Neurons: An Interactive Composition Using a Neural Network for Recognition of Playing Techniques

Artemi - Maria Gioti
Institute of Electronic Music and Acoustics (IEM)
University of Music and Performing Arts Graz, Austria
gioti@iem.at

Abstract

This paper describes a machine-listening based system developed for an interactive composition for soprano saxophone and electronics. The auditory processing stage of the system consists of a feedforward Neural Network trained to perform real-time recognition of different playing techniques (single notes, multiphonics, air tones and slap tones). This classification algorithm is embedded in four interaction scenarios that entail different compositional instructions and listening modes, including selective listening modes and a non-listening state. The integration of a classification task in the auditory processing stage of the system has the purpose of shifting the focus of machine listening from sensory (signal-level features) to symbolic information (composer-defined sound classes), enabling the design of idiosyncratic agent behaviors in the context of composed, scenario-based sonic interaction.

Introduction

Assuming Paine's (2002) conversational model of interaction (i.e. two humans engaging in a conversation) and the fundamental, yet often overlooked, distinction between responsiveness (*re-action*) and *interaction* (Paine 2002; Drummond 2009), most examples of human-computer interaction fall into the category of reactive, rather than interactive systems. Paine (2002) suggests that cognition is a requirement for interactivity and claims that most computer music systems are in fact reactive and not interactive, because they lack the element of cognition. Machine listening, as a simulation of a human cognitive process, has therefore unsurprisingly become the main modality for human-computer interaction in interactive performance and improvisation systems. In these systems, "listening" functions as a two-way communication channel between the musician and the computer, allowing both parties to perceive each other's actions through auditory information.

According to Rowe's taxonomy of interactive music systems (Rowe 1993), such systems can be described as performance-driven systems that follow the player paradigm – as opposed to the instrument paradigm. The majority of these systems were designed for human-computer improvisation and only a few have been used in individual compositions (Young 2007; Van Nort et al. 2009; Smith and Deal 2014).

The system presented in this paper falls under the latter category and was developed for an *interactive composition* for soprano saxophone and electronics. An *interactive composition* is a composition involving real-time interaction and mutual adaptation between a musician and a software agent, and entailing one or more *interaction scenarios*, pre-scripted in terms of sound material and interaction affordances. Both compositional instructions (i.e. score) and agent behaviors are summarized under the term *scenario*, which in this context takes a different meaning than its use in the context of human-computer improvisation (a predefined temporal structure used to guide improvisation (Nika et al. 2017)).

Along with "composed" improvisation and "comprovisation" (Dudas 2010), interactive composition is another point on the spectrum between free improvisation and "fixed" composition. What distinguishes it from free and "composed" improvisation is idiosyncratic (composition-specific) agent behavior in combination with a non-linear score, consisting of both descriptive and prescriptive notation (i.e. *sound descriptions* and *prescribed actions*, to be performed in response to the software agent's "actions"). A significant difference between an *interactive composition* and the act of *comprovisation* or *interactive composing* (composing while performing with an interactive system (Chadabe 1984)) is that *in the case of an interactive composition the act of composition precedes that of perfor-*

mance and does not coincide with it. What takes place in real-time is the interaction between the musician and the software agent and not the compositional process itself. Furthermore, composer and performer are not necessarily the same person (hence, the need for a score).

An example of such an interactive composition is described in the following section. The auditory processing stage of the software agent consists of a feedforward Neural Network trained to recognize four predefined sound classes (single notes, multiphonics, air tones and slap tones), allowing the software agent to interact with the musician on the basis of *symbolic* music information (a dictionary of sound classes defined by the composer), obtained through lower-level *sensory* information (signal-level descriptors). The results of this instant recognition are stored, enabling the agent to deduce information regarding the texture variability of bigger sections of the piece. Information collected in the auditory processing stage of the system is used to control the parameters of a signal processing and sound synthesis algorithm. Correspondingly, the musician is asked to adapt to the sound output of the computer in real-time, by interpreting a non-linear score[1].

Neurons

The Neural Network

The use of machine learning, and specifically unsupervised learning algorithms such as clustering algorithms, Self-Organizing Maps (SOMs) etc., in the auditory processing stage of interactive music systems is becoming increasingly common (Thom 2000; Young 2007; Collins 2008 and 2011; Lévy et al. 2012; Yee-King 2011; Smith and Deal 2014; Tatar and Pasquier 2017; Gioti 2017). Unsupervised learning favors the discovery of structure in unlabeled data, a feature that has obvious advantages for applications in improvised music.

In the case of compositional applications, however, the use of supervised learning could be equally interesting, by allowing the design of less generic and more idiosyncratic – i.e. *composition-specific* – listening strategies. An example of such an approach is the system described in this paper, which employs a supervised machine learning algorithm, particularly a feedforward Neural Network (NN) trained to recognize four different sound classes: single notes, multiphonics, air tones and slap tones. Background noise was added as a fifth class to the classification task in order to integrate noise gating in the recognition process. As a result, the listening algorithm requires no noise gating and therefore no threshold adjustment in run-time.

[1] A video recording of a performance of the piece is available at http://www.artemigioti.com/demos/Neurons.html

The training data for the NN was collected in four recording sessions with the help of two saxophonists. The recording sessions were conducted in two different rooms with microphones of different directionality (one super and one hyper-cardioid) and placement (clip-on and stand respectively). Each of the musicians used a different soprano saxophone. Collecting samples from more than one musicians/instruments and recording setups aimed at ensuring adequate variability in the training set and avoiding overfitting (the problem of a machine learning algorithm fitting the training set very well, but failing to generalize on previously unseen examples). For the same reason, initially synthetic data was generated by applying filters and artificial reverberation on some of the recorded examples. However, while the examples from the second saxophonist seemed to improve the performance of the algorithm, the synthetic data had the opposite effect and was abandoned later in the training process.

The recorded examples were edited manually to remove any ambiguities that could lead to data mislabeling (e.g. unstable multiphonics) and analyzed using a window size of 2048 samples and 50% hop size. The data set was partitioned into three separate sets: a training set consisting of 23889 examples (about 60% of the data set), a cross-validation and a test set (each about 20% of the data set). Each example consisted of a feature vector and a label between 1 and 5 (e.g. 1 for single notes, 2 for multiphonics etc.). The feature vector included 13 Mel Frequency Cepstral Coefficients (MFCCs) and a few additional features such as spectral flatness, onset, pitch variation (frequency ratio between the current and previous pitch value) and frequency beats. The latter is a binary-valued feature used to signal amplitude periodicities that are indicative of interference among frequency components of a multiphonic.

The NN consisted of the same number of input and hidden units that used the logistic sigmoid as an activation function, and was trained using backpropagation. During the training process several feature sets were tested and evaluated both on a separate test set and live with the collaboration of the saxophonist. These run-time tests were crucial to the development process, since they helped identify the weaknesses of the algorithm and provided valuable feedback for the ongoing training process. For example, whenever the network would consistently fail to identify certain examples, an additional number of similar examples would be recorded, or new features would be added to the feature vector, in order to help correctly identify the misclassified examples (Fig. 1).

One of the main challenges of the training process was finding a workaround for polyphonic pitch detection. Several polyphonic pitch detection algorithms were tested and rejected due to their poor performance. Instead, fluctuations in the detected pitch values (resulting from the pres-

ence of more that one pitches) and beat frequencies were used to facilitate the recognition of multiphonics. However, the detection of beat frequencies made use of a larger FFT window size, resulting in a delay in the detection of multiphonics.

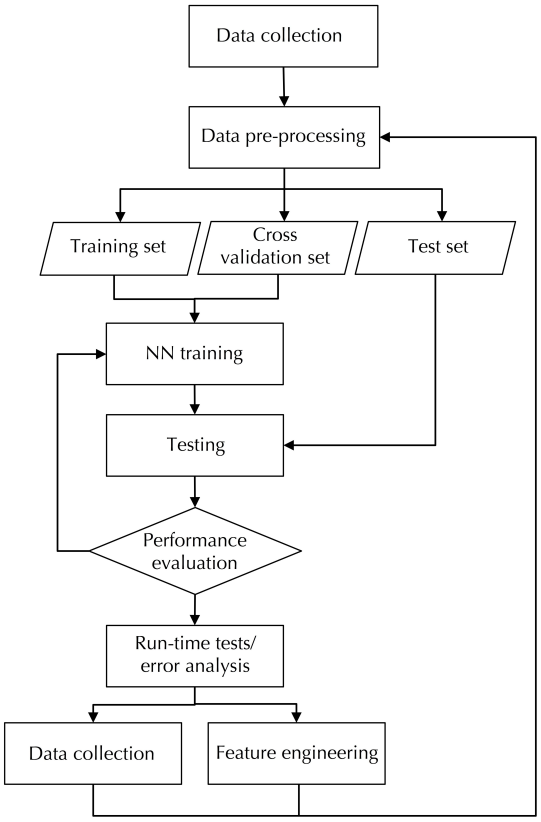

Figure 1. NN: training and testing

When the training process was completed, the accuracy of the network on the test set reached 91%.[2] In order to further improve the performance of the algorithm in run-time, two common machine learning strategies were explored: averaging the predictions of the network over a certain time span (e.g. averaging every 2-5 predictions) and filtering the output of the network based on its confidence (outputting only predictions with a probability higher than a certain threshold). The first method made the system less flexible by increasing its response time, a weakness particularly noticeable in denser musical textures. The second method improved the performance of the network significantly, by filtering out some false predictions and increasing its overall accuracy. An additional gain from the use of the confidence filter was the integration of sound source separation in the recognition process. Concretely, the NN seemed to only output predictions for the four classes it was trained to classify, "ignoring" any other sounds (i.e. the electronics).

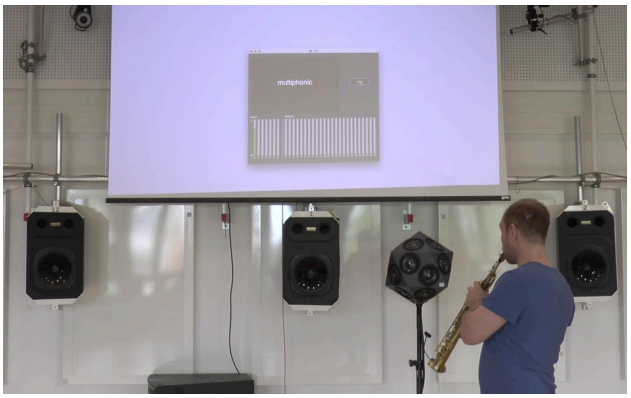

Figure 2. Saxophonist Joel Diegert testing the NN in run-time.

Interaction Scenarios

The classification algorithm described in the previous section was embedded in various interaction scenarios, entailing different compositional instructions, agent behaviors and, by extension, sonic interaction affordances. Two of Truax's (1984) levels of aural attention (*listening-in-readiness* and *listening-in-search*) are referenced as metaphors for some of the listening modes involved in these scenarios.

Scenario 1: Listening-in-readiness, listening-in-context

In this scenario, the occurrence of each of the four sound classes causes a different response (*listening-in-readiness*). Single notes and slap tones trigger textures of synthesized sounds, air tones are processed by a signal processing chain and multiphonics are resynthesized using a "spectral freeze" effect.

In parallel to this instant recognition process, a measure of texture variability is calculated every second, providing information on the variability/uniformity of the sound material played by the saxophonist over the last ten seconds. The value of estimated texture variability is used to control the amplitude of low frequency components of the electronics that become louder as texture variability increases. When texture variability reaches a certain threshold value, the system temporarily switches off its input and enters a non-listening state.

However, this should happen no sooner than when indicated in the score, meaning that the musician has to make sure that the system does not enter its non-listening state too soon. When the low frequency components of the electronics become considerably louder, the performer has to

[2] A video demonstration of the machine listening algorithm is available at http://www.artemigioti.com/demos/soprano_sax_sound_event_recognition.html

intervene by taking some control action (i.e. playing less variable sound material). Air tones are ignored by the variability measure and can also be used as a regulatory measure, in order to prevent the system from entering the non-listening state.

In this scenario, the value of estimated texture variability is key to the interaction between the musician and the software agent, partly due to the "error" factor. Concretely, the estimated texture variability can increase both due to recognition errors and due to "human error" (e.g. an unsuccessful execution of an unstable multiphonic, in which the second pitch is difficult to obtain). This part of the score contains a large number of such unstable multiphonics, along with other material, organized in short fragments the order of which is left to the performer (Fig. 3).

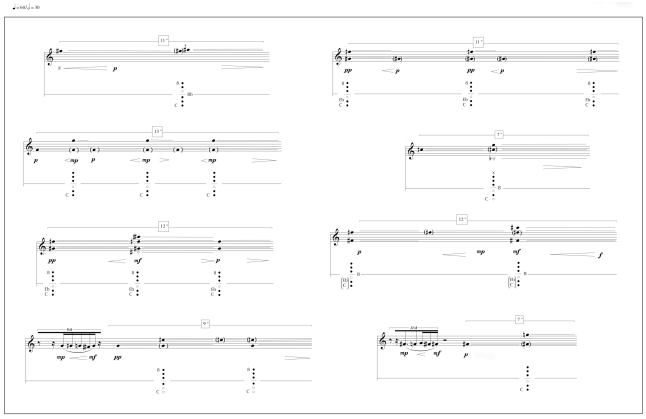

Figure 3. Neurons: score excerpt.

Scenario 2: Non-listening state

In this scenario, the sound output of the computer is controlled exclusively by algorithmic processes, involving sound synthesis and feedback. During this part of the piece, the musician stops playing and waits for an auditory cue signaling that the agent is listening again.

Scenario 3: Listening-in-search, listening-at-will

In scenario 3, the input of the machine listening algorithm is switched on and off in search of multiphonics. The agent randomly "chooses" when to switch its input on (*listening-at-will*) and provides the musician with an auditory cue when doing so. The term "at-will" in this context is suggestive of the changed dynamics of the interaction between the musician and the software agent and not "free will": in contrast to scenario 1, in this scenario the software agent becomes pro-active, limiting the musician to a merely reactive role (Fig. 4).

When the software agent is "listening", the musician has to choose from a number of multiphonics in the score and play that multiphonic for several seconds. If the execution is evaluated as stable by the software agent, the algorithmic synthesis processes initialized in scenario 2 are temporarily interrupted by a "spectral freeze" effect. Otherwise, the musician has to choose a different multiphonic and repeat the effort. Sound events other than multiphonics (e.g. single notes, air tones etc.) are ignored (*listening-in-search*).

Scenario 4: Listening-in-search

In scenario 4, the agent responds selectively to air tones. In addition to signal processing, the detection of air tones in this mode triggers the playback of resynthesized spectra of multiphonics played by the saxophonist earlier in the piece. In this part of the piece, the performer can choose from a number of actions in the score, which can be executed in any order, one or more times.

	Scenario 1	**Scenario 2**	**Scenario 3**	**Scenario 4**
Performer	Pro-active	Inactive	Reactive	Pro-active
Software agent	Reactive	Pro-active	Pro-active	Reactive

Figure 4. Types of decision-making in the 4 interaction scenarios of Neurons.

In the interaction scenarios described above, machine listening and decision-making extend beyond a generic one-to-one input-output mapping. The *selective listening* and *non-listening* modes constitute behavioral elements of the agent, which exhibits varying degrees of autonomy and responsiveness (Fig. 4). The concept of "error" and its double meaning with regard to the human-machine divide (human vs computational error) is also explored for its potential as an interaction component. For example, in scenario 3 the software agent evaluates the "stability" of the execution of multiphonics and responds only when a multiphonic is stable, while in scenario 1 recognition errors can cause the software agent to enter a non-listening state, forcing the musician to take preventative actions.

With the exception of "listening-in-readiness", all other listening modes involved in these four interaction scenarios are exemplary of idiosyncratic agent behavior, specifically designed for these scenarios. "Listening-in-search" is a particularly good example of such a listening mode/attentional strategy. In scenario 3, the software agent "listens-in-search" of multiphonics and ignores any other sound events. Similarly, in scenario 1 the musician "listens for" increases in the amplitude of low frequency components of the electronics, requiring him/her to take certain control actions. These sounds carry extrinsic information related to the compositional idea and the rules of the interaction between the two agents and are anticipated by the performer – hence "listens for" – as part of that interaction scenario.

Discussion

This paper presented an interactive machine-listening based system using supervised learning to perform real-time recognition of specific playing techniques. The use of machine learning and the integration of a classification task in the auditory processing stage of the system aimed at shifting the focus of machine listening from *analysis* to *interpretation* and from *sensory* to *symbolic* (McAdams and Bigand 1993) information. The term *sensory information* is used to denote signal-level features, extracted in the analysis stage of the system, while *symbolic information* refers to higher-level representations of the human input (in this case, the four classes recognized by the NN), obtained by interpreting analysis data. The objective of this approach was to integrate musical terminology in the listening task; particularly, a terminology that goes beyond MIDI-level information (pitch, duration, loudness).

The machine listening algorithm described in this paper was developed specifically for the composition *Neurons*. Its applicability to other compositions and/or improvised performances – though theoretically possible – lies beyond the scope of the work described here. The purpose of this work was not to create a general-use performance/improvisation system, but to explore the application of supervised learning in specific interaction scenarios within an interactive composition.

While the use of unsupervised learning may be preferable for human-computer improvisation systems, due to the higher degree of unpredictability involved in improvisation, in the case of compositional applications idiosyncratic agent behavior is often a desideratum. In interactive compositions, listening strategies are highly dependent on the instrumentation and compositional idea and are usually designed for specific interaction scenarios, as opposed to the more generic listening modes of improvisation systems. A supervised learning algorithm, such as the algorithm described in this paper, enables the design of a listening behavior that is specific to a certain composition (or several interaction scenarios within it), allowing for a greater degree of freedom in designing sonic interactions. Further, technical advantages of the use of supervised learning include the possibility to integrate sound source separation and noise gating in the training process, thus avoiding manual configurations before or during the performance.

Acknowledgements

The author would like to thank Marko Ciciliani and Gerhard Eckel for their valuable comments and suggestions for this manuscript.

References

Chadabe, J. 1984. Interactive Composing: An Overview. Computer Music Journal 8(1):22–27.

Collins, N. 2008. Reinforcement Learning for Live Musical Agents. Proceedings of the 2008 International Computer Music Conference (ICMC 2008). International Computer Music Association.

Collins, N. 2011. LL: Listening and Learning in an Interactive Improvisation System. Research Report. (https://composerprogrammer.com/research/ll.pdf, accessed 01.03.2017).

Drummond, J. 2009. Understanding Interactive Systems. Organised Sound 14(2):124–133.

Dudas, R. 2010. "Comprovisation": The Various Facets of Composed Improvisation within Interactive Performance Systems. Leonardo Music Journal 20:29–31.

Gioti, A.M. 2017. Machine Listening in Interactive Music Systems: Current State and Future Directions. Proceedings of the 2017 International Computer Music Conference (ICMC 2017), 216–220. International Computer Music Association.

Lévy, B.; Bloch G. and Assayag, G. 2012. OMaxist Dialectics: Capturing, Visualizing and Expanding Improvisations. Proceedings of the 2012 International Conference on New Interfaces for Musical Expression (NIME 2012), 137–140.

McAdams S. and Bigand, E. 1993. Introduction to Auditory Cognition. In: McAdams S. and Bigand, E. (Eds.) Thinking in Sound: The Cognitive Psychology of Human Audition. Clarendon Press.

Nika, J.; Chemillier M. and Assayag, G. 2017. ImproteK: Introducing Scenarios into Human-Computer Music Improvisation. ACM Computers in Entertainment.

Paine, G. 2002. Interactivity, where to from here? Organised Sound 7(3):295–304.

Rowe, R. 1993. Interactive Music Systems: Machine Listening and Composing. MIT Press.

Smith, B. D. and Deal, W. S. 2014. Ml.* Machine Learning Library as a Musical Partner in the Computer-acoustic Composition Flight. Proceedings of the 2014 International Computer Music Conference (ICMC 2014),1285–1289. International Computer Music Association.

Tatar, K., and Pasquier, P. 2017. MASOM: A Musical Agent Architecture based on Self Organizing Maps, Affective Computing, and Variable Markov Models. Proceedings of the 5th International Workshop on Musical Metacreation (MUME 2017). AAAI Press.

Thom, B. 2000. BoB: an Interactive Improvisational Music Companion. Proceedings of the Fourth International Conference on Autonomous Agents, 309–316.

Truax, B. 1984. Acoustic Communication. Ablex Publishing Corporation.

Van Nort, D.; Braasch, J. and Oliveros, P. 2009. A System for Musical Improvisation Combining Sonic Gesture Recognition and Genetic Algorithms. Proceedings of the 2009 Sound and Music Computing Conference (SMC 2009), 131–136.

Yee-King, M. J. 2011. An Autonomous Timbre Matching Improviser. Proceedings of the 2011 International Computer Music Conference (ICMC 2011), 122–125. International Computer Music Association.

Young, M. 2007. NN Music: Improvising with a 'Living' Computer. Proceedings of the 2007 International Computer Music Conference (ICMC 2007), 508–511. International Computer Music Association.