Technical feedback and resources

zoennchen.benedikt@hm.edu teo.sanchez@hm.edu

Group A

Provisional title : Human-machine musical co-improvisation
Group members:

Idea 1 (Using an existing co-improvisation software system)

The french research institute for cooperation between acoustic and music (IRCAM) has studied co-improvisation since 2009. Gérard Assayag, Marc Chemilier, Jérôme Nika and colleagues developed co-improvisation softwares. It was first named Omax (standalone software) and later Dicy2 (package). Dicy2 is a collection of improvising agents you can use within Max/MSP, a graphical programming software for music and sound. Unfortunately, MaxMSP is a proprietary and paying software even if Dicy2 is free. You can benefit from a free trial period.
Video tutorials of Dicy2 are available.
Omax and Dicy2 support both MIDI signals and audio signals.
An event dedicated to human-machine co-improvisation named Improtech is organized every year in France. It might be a good entry point to the community of co-improvisation.

An free alternative was developed by Adam James Wilson from the New York City College of Technology. It is a factor oracle you can use in Pure Data.

What we recommend:

- Look at the video of musicians improvising with factor oracles.
- Install a factor oracle using Pure Data
- Learn about how such improvising agents works under the hood. From what I understand, they are not based on neural network but automata, in which each note is a state, and the model updates transition probabilities between notes when the musician is improvising. You can learn more about it in one of the following papers (by order of importance):
    - *Assayag, G., & Dubnov, S. (2004). Using factor oracles for machine improvisation. Soft Computing, 8(9), 604-610.*
    - *Wilson, A. J. (2016). factorOracle: an Extensible Max External for Investigating Applications of the Factor Oracle Automaton in Real-Time Music Improvisation.*
    - *Nika, J., Chemillier, M., & Assayag, G. (2017). Improtek: introducing scenarios into human-computer music improvisation. Computers in Entertainment (CIE), 14(2), 1-27.*
    - *Dubnov, S., Assayag, G., & Cont, A. (2007). Audio oracle: A new algorithm for fast learning of audio structures. In Proceedings of International Computer Music Conference (ICMC). ICMA.*

- *Assayag, G., & Bloch, G. (2007, August). Navigating the oracle: A heuristic approach. In International Computer Music Conference'07 (pp. 405-412).*
- *Rueda, C., Assayag, G., & Dubnov, S. (2006, August). A concurrent constraints factor oracle model for music improvisation. In XXXII Conferencia Latinoamericana de Informática CLEI 2006 (pp. 1-1).*
- *Déguernel, K., Vincent, E., & Assayag, G. (2018). Probabilistic factor oracles for multidimensional machine improvisation. Computer Music Journal, 42(02), 52-66.*
- *Keller, R. M., & Morrison, D. R. (2007, July). A grammatical approach to automatic improvisation. In proceedings of the sound and Music computing conference (pp. 330-337).*

A good **technical** contribution would be to try to implement more elaborated version of a factor oracle in a free and open-source software such as Pure Data.
Pure data can also run on micro-computers such as Raspberry Pi, simplifying a later implementation as an hardware stompbox/effect pedal (if enough time).
A good **musical** contribution would be to compose a piece using an improvisational agent, comment, and document this piece (video recording for instance).

Keep in mind that the hard problem is co-improvisation and adding a midi-to-sound plug-in later on might be much easier. Just look for “audio-to-midi real time free” in google and you might already have some choices to start with. Ideally, such audio-to-midi in Pure Data (for example) might allow you to be more flexible for the final implementation of your performance/tool.

Idea 2 (Basic machine learning and signal processing)

Instead of complex generative machine learning algorithms, a simple approach can be employed. This involves non-generative techniques, meaning the sound isn't directly generated by the machine learning algorithm–a task that usually requires advanced software solutions.

1. **Record and Label Sounds:**
    a. Record a database of sound snippets by playing your instrument.
    b. Label each sound based on your aesthetic preference.
    c. Optionally, if you have access to other recorded sounds, you can include them too.
2. **Sound Analysis:**
    a. Divide your recording into smaller segments.
    b. Use signal processing techniques such as generating a spectrogram or mel frequency cepstral coefficients (MFCC) for each sound snippet.
    The Python library librosa can do the trick.

A spectrogram shows the amplitude of sound frequencies over time. MFCC provides a condensed representation, capturing essential timbral characteristics, making it suitable for comparing different timbres.

Understanding Limitations: Note that some information is inevitably lost in this process. Analyzing sound in small segments (using a sliding window) means the frequency range captured depends on the window size. A larger window size decreases temporal accuracy, while a smaller window size reduces frequency accuracy.

3. **Create and Train a Classifier:**
    a. Convert sound analyses (spectrograms or MFCCs) into image-like formats.
    b. Categorize and label these images.
    c. Train a neural network to classify these images, predicting your assigned labels.
4. **Utilizing Classifier Predictions:**
    a. Use the classifier's predictions to select sounds. For example, if the classifier 'likes' a played sound, it could playback a similar sound from the database, determined by comparing MFCCs.
    b. If the classifier 'dislikes' a sound, it could remain silent or select a contrasting sound from the database.

Instead of a classifier you could also predict the next sound snippets in a sequence of snippets.

3b. Generating Sound Sequences:
1. Experiment with creating sequences of sounds that you think are pleasing.
2. Label these sounds with numbers (each unique sound has a specific id) and train a machine learning model on these numbered sequences.
3. Use the trained model to predict the next number in a given sequence.
4. Convert these numbers back into sounds either by re-synthesizing them or playing pre-recorded snippets.

For this prediction, you could use a simple **neural network** or more advanced **recurrent neural networks (RNN/LSTM)**.

How to use the prediction can be much more creative. You may come up with different ideas! If you wanna learn more about signal processing in combination with machine learning Valerio Velardo has a lot of explanatory material on his YouTube-channel. Also, Artemi-Maria Gioti was also interested in machine-artist communication and gave a talk in the last Wintersemester for AICA and here is also an interview where she talks about some implementation details.

Group B

Provisional title : Collaborative visual storytelling
Group members:

You can run Stable Diffusion locally on your machine and there is a WebGUI to play with it: https://github.com/AUTOMATIC1111/stable-diffusion-webui
AUTOMATIC111 also provides an API so it should be possible to

1. Make a history of prompts via some GUI, for example a Python application
2. Send parts of the history as prompt to Stable Diffusion
3. Receive and display the image and save the pair (image, prompt) into the archive/history

You can also access Stable Diffusion and other image generators via an API but if it does not run on your machine you have to pay for it.

Group C

Provisional title : Emotion recognition for personalized recommendations
Group members:

The following libraries might help you to recognize emotions but keep in mind that it is a rather controversial topic so maybe you can think about a creative way to face this issue:

1. DeepFace Library
2. FER Library
3. there are probably more

Group D

Provisional title : Virtual archives of clothe artefacts
Group members:

Group E

Provisional title : AI-augmented audioguide
Group members:

Idea 1 (Realization of a prototype)

What you probably need is a large language model such as LLama (it is publicly available) combined with special information using something like LLamaIndex or/and LangChen. The following blog post explains how one can combine LLama and LangChen and run it on a local computer. Of course another possibility is to proprietary LLMs like GPT models that run on the cloud and combine them with LangChen but this will cost money.

Idea 2 (Realization of a concept)

You could try to realize a demo which “fakes” the machine learning part by using a scripted solution. To generate speech you could still use machine learning i.e. a text to speech model (such as gTTs or some high level application) and maybe a speech to text mode if needed. If you need personal voices with different ancients you could use celebrity voices but this is highly controversial and should be critically discussed in your work.