Remixing AIs: mind swaps, hybrainity, and splicing musical models

Nick Collins[1]
Vít Růžička[2]
Mick Grierson[2]
[1] Durham University, nick.collins@durham.ac.uk
[2] Creative Computing Institute, University of the Arts London
previtus@gmail.com, m.grierson@arts.ac.uk

Abstract. Musical models created using machine learning techniques can be hacked, repurposed and spliced in many new ways. The current generation of models is not at any level of simulated consciousness sufficient to trigger immediate ethical blocks, and stands open to systems surgery. Beyond morphing of inputs and outputs, hidden layers of deep learning models may themselves be actively stimulated and substituted, from unit weights and biases to activation functions. Layers from multiple networks may be swapped and interpolated, from single units to complete layers. The hybridity of musical formation takes place at the level of model internals, in artistic transformations beyond standard transfer learning. This activity of AI code bending is dubbed here 'hybrainity', and alongside theoretical discussion of its origins, potential and ethics, examples of hacking particular machine learning models for new creative projects are provided, including applications in live performance and audiovisual generation.

Keywords: Hybrid models; music AI; audiovisuals; machine learning; deep neural network; autoencoder; generative adversarial network; transfer learning.

1 Introduction

Machine models are highly open to creative manipulation, as music AI provides new angles on the musical mind. This paper owes a debt to the artistic approach to machine listening algorithms conducted by Bowers and Green (2018); here machine learning algorithms are creatively exploited to similar artistic ends. The act of creative tampering within software, of 'code-bending' is relevant (Bergstrom and Lotto 2015), as is the practice of embracing glitches (Cascone 2000).

Central to this research is the idea that the notion of a brAIn transplant, of mind swapping or moving a brain to a new host, can provide artistic stimulation. It is a recurring literary device in popular culture, as illustrated by Edgar Rice Burroughs' *The Master Mind of Mars* (1927), Robert Sheckley's *MINDSWAP* (1966), The Star Trek episode *Spock's Brain* (1968), *The Man with Two Brains* (1983), *Big* (1988), *Jumanji: Welcome to the Jungle* (2017), and many more. There are earlier precedents in Hinduism (Kumar and Mahapatra 2009), great enthusiasm in the transhumanist movement (O'Connell 2017), and philosophical consideration (Snowdon 1991), even if such procedures in wetware remain far from science fact; the only successful monkey head swap led to the host's death in nine days from immune system rejection, and human nerve tissue is very prone to damage. The level of a cultural exchange is more amenable, memetic transfer not requiring immunosuppressant drugs.

There has been an explosion of interest in machine learning for computer music in recent years, particularly with research into deep learning over large training corpora (Briot et al. 2017; Choi et al. 2017). New models, such as autoencoders, Generative Adversarial Networks (GANs), and Long Short Term Memory (LSTM) Recursive Neural Net (RNN) sequence models, are ripe for exploitation. All have been used in both the audio and symbolic music representation domains.

In terms of the swapping of musical information between models, we might point to many existing approaches to cross-synthesis and style transfer, exemplified by signal processing devices such as vocoding and interpolation between models in spectral modelling synthesis, or by symbolic domain algorithmic composition projects such as David Cope's work (Cope 2001). Though Cope later destroyed his own database, hindering reproducibility for researchers, he observed that in recreating the style of particular composers he often salted a corpus intended to represent a particular composer with further works from their historical contemporaries, and he explored hybrid models derived from mixing corpora. Transfer learning is an explicit method of reusing previously trained models, particularly the earlier layers responsible for feature detection in an end-to-end system, ahead of substituting one or more of the final layers or an additional further network, to be trained on a novel problem (Choi et al. 2017b). It doesn't normally proceed with artistic intent, including on intermediate layers.

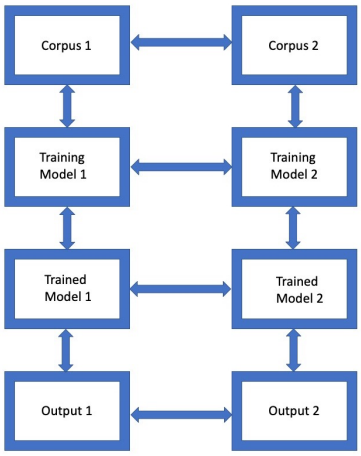

Fig. 1. Points of intervention between two (or more) models, from inputs (source corpus) through the training stage, then on to already trained models, and their outputs

With respect to splicing and hacking machine learning models, Figure 1 illustrates many points of intervention. Coupling of two or more models can occur at the stage of inputs, of outputs, or upon internal layers. Continuous interpolation and morphing is possible, as well as discrete substitutions and swaps. The diagram could continue to the right and place more cross-connections to introduce further models beyond two; model training has been simplified for the clarity of the diagram eliding separate training, validation and test sets. Training tends to be a slow process, but running of a trained model can be carried out live, and interactive interventions will tend to be in the latter situation (though there are online trained machine learning algorithms that develop on the fly). The sorts of model interactions possible range from gentler interventions mixing up corpuses and cross-fading outputs, to joint training with interference from one model to the other, and Frankensteinian hybrid models (Todd and Werner 1999).

Action between models can place any model as primary and another receiver, or blend halfway, illustrated by the bidirectional horizontal arrows. The vertical arrows are also all bidirectional rather than simply top-to-bottom to encompass cases of feedback, such as audio outputs joining a new corpus for training, feeding into learning algorithm parameters, etc.

Two case studies now follow in the manipulation of learning models. Section 2 explores spectral domain autoencoders as audio effects units, hacked through live interpolation of models at the level of internal layers, both pre (weights and biases) and post (on outputs) calculation, as well as live synthesised stimulation of the units at any layer. Section 3 investigates an audiovisual model built by passing information between an audio domain LSTM sequence model, and visual domain GAN variant, generating creative outputs through navigation of the latent space and through dynamic reconnection of convolutional layers. The creative and ethical implications of such work are then further discussed.

2 Activating Allotransplanting Autoencoders

Autoencoding neural networks have shown great promise as new synthesisers and effects units acting directly on audio signals, operating in the time domain or the spectral domain (Colonel et al. 2017, Engel et al. 2017, Roche et al. 2018, Sarroff and Casey 2014). Both shallow and deep encoder/decoder networks trained over spectrograms derived from corpora of audio files are considered here. In some cases, the Open-Unmix deep learning source separation algorithm was first run to isolate a stream (Stöter 2019), for example, the vocoded vocals from New Order's *Ecstasy* (1983).

Initial Web Audio API web browser experiments used convnet.js (https://cs.stanford.edu/people/karpathy/convnetjs/), though a github project was subsequently set up demonstrating live manipulation of deep networks both in javascript, and for a plugin (written in C++) for the SuperCollider audio programming language (https://github.com/sicklincoln/Keras-to-Realtime-Audio). Networks are first trained via the keras python library with librosa (https://librosa.github.io/librosa/) used for audio file analysis, allowing much faster training over larger corpora than pure javascript (this can give a difference of minutes compared to hours!). Trained models were exported via ONNX (https://onnx.ai/) for javascript, or with Kerasify (https://github.com/moof2k/kerasify) for C++. Subsequent to loading a model, the C++ plugin utilised a custom implementation of a deep neural network for its internal data structure, to make it straight forward to explore the interpolation of layers, signal injection, activation function substitution and other manipulations; kerasify's own code was too slow for effective realtime work. The ONNX route was able to cope with more types of network model, but was trickier to modify on the fly; convnet.js was more flexible for experimentation than ONNX for javascript. Web Audio API work utilised the MMLL library (https://github.com/sicklincoln/MMLL) for audio file analysis, both before convnet.js training, and for live deployment of trained models, including realtime control from analysis of microphone input and GUI widgets (Collins and Knotts 2019). The web browser code has the advantage of facilitating the creation of interactive systems easily shared with musicians via a single weblink, rather than requiring a SuperCollider plugin build for a particular operating system.

Finding the right network architecture and hyperparameters required many experiments, with some hints from the spectral autoencoder papers mentioned above. Working at 44.1kHz sampling rate, all layers were fully connected to their

predecessors, and inputs were log magnitude spectra (0.5*log(power+1)*scalefactor), normalised within 0 to 1 by a scale factor obtained from the audio corpus (after Engel et al. 2017 rather than the magnitude spectra of Colonel et al. 2017; power spectra also performed poorly); scalefactor=0.18326653390262 was derived from the maximum value seen across many audio files.

Two successful lower latency architectures used with convnet.js were:

- A shallow network based on a 1024 point FFT (512 hopsize, 512 magnitudes with Nyquist bin zeroed), with three layers: 512 inputs, a 128 sigmoid activation function hidden layer, and 512 outputs (paraphrasing Sarroff and Casey 2014 including quarter size hidden layer)
- A deep network in hourglass form based on a 2048 point FFT (1024 hop size, 1024 magnitudes with Nyquist bin zeroed): 1024 inputs, 512 fully connected ReLU activation layer, 256 neuron sigmoid activation layer, 512 ReLU layer, 1024 output (following Colonel et al. 2017)

256 unit input and output layers led to very poor results, with too many horizontally continuous low pitches howling in the re-synthesis. 4096 point FFT/2048 unit input and output layers were more reliable in training, but traded off latency and greater runtime cost; this FFT size became standard for the keras experiments, and had the helpful side effect of reducing the number of network prediction calculations per second, avoiding the situation where network calculation takes longer than realtime CPU availability. No great advantage accrued artistically from experiments with larger input sizes based on the inclusion of past spectral frames, only slower training and with more parameters increasing model size and realtime calculation time. Training with the adadelta Trainer in convnet.js was more reliable in convergence (l2_decay: 0.001, batch_size: 100) than stochastic gradient descent with momentum. In keras, the adam optimiser with mean-squared error (MSE) loss was the standard choice, typically with batch size 50.

With convnet.js, for 100 epochs, training of the first architecture might take around 3 hours, and the second architecture 5 hours. Some experiments used fewer epochs of training, since under-fitting could be creatively rewarding. The typical corpus for training was a three to four minute pop song. For example, with Radiohead's *Idioteque* this gave for the first model above 26628 spectral frame training instances, and after ten epochs of training, an L2 norm error of 0.22155, down from initial epoch 2.88624.

In keras, training was orders of magnitude faster, often a matter of minutes rather than hours, enabling training over many more epochs, and many more audio files' worth of source spectra.

Once networks were trained, they were activated in unconventional ways:

- Running the network on highly unrelated unseen input
- Morphing two models live at the level of swapping particular layers
- Activating a hidden layer of a network via direct sound synthesis, or injection of another audio signal

Direct sound synthesis or sound injection was used to set hidden layer activations in contrast to Sarroff and Casey's (2014) exploration of MIDI controllers. Interesting effects in latent activation were achieved by using noise sources, by a Karplus-Strong algorithm applied to an initial noise buffer, and by one co-prime frequency oscillator per hidden unit. A second audio signal was also injected directly into hidden units, rather than at the input.

For live re-synthesis, input phase was stored and re-used. These live networks could be run at around 20-30% CPU on a core (as opposed to 100% CPU training), and were low latency based on only the delay to fill an FFT buffer, transform and run through a network, IFFT, and enter into overlap add resynthesis. Thus for 512 sample hopsize, delay was on the order of 11-23 milliseconds (512 until next FFT triggered, 512 fade up with next overlap add, 1024 samples of input affecting over two output windows).

A further precedent to this work is the exploration of Echo State networks for sound synthesis, exploiting the inherent feedback loops within a reservoir of neurons (Holzmann 2009, Kiefer 2019). For our networks, output could be fed back to the input, or even fed back (suitably subsampled or downsampled as necessary) to an intermediate layer, or cross coupled at various stages between two or more trained networks.

A particularly noise music worthy interaction was achieved by turning up the output gain and working with live voice control; the frequency content of the voice (sibilant vs low growl for instance) led to related spectral responses found in the training sound file, creating an interesting hybrid modulation.

Network layers could be interpolated between two models. The SuperCollider UGen PV_DNNMorph was built to make this available as a live facility after loading two models. A user could choose which layer to interpolate and the interpolation parameter, or whether to interpolate the weights and biases ("pre") or the output after activation function ("post"), all adjustable on the fly to determine the network to be used at that moment. Interpolation of weights and biases between layers typically leads to worse performance than either pre-trained net on given data, but is musically interesting in introducing non-linear distortion to the reconstruction. Figure 2 demonstrates the effect on reconstruction scores (MSE) on moving from model 1 to model 2, tested via model 1 data. Figure 3 does the same with model 2 to model 1 over model 2 data.

A further SuperCollider UGen, PV_KerasifyActivationFromBuffer, enables direct stimulation of the units in a particular layer, with the choice of layer itself dynamically controllable; the buffer of activations simply has to be big enough to cover the maximum number of units in any layer. This leads to some rich sounds, often of a 'spectral synthesis' flavour given the phaseless log magnitude spectrum creation, but modulated via the net's training on particular sound file(s); direct stimulation at later layers led to greater higher frequency content activation, whereas earlier layers tended to keep more constrained in spectral envelope (unsurprisingly, given the greater influence of the training data over weights used in subsequent calculation). Audio examples and code are at https://composerprogrammer.com/remixingAIs.html

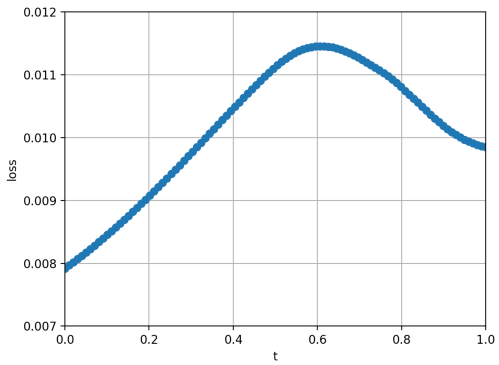

Fig. 2. Morphing in interpolation parameter t from model 1 (left) to model 2 (right), plotting mean squared error (MSE) loss on model 1's training data

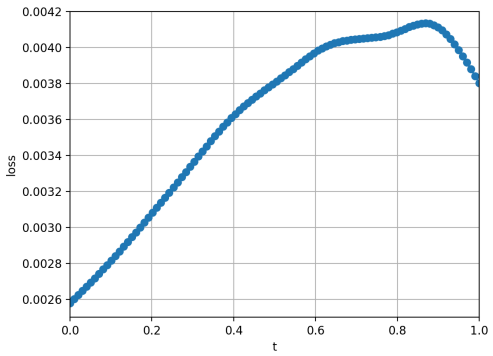

Fig. 3. Morphing in interpolation parameter t from model 2 (left) to model 1 (right), plotting mean squared error (MSE) loss on model 2's training data

3 Audiovisual Network Bending

In a further example of 'Hybrainity' that exploits the multimodal potential of contemporary neural network architectures, we demonstrate how audio and visual models can be interactively connected to create audiovisual hierarchical systems. These systems, although trained separately on a variety of sound and image corpora, can be configured to generate interesting and novel predictions based on input from each other, even in real-time, for the purposes of exploring potential creative outcomes for composers and visual artists. In these cases, different audio and visual representations from separate, pre-trained artificial neural networks are used as a means to modify and control the generation of new sound and image outputs.

As a potential use case, we have extended our work for the real-time control of generative audio sample synthesis (Grierson et al. 2019) using LSTMs, and combined it with a new real-time image synthesiser based on the Progressive GAN model (Karras et al. 2017). The model structure for the audio generator follows the same approach in *Mezzanine Vs. MAGNet* (https://github.com/Louismac/MAGNet), which creates novel sequences of audio magnitude spectra which are then re-synthesised as PCM data by phase reconstruction (Griffin and Lim 1984, LeRoux et al. 2010). Python versions of this software are available on the web (https://github.com/ual-cci/music_gen_interaction_RTML), where a complete series of audio examples produced by the system can be found, including examples of real-time use. The image generation uses a novel technique called 'Convolutional Layer Re-Connection'; a real-time demonstration is at https://www.youtube.com/watch?v=w24XcLon1Ns. This software allows progressive GAN models to be interactively controlled through interpolation and Network Bending approaches, where weights are interactively manipulated to control visual outcomes.

A diagram demonstrating the technique in a Creative AI pipeline is shown in Figure 4. We use a modified neural network model, the 'Re-Connected' GAN, in place of a standard progressive GAN without issues. In the diagram below, the LSTM model produces spectral frames interactively, and these are visualised by splicing the generated spectral frames as part of the Re-Connection approach. In this way, audio spectral frames are used as an auto-encoded feature representation with a Re-Connected Progressive GAN model. Using the large strength of the reconnection operation, we

can generate a range of potential visual outcomes that are tightly coupled to the spectral representations, both temporally and also across the complex space of the multidimensional vectors within the Re-Connected model. Importantly, the outcomes are potentially quite flexible to the user; the outputs can be both highly abstract and also maintain significant shapes and details from the original generative model. This is among the first hierarchical multimodal model of its kind, and operates in real-time, allowing for entirely new control methods for generative neural network models. https://github.com/ual-cci/GAN_explorer#osc-listener--audio-visual-application-demo

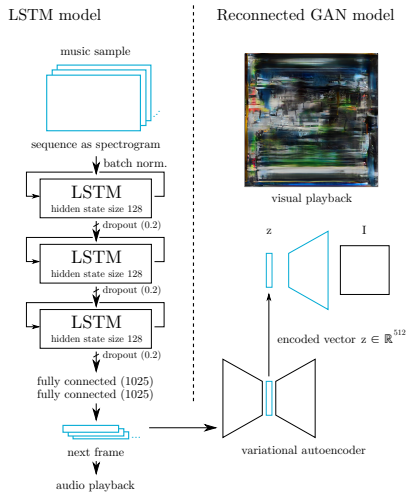

Fig. 4. Diagram of the multi-modal Audiovisual Network Bending architecture using LSTM, VAE and GAN approaches for real-time, interactive control of neural network inference.

4 Discussion

AI is not just machine learning; but learning is so critical to human music making, that no recent respectable AI lacks some sort of education. More radical approaches to subvert machine learning algorithms might deliberately miseducate otherwise worthy models (‘corrupting the young’). Human education can be more haphazard than laboratory-controlled machine learning, with a messier picking up of influences. A punk aesthetic might even look to minimise training. The artist Terence Broad has already taken this to the natural extreme with generative adversarial network models, training them on no external data and letting their internal competitive dynamics lead to interesting visual outputs (Broad and Grierson 2019).

Further possibilities for interventions in machine learning systems might include:

- Voting irregularities in ensemble learning
- k-Means or other unsupervised clustering algorithms updating cluster representatives based on dissimilarity rather proximity
- Unrelated input and output training; setting up an arbitrary association, or deliberately impossible learning tasks
- Growing or pruning neural nets over time, stretching out or shrinking layers (new nodes may have random connection weights and bias, or be seeded by some other input)

- Pointlessly over-complicated deep networks (‘too many layers Mr Mozart’)
- Multi adversarial learning (Durugkar et al. 2016), with a confused fight between many discriminators and generators, all in conflict analogous to a richer human debate

Whilst machine learning purists concerned with maximising performance on engineering tasks may baulk at this license, fearing that such exploration is more likely to obfuscate understanding of learning algorithms’ workings, these suggestions echo the artistically stimulating machine listening corruptions of Bowers and Green (2018).

The notion of remixing an AI has headier ethical implications the more developed our AI technology becomes (Collins 2011, Holzapfel et al. 2018, Sturm et al. 2019). Changing an AI’s mind, beyond gentle debate between Turing-test-passing conversationalists, heads into the moral and psychological issues of brain washing (Taylor 2017). It would be hard to argue for any consciousness and moral rights for current generation models, but AI researchers do have a responsibility to consider such issues ahead of time (Tegmark 2018). The AIs themselves continue to fail to embed socially as autonomous agents; it is the human researchers who set agendas and act to promote their own research interest. Initiatives such as safe interactive music AI playgrounds (Bown, Carey and Eigenfeldt 2015) may be medium-term havens for the sharing of musical BrAIns.

The mind-body problem in philosophy (McGinn 1989) has been echoed by Thor Magnusson with respect to the common situation where a new instrument has an interface (body) and virtual synthesis engine (mind) (Magnusson 2019). In raising the spectre of mind swapping, we have simultaneously raised body swapping; the possibility of moving mind somewhere else necessarily takes a position on the feasibility of this duality. There is a rich possibility in multiple mind swapping, illustrated by the combinatorial dilemma of the Futurama episode *The Prisoner of Benda* (2010) and an associated mathematical theorem on restoring multiply swapped individuals to their original body via intermediaries (Singh 2013). All of this may or may not recommend a sub-field of transplantation studies for AIs to the reader. Nonetheless, ideas of hybridisation are key to creative practice, creative action is founded upon taking something of other people’s work into ourselves (Cole 2020; Knobel and Lankshear 2008), and we are memetically helpless to avoid some transmogrification in our lives.

5 Conclusion

This paper took a more artistic approach to music AI founded in the notion of brAIn surgery, operating on machine learning algorithms via live manipulation of audio and visual domain neural networks. Notions of mind swapping and hybridising of models (‘hybrainity’) were productive prompts to consider new avenues in music, music's powerful hybridity offering a natural route to creative outcomes for AIs. Interventions could operate at many stages and with many feedback routes in the application of machine learning, from corpuses, through training, to manipulation of the eventual runtime models. There is great potential for machines which prompt new music and new audiovisuals, and for creative computing to modulate machine learning research.

Acknowledgments. This work was supported by the Arts and Humanities Research Council funded project Musically Intelligent Machines Interacting Creatively (MIMIC) (mimicproject.com). No AIs were harmed in the conduct of this research, within current parameters of machine consciousness.

References

Ames, C. (1989). The Markov process as a compositional model: A survey and tutorial. *Leonardo*, 22(2), 175-187.

Bergstrom, I. & Lotto, R. B. (2015). Code Bending: A new creative coding practice. *Leonardo*, 48(1), 25-31.

Bowers, J. & Green, O. (2018). All the Noises: Hijacking Listening Machines for Performative Research. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, 114–119.

Bown, O., Carey, B. & Eigenfeldt, A. (2015). Manifesto for a Musebot Ensemble: A platform for live interactive performance between multiple autonomous musical agents. In *Proceedings of the International Symposium of Electronic Art*.

Briot, J-P., Hadjeres, G. & Pachet, F. (2017). Deep learning techniques for music generation: A survey. arXiv preprint, arXiv:1709.01620

Broad, T. & Grierson, M. (2019). Searching for an (un)stable equilibrium: experiments in training generative models without data. *NeurIPS 2019 Workshop on Machine Learning for Creativity and Design*, https://arxiv.org/abs/1910.02409

K. Cascone, K. (2000). The aesthetics of failure: "Post-digital" tendencies in contemporary computer music. *Computer Music Journal*, 24(4), 12-18.

Choi, K., Fazekas, G., Cho, K. & Sandler, M. (2017). A tutorial on deep learning for music information retrieval. arXiv preprint, arXiv:1709.04396

Choi, K., Fazekas, G., Sandler, M. & Cho, K. (2017b). Transfer learning for music classification and regression tasks. *Proceedings of ISMIR*.

Cole, R. (2020). Musical Plagiarism: Why it Can be Admirable to Steal. Published online May 04, 2020 https://www.rosscole.co.uk/blog/tag/plagiarism

Collins, N. & Knotts, S. (2019). A Javascript Musical Machine Listening Library. *Proceedings of the International Computer Music Conference*, New York.

Collins, N. (2011) Trading Faures: Virtual Musicians and Machine Ethics. *Leonardo Music Journal* 21, 35-39.

Colonel, J., Curro, C. & Keene, S. (2017). Improving neural net auto encoders for music synthesis. In *Audio Engineering Society Convention* 143, Audio Engineering Society, October 2017.

Cope, D. (Ed.) (2001). *Virtual music: Computer synthesis of musical style*. Cambridge, MA: MIT Press.

Durugkar, I., Gemp, I. & Mahadevan, S. (2016). Generative multi-adversarial networks. arXiv preprint arXiv:1611.01673

Engel, J., Resnick, C., Roberts, A., Dieleman, S., Norouzi, M., Eck, D. & Simonyan, K. (2017). Neural audio synthesis of musical notes with wavenet autoencoders. In *Proceedings of the 34th International Conference on Machine Learning*, 70, 1068-1077.

Grierson, M., Yee-King, M., McCallum, L., Kiefer, C. & Zbyszynski, M. (2019). Contemporary Machine Learning for Audio and Music Generation on the Web: Current Challenges and Potential Solutions. *Proceedings of the International Computer Music Conference*, New York.

Griffin, D. & Lim, J. (1984). Signal estimation from modified short-time Fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2), 236–243.

Holzapfel, A., Sturm, B. L. & Coeckelbergh, M. (2018). Ethical Dimensions of Music Information Retrieval Technology. In *Transactions of the International Society for Music Information Retrieval*, 1(1), 44–55.

Holzmann, G. (2009). Reservoir computing: a powerful black-box framework for nonlinear audio processing. In *International Conference on Digital Audio Effects (DAFx)*.

Karras, T., Aila, T., Laine, S. & Lehtinen, J. (2017). Progressive growing of GANs for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196

Kiefer, C. (2019). Sample-level sound synthesis with recurrent neural networks and conceptors. *PeerJ Computer Science*, 5, DOI 10.7717/peerj-cs.205

Knobel, M., and Lankshear. C. (2008). Remix: The art and craft of endless hybridization. *Journal of Adolescent & Adult Literacy* 52(1), 22-33

Kumar, R. & Mahapatra, A. K. (2009). Concept of brain transplant in pre-historic era. *Childs Nerv. Syst.* 25, 393. https://doi.org/10.1007/s00381-008-0733-2

LeRoux, J., Kameoka, H., Ono, N. & Sagayama, S. (2010). Phase initialization schemes for faster spectrogram-consistency-based signal reconstruction. In *Proceedings of the Acoustical Society of Japan Autumn Meeting*, 3.

Magnusson, T. (2019). *Sonic writing: technologies of material, symbolic, and signal inscriptions*. Bloomsbury Academic.

McGinn, C. (1989). Can We Solve the Mind-Body Problem? *Mind*, 98(391), 349-366.

O'Connell, M. (2017). *To be a machine: Adventures among cyborgs, utopians, hackers, and the futurists solving the modest problem of death*. London: Granta.

Roche, F., Hueber, T., Limier, S. & Girin, L. (2018). Autoencoders for music sound modeling: a comparison of linear, shallow, deep, recurrent and variational models. arXiv preprint, arXiv:1806.04096.

Sarroff, A. and Casey, M.A. (2014). Musical audio synthesis using autoencoding neural nets. In *Proceedings of the International Society for Music Information Retrieval Conference*, 2014

Singh, S. (2013). *The Simpsons and their mathematical secrets*. London: Bloomsbury.

Snowdon, P. F. (1991). Personal identity and brain transplants. *Royal Institute of Philosophy Supplements* 29, 109-126.

Stöter, F.R., Uhlich, S., Liutkus, A. & Mitsufuji, Y. (2019). Open-unmix-a reference implementation for music source separation. The Journal of Open Source Software. 4(41), 1667, https://doi.org/10.21105/joss.01667

Sturm, B. L. T., Iglesias, M., Ben-Tal, O., Miron, M. & Gómez, E. (2019). Artificial Intelligence and Music: Open Questions of Copyright Law and Engineering Praxis. *Arts*, 8(115), doi:10.3390/arts8030115

Taylor, K. (2017). *Brain Washing: The Science of Thought Control* (2nd edition). Oxford: Oxford University Press.

Tegmark, M. (2018). *Life 3.0: Being Human in the Age of Artificial Intelligence*. London: Penguin.

P. M. Todd and G. M. Werner. (1999). Frankensteinian methods for evolutionary music. In N. Griffith & P. M. Todd (Eds.), *Musical networks: parallel distributed perception and performance*, Cambridge, MA: MIT Press, 313-340.