasions. Learning informs for predicting the future based on past experiences, therefore, machine learning (ML) algorithms must generalize from exposure to specific situations.

More formally, a ML program is "the process of estimating a model that's true to the real-world problem with a certain probability from a data set (or sample) generated by finite observations in a noisy environment" [7] This implies ML to discover the *relationships* between features in a real-world problem and its long-term behavior. ML faces two basic difficulties; the number of dimensions in a real-world problem is typically massive while the sampling set is small and, real-world problems normally exhibit highly non-linear behavior.

A classic text by Barto and Sutton [8] provides a clear introduction to the complex scientific field of ML. Globally, a distinction is made between supervised, unsupervised, semi-supervised and reinforcement learning. For example, in supervised learning we might train a network with a vocabulary of bodily gestures and expect the network to classify similar perceived gestures later on. Unsupervised learning also involves exposure to training examples, however the algorithm must develop an internal representation by addressing the complexity of the training data – for example in hierarchical pattern recognition and automatic feature recognition. Semi-supervised learning combines such feature detection to expand the quality of the predictions in supervised learning. In reinforcement learning (RL), the system learns from interacting with a particular dynamic environment while trying to pursue a particular goal. As a consequence of actions exercised on the environment, the algorithm receives immediate positive or negative feedback i.e. reward or penalty. RL aims to maximize the rewards by developing an agenda of successful actions leading to the discovery of proficient behaviors in terms of a particular goal. Unsurprisingly, RL is crucial in the field of autonomous robotics. Q-learning is a unique form of RL as detailed in section 3.

2.2 Machine learning in interactive music systems

Over the last few years, ML technology found a wide range of applications in creative music production, musicology, music analysis and understanding. Examples include problems of music information retrieval (MIR) such as polyphonic music classification [10] performer identification in audio recordings [11], style analysis viewed as the extraction of musical signatures from a musical corpus [12], composition using time series probability density techniques [13] and more recently, the design of a generative model from a deep artificial neural network architecture for integrated composition of harmony and melody [14].

Real-time interactive music systems listen to a human improviser as to acquire live stylistic information characterizing the performer and create and update knowledge representations on the fly. Techniques include histograms as in *Voyager* [15] Hidden Markov Models basic to the *Continuator* [16] and factor oracle algorithms used in the *OMax* system [17]. All these systems afford advanced human-machine experiences where both partners develop truly interactive relationships and mutual dependencies and perhaps a common agenda – human and artificial performers typically engage in conversational affiliation – machine responses echo musical stylistic aspects of human input.

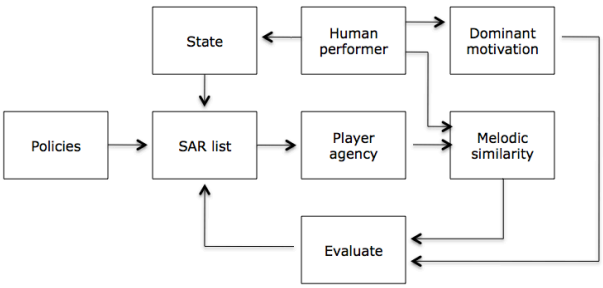

Figure 1: Schematic overview of computational architecture

Specifically, interactive systems investigating machine learning include *BoB*, an artificial improviser using unsupervised learning [18], an actor-critic model for jazz improvisation implemented as a recurrent neural network with rewards from explicit rules derived from jazz theory [19] and Collins' *Improvagent* which embodies the implicit affordance of leaning; it is "a musical agent which seeks to improve itself through constant learning" [20, p. 150]. Similar to our orientation, *Improvagent* views human behavior as sequences of observed environmental states but uses variants of Sarsa(λ) learning and figures a double reinforcement strategy involving predictions and consequences.

Interestingly, as part of a theory of expressive interactions, Leman develops the concept of cognitive-motivational loops – driven by processes of affect, energy and motivation, this loop creates a global equilibrium state amongst contributing processes, a state that is empowering, energizing and reinforcing [21]. Leman views expressive interactive machines as capable of intercepting and changing information and feeding this back into the loop. In addition, reward processing is recognized as essential to inform the cognitive-motivational loop, which in turn, affects predictive processing and motivation.

3. Q-LEARNING

Consider an algorithm learning to fly a helicopter facing heavy weather [22], the pilot exercises activity over a many control parameters, simultaneously accommodating real-time multimodal feedback. Control actions are successful if the machine remains stable, under control and at a particular desired distance from the ground. However, we don't know the actual impact of an action until after it is applied. In addition, we don't necessarily know all the details of the current situation. Consequently, continuous exploration of the problem

at hand is encouraged while still recalling profitable past actions.

Let's identify a striking correspondence between flying a helicopter and open musical improvisation. Metaphorically speaking, two improvising individuals fly through a common multi-dimensional musical space, connecting in many ways while equally trying to remain at a given musical distance – they connect and disconnect intermittingly, a wave-like pattern of mutual affinities emerges spontaneously over time. This notion of 'musical distance' is key to the learning algorithm in Pock and is explained briefly in section 4.1.

Q-learning [9] is a form of model-free learning and works by assigning values to state-action pairs; the environment is represented in terms of possible states and a collection of actions – the algorithm learns the value (efficiency) of each prospective action in each possible state while trying to achieve a particular goal. In a complex environment, it has no clue what action to select, so it takes speculative action by exploring the state-action space. One gets feedback *after* the action is taken, the new state is evaluated as desirable or not, a value to the state-action pair is attributed accordingly. Gradually, after trying many actions given many different circumstances, we gather an impression of which action works well given a particular state. Later on, the most promising actions are selected: the algorithm progresses from initial exploration to ever more exploitation i.e. selecting actions that proved successful in previous interactions. Equation 1 specifies how the value previous state-action pair is updated:

```
Q(s, a) = Q(s, a) + (alpha * (reward(s,a) + max(Q(s')
- Q(s,a)))   [eq. 1]
```

```
With s: current state, a: the current action,
alpha: the learning rate, s': the previous
state, max(Q(s') - Q(s,a)): the maximum value of
Q for all possible actions in the next state.
```

`Alpha` is the learning rate `(0 < alpha < 1)`, with `alpha` close to zero, the agent prefers only immediate rewards, with alpha nearing the value of one, the agent will favor considering delayed rewards. The Q-value for performing action `a` in state `s` is the difference between the actual reward `reward(s,a) + max(Q(s')` and the expected reward `Q(s,a)` scaled by the learning rate `alpha`.

States and actions are often represented in a 2D matrix however, given large state spaces, this becomes problematic [23]. Our implementation turns to a dynamic data structure that adapts according to the perceived complexity of the environment, as explained in section 4. Learning follows from trial-an-error interactions with the environment, so intuitively, we understand it as tightly linked to both speculative computing and the nature of real-time musical extemporization, for instance, within the idiom of free jazz improvisation. Our research views a single human interactor as "an unpredictable environment" exposed to a synthetic performance agency equipped with a ML algorithm offering dynamic adaptation and autonomy to the global system.

4. IMPLEMENTATION

Pock is implemented in SuperCollider [24] in a style of object-oriented programming by means of a hierarchy of interlocking modules. Pock features an adaptive ear listening to symbolic input (MIDI), a distributed player agency thought of as a small instrumental performance ensemble, a variable collection of musical responses (considered an assembly of policies in the learning process) and the reinforcement-learning component. Figure 1 shows global system layout revealing multiple information flows between system modules. In particular, coordination of learning through the SAR-list is conditioned by data from the evaluator in relation to the current state.

4.1 Listening

The Ear module acquires user input via MIDI (pitch-to-MIDI conversion of live audio input) into a FIFO data structure typically tracking the 16 most recent events, performs segmentation of the continuous input stream and computes relative contextual change by evaluating the difference of the last and previous segment. Specific features in support of learning include (1) tracking quantity and quality of the current input stream and (2) tracking human motivation. Quality refers to the complexity of user input, computed by considering diversity in the dimensions of pitch, velocity, duration and inter-onset time of the incoming events. Quantity level reflects how much information comes in, if any, i.e. density of the input stream is considered. The Ear also holds a task performing continuous tonal inference in addition to an extensive range of analysis functions resulting in a 48-bit feature vector reflecting the overall complexity of its current buffer content.

To assist learning, the Ear also tracks human motivation in two competing dimensions denoted as *integration* and *expression*. Integration level reflects the human performers' willingness to connect to the machine performer by minimizing musical distance. For example, if current human input is very similar to the last machine statement, the integration level is incremented proportionally. In case the distance increases, human and machine are drifting apart and human expression level is incremented proportionally. Thus, both values are subject to activation or inhibition as a function of musical distance (human-machine similarity) over time – the underpinning assumption being the gestalt principle of *proximity*; integration and expression reflects melodic distance. All values in the Ear are normalized to fit 0 ~ 1.0. In summary, the ear features sensors to address human-machine similarity, track quantity and quality of user input, manage levels of integration and expression and, finally, compute relative change (magnitude of the contextual step) in the interval of last and previous user events.

4.2 Player Agency

A Player Agency holds a variable number of player agents, one of which acts as a reference agent, it performs as a reference melody to the Ear, as described above. An actual reference agent is recruited from a pool of potential agents available in the Policy Agency. Agents are complex MIDI player objects featuring start-, repeat- and stop-functions with access to a large library of analysis and processing functions (outside the scope of the present paper). Agents can act as social objects expressing variable mutual affinities when configured in an agency. In addition, an agent can borrow stylistic material from a human performer; the external input might then propagate as to influence the behavior of all current agents in the agency.

4.3 System parameters

Since we expect our system to expose life-like behavior, we avoid the notion of mapping (and its inherent restrictions) i.e. the creation of pre-designed links between user input and system response [25]. Still we are looking for an implicit cognitive link between the human and machine partners – a delicate non-trivial affinity is to be perceived in a form of reciprocal interaction. Apparently, musical initiative is seemingly floating back and forth through the forces of mutual

influence, however, without any one being in control. Pock features a set of global high-level parameters allowing tuning the system into particular behavioral niches.

For instance, *connectedness-level* specifies how much information is gathered from the human performer; variable individual windows of intervals are extracted form the Ear's buffer in terms of pitch, loudness, duration and rhythmic articulation (inter-onset times). High levels initiate relatively large coordinated buffer areas (equal size and same read-pointers for the four dimensions) and borrowing most recent data (close to the tail of the buffer). Lower levels imply independent areas of smaller and different sizes and distinctive read-pointers anywhere in the buffer. When Connectedness-level equals zero, Pock plays without external influence.

Responsiveness-level conditions the willingness and swiftness of the system to respond to human input and *energy-level* influences the average number of events generated.

Continuity-level specifies the probability of generating new material vs. the reinjection of variations of existing material while computing a nascent response and *complexity-level* informs the number and relationships between the active agents configured in the current Player Agency. Collectively, complexity-level and energy-level condition response density. High complexity-levels produce a dense population of auxiliary player agents all reflecting stylistic aspects of the reference agent. A process of orchestration takes place; (1) single reference events are split into multiple new events or (2) parametric features of sequential reference events are grouped in to a single new event. Both processes compute appropriate durations of the secondary events, which guarantees perfect synchronization of all agents playing in parallel. All agents hold a private MIDI instrument and functionality to condition generative material to fit the instrument's range.

Complexity-level controls the number (0 ~ 5) of instantiated parallel agents in the global player agency – their content derived from and sounding in parallel to the source reference agent.

Autonomy-level conditions system behavior in case human input is minimal or completely missing. In addition, *bootstrap-probability* defines the chance for a fresh set of random policies to be computed potentially creating gradual stylistic discontinuity in system actions.

To conclude, two parameters directly inform the learning algorithm: the *learning-factor* (alpha in equation 1) specifies the relative value of delayed vs. immediate rewards (0 ~ 1.0) and the *exploration-rate* (also valued 0 ~ 1.0) sets the chance to learn without following the currently optimal policy. Balancing exploration (investigating a random response) vs. exploitation (making good used of previously successful responses) and handling a variable length, self-optimizing SAR-list rather than a conventional matrix representation are experimental specifics to our implementation.

4.4 Learning in Pock

Learning is coordinated from the *SAR-list*, a variable length collection of State-Action-Reward entries. A single SAR combines a particular state, the action taken in the presence of that state and the associated reward gathered after performing that action. A state reflects the current condition of the environment. A core problem in reinforcement learning is to capture the complexity of a typically high dimensional environment in a reduced number of states – Pock computes the current state (0 ~ 9) as the integer mapping of the current

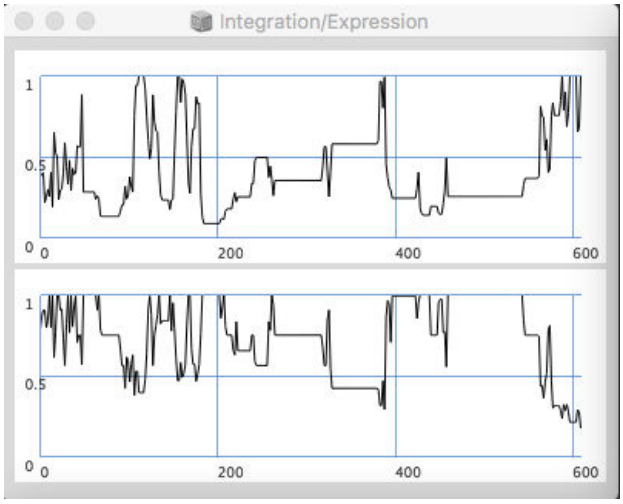

Figure 2: Competition between the two motivations of integration (top) and expression

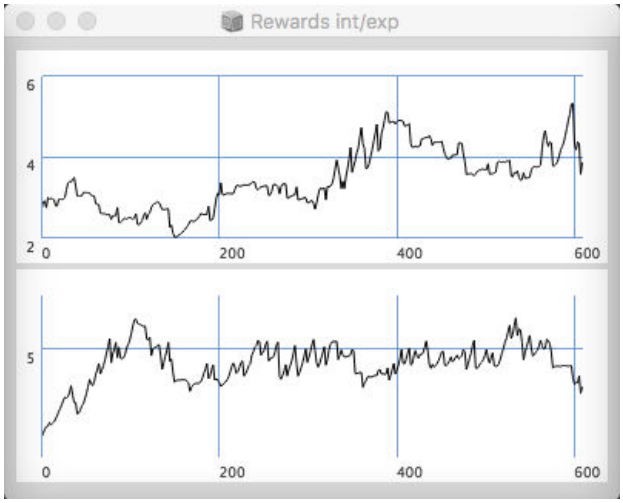

Figure 3: History of interaction rewards for two motivations; integration (top) and expression

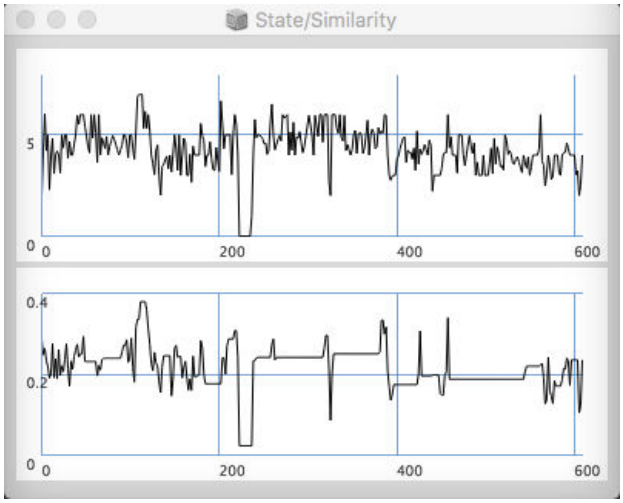

Figure 4: History of the perceived state (top) and melodic similarity

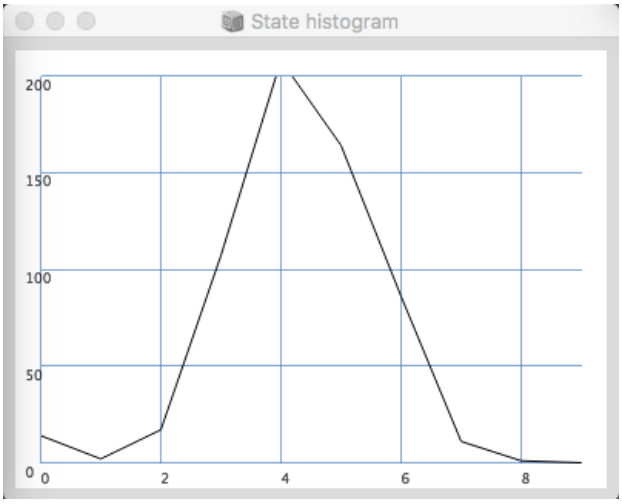

Figure 5: Perceived states histogram

human-machine distance in an adaptive window of minimum and maximum distances, the sensing window shrinks and expands according to the current distance. Windows contract when the sensed value is inside as to become more sensitive i.e. gradually focus on that value, windows expand as to accommodate values outside its current limits. Pock keeps counters tracking the frequency of all perceived states.

Coordinated sequential system activity is organized as follows:

a. Collect the current state *after* the agency completed performing its current policy – a short (typically less than 8 seconds) musical statement and consults the Ear to retrieve the presently dominating human motivation (integration or expression).

b. Collect a new policy; a response in the light of the apparent state i.e. compute the SAR-index as a function of current state and current motivation; Pock collects all list entries for this state. If none available (the first time this state is perceived) we select a random policy (any agent from the policies), attribute a small random reward (to be progressively adjusted in future interactions) and add a new entry to the SAR-list. However, just before taking this action, we optimize the SAR-list by checking the current number of entries, if beyond the capacity (typically 8) the weakest entry (summing the two rewards, integration and expression) is removed creating room for a fresh entry.

   In case the SAR-list is not empty, there are two orientations for selecting a policy; (1) retrieving a policy from the list that proved proficient in previous interactions i.e. we engage in exploitation of the learned options, typically, the action with the highest reward is retrieved, or (2) we choose at random from Policies (a small container of random policies initially computed at system start time) in a wish to explore all possible options. Finally, the reference agent is designated to perform the chosen action.

c. Compute the events performed by the player agency by considering all system parameters, the selected response and the current contents of the Ear's buffer. Parameters influence the complexity and responsiveness of the actual feedback as explained above. The responsiveness parameter controls the task running the player process; the task delay is proportional to the reciprocal of (1.0 - responsiveness (0.1 ~ 1.0)) multiplied by the total duration of the response – therefore, high levels of responsiveness creates overlapping responses.

d. Once the player agency performed its response, the reward is computed based on the evaluation of the change in human-machine similarity (similarity between the reference agent and the current events in the ear's buffer) as reflected in the new state:

```
newState = remap(currentSim, minSim, maxSim,
0, (nrStates-1), quantize);

deltaSim = currentSim – previousSim;
```

   The message `newState` returns an integer in the range 0 ~ 9 by remapping the current similarity in the space defined by minimum and maximum similarities. In addition, `deltaSim` documents the signed magnitude of the interval of the current similarity `cSim` and the previous similarity `pSim` – the value gathered in the previous process cycle. The value of `deltaSim` is critical in updating the reward, for example, if human-motivation equals 'integration' `and deltaSim < 0`, we receive confirmation the most recent machine response was indeed helpful in sustaining this motivation. According the these criteria, the rewards of the current SAR-list entry are updated:

```
if HMotiv == expression && DeltaSim < 0 ||
HMotiv == integration && DeltaSim > 0 then
   Sar-list[index].reward[1] = Sar-
list[index].reward[1] * 1.05
   Sar-list[index].reward[0] = Sar-
list[index].reward[0] * 0.95
else
if HMotiv == expression && DeltaSim > 0 ||
HMotiv == integration && DeltaSim < 0 then
  Sar-list[index].reward[0] = Sar-
list[index].reward[0] * 1.05
  Sar-list[index].reward[1] = Sar-
list[index].reward[1] * 0.95
```

   Rewards (array with two values, integration-reward and expression-reward) are scaled up or down according to the sign of the interval in similarity. Consequently, within the SAR-list, we search for the highest reward `maxRew` of any action that was taken before while facing the `newState` and update according to eq. 1.

```
maxRew = this.getMaxReward(newState, HMotiv);
p = [\int, \exp].indexOf(HMotiv);
Sar-list[index].reward[p] =
       min(1.0, Sar-list[index].reward[p] +
(learningFactor * maxRew));
pSim = cSim;
```

e. Finally, according to the Bootstrap-probability parameter (a typically very low value), a completely new random list of Policies might be computed signaling a break in system continuity and creating a fresh ground to accommodate human-machine interaction. The final state also handles extensive bookkeeping, in particular, tracking many values documenting systems behavior aiming future analysis and visualization.

5. DISCUSSION AND OUTLOOK

Figures 2 to 5 capture system activity over 600 learning cycles and offer insight on the relationships between complementary behavioral features. For example, figure 2 shows the two

competing drives of integration and expression emerging from the interaction process. Expression levels globally dominate significantly, flat areas signal no human input. Figure 3 considers actions in the SAR-list and depicts the sum of the accumulated rewards for the purpose of respectively integration and expression for all current actions – a global value reflecting the system's present ability to comply with external human pressures. Spiking behavior is evident, however strong tendencies equally emerge – expression capacity increases steadily at the very beginning, integration capacity increases at about generation 300. Overall, a wave-like pattern reminiscent of Brownian motion is clearly apparent. Comparing figures 2 and 3, considering integration data depicted in the upper panes within respectively the frames 200 to 400 and 400 to 600, an incremental tendency is clearly visible providing evidence of the systems' capacity to gradually develop musical functionality in agreement with the nature of external human pressure. Considering the lower pane in figure 2 showing expression data, persistently high amplitudes though smaller oscillations are observed, coherent with the expression rewards circling around 0.5, probably signaling (1) the drive of expression proving inherently stronger than integration and (2) the lack of diversity in the algorithm generating random melodies. Figure 4 shows the history of the perceived state (0~9) and the melodic similarity between human input and the melody performed by the reference agent. The states histogram in figure 5 reveals a normal distribution, Gauss-like curve with the frequency of states 0 to 9 observed: (14, 2, 17, 107, 208, 164, 86, 11, 1, 0).

Obviously, many more experiments are required to appreciate and understand the complexity and behavioral scope of interactive system using this type of machine learning. As always, we need to find a working balance between the resolution of the analysis algorithm (i.e. the number of different system states), the number of available policies, the capacity of the SAR-list, and finally, the number of interaction cycles required to articulate the implied behavioral dimensionality of the system. In terms of computational efficiency, while facing the challenge to learn quickly i.e. within the time span of minutes, more robust algorithms are needed to (1) reduce the infinite dimensionality of human intentionality captured through sound and (2) following the principle of 'influence' rather than 'control' – more complex and flexible generative algorithms for generating machine responses by subtly blending human input and native machine patterns.

6. REFERENCES

[1] Rowe, R *Interactive Music Systems*, The MIT Press, Cambridge, MA (1993)

[2] Hunt, A Wanderley, M and Paredis, The Importance of Mapping in electronic instrument design, *NIME 02 Proceedings*, Dublin, Ireland, 2002

[3] Beyls, P Towards Symbiotic Human-Machine Interaction, *Proceedings of the Generative Arts Conference*, Ravenna, Italy, 2017

[4] Watkins, C and Dayan, P Q-learning, *Machine Learning*, 8, 279-292 (1992) Kluwer Academic Publishers, 1992

[5] Mook, DG. *Motivation: the organisation of action*. W. W. Norton and Company, Inc, New York, 1987

[6] De Loor, P Manach, K and Tisseau, J *Enaction-based Artificial Intelligence: Toward Co-evolution with Humans in the Loop*, Minds and Machine, nr. 19, pp 319-343,2009

[7] Machine Learning: State of the Art, *IEEE Intelligent Systems,* Nov/Dec, 2008

[8] Sutton, R., and Barto, A. G.: *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998

[9] Watkins, J. C. H., Dayan, P. Technical Note: *Q*-Learning. *Machine Learning* 8: 279-292, 1992

[10] Ruben Hillewaere, Bernard Manderick and Darrell Conklin: Melodic models for polyphonic music classification, *Second International Workshop on Machine Learning and Music*, Bled, Slovenia, 2009

[11] Rafael Ramirez and Esteban Maestre: A Framework for Performer Identification in Audio Recordings, *Second International Workshop on Machine Learning and Music*, Bled, Slovenia, 2009

[12] Cope, D *Computers and Musical Style*, Madison, WI: A-R Editions, 1991

[13] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. *Proceedings of the 29th International Conference on Machine Learning*, (29), 2012

[14] Allen Huang and Raymond Wu, Deep Learning for Music, *arXiv:1606.04930 Preprint*, 2016

[15] Lewis, G Too many notes: computers, complexity and culture in Voyager, *Leonardo Music Journal*, 10:33-9, 2000

[16] Pachet, F The Continuator: Musical Interaction with Style *Journal of New Music Research*, 3:333-41, 2003

[17] Assayag, G., et al. Omax Brothers: a Dynamic Topology of Agents for Improvization Learning. *Workshop on Audio and Music Computing for Multimedia,* ACM Multimedia, Santa Barbara, 2006

[18] Thom, B BoB: an Interactive Improvisational Music Companion, *Proceedings of the 4th International Conference on Autonomous Agents*, Barcelona, Spain, 2003

[19] Franklin, JA and Manfredi, V Nonlinear Credit Assignment for Musical Sequences, *Second International Workshop on Intelligent Systems Design and Applications*, pp. 245-250, 2002

[20] Collins, N Reinforcement Learning for Live Musical Agents, *Proceedings of the ICMC*, Belfast, Ireland, 2008

[21] Leman, M *The Expressive Moment*, The MIT Press, Cambridge, MA, 2016

[22] Ng, A Kim, J Jordan, M and Sastry, S Autonomous helicopter flight via reinforcement learning, *Advances in Neural Information Processing*, 2004

[23] Tuyls, K Maes, S and Manderick, B Reinforcement Learning in Large State Spaces, RoboCup 2002, *Springer Lecture Notes in Computer Science*, vol. 2752, 2002

[24] Wilson, S Cottle, D Collins, N and McCartney, J *The SuperCollider Book*, The MIT Press, Cambridge, MA, 2011

[25] Chadabe, J The Limitations of Mapping as a Structural Descriptive in Electronic Instruments, *NIME 02 Proceedings*, Dublin, Ireland, 2002