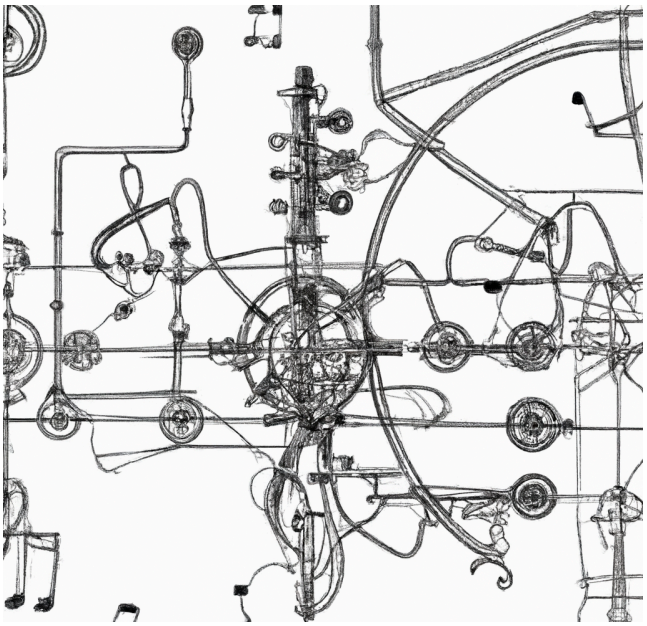
MSc. Software Design and AI (AL_KSAIM_9_1)

Advanced Machine Learning (Dr. Yuhang Ye)

Assignment 1- Literature Review



NEURAL NETWORKS IN AUDIO ENGINEERING AND MUSIC PRODUCTION

Jack Cawley

A00315252@student.tus.ie

5th April 2024

1. Introduction

1.1 Music and Audio Production Technology

Music Recording has historically benefited from adopting new technologies, which has driven innovation and creativity throughout all areas of the industry. Audio Engineering involves using electronic devices to manipulate timbre through frequency equalisation and to recreate ambience through manipulation of the time response of the audio signal. In the digital domain, these devices use mathematical algorithms to model filters and reverberation characteristics. The adaptation of digital techniques has given rise to the Digital Audio Workstation (DAW) which uses multiple software packages to perform a number of necessary tasks in the production process. This has transformed the entire process of writing, recording, producing and mixing music into a non-linear environment where all of the skills needed to create music are in use at every part of the process.

The use of Artificial Intelligence (AI) in the Audio Production industry is increasing rapidly. New AI architectures and methods are being developed all the time, and software manufacturers such as Adobe and Izotope are using AI to create audio plugins which are faster and more precise at common audio editing tasks such as noise removal. In addition, AI audio processing offers possibilities that go beyond those offered by existing software, some of which will be explored in this paper.

2. Neural Networks in Audio Applications

2.1 Neural Network Architectures

Neural Networks are being used for a range of tasks in audio processing, and in practice specific architectures have been found to be useful at extracting specific features from audio. Convolutional Neural Networks (CNNs) similar to those used in image processing have been found to be effective in capturing pitch, timbre and rhythm. Recurrent Neural networks (RNNs) are useful in capturing sequential and temporal dependencies in speech and music. Attention-based models can be used to focus on specific parts of the audio, such as keywords or emotions. It is also possible to combine different types of neural networks to create more powerful models.

This paper will look at some audio applications which use different neural network architectures. First, it is necessary to look at some methods used in data preparation for audio.

2.2 Audio File Formats and Data Preparation

Audio is most commonly used in the waveform format, which uses a train of audio ‘samples’ which represent the variation in sound intensity over time. These files use high resolution samples of up to 24 bits, and high sample rates, often between 16,000 to 96,000 samples per second depending on the application. Data preparation for audio often requires conversion to a more manageable format, and some of the most commonly used formats are described below.

A commonly used visual representation for audio data is the Spectrogram. A waveform and spectrogram representation of a short piece of audio is shown in Fig. 1 below.

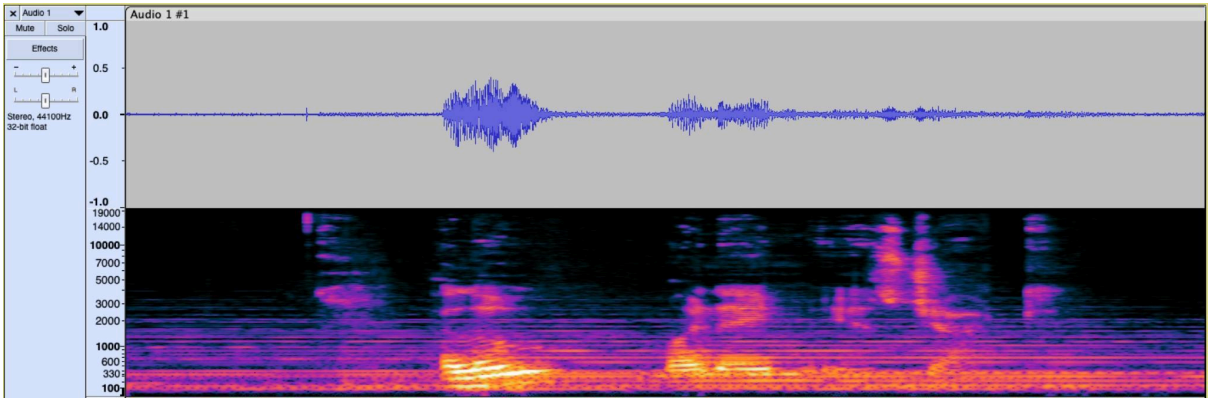


Fig. 1 Waveform and Spectrogram view of a voice sample, shown in Audacity

The Spectrogram displays the acoustic energy in coloured areas corresponding to specific frequencies plotted against time, as opposed to a series of instantaneous amplitude values. This makes it easier to visualise the timbre and tonal content of the audio clip. There are many types of spectrogram, but the most commonly used for this application is the Mel-Spectrogram, which uses frequency divisions that are derived from the response of the human ear.

A common method to derive a Spectrogram is to use a Fourier Transform, which converts a function (in the case of an audio waveform, a time-variant function) into a form that describes the frequencies present in the original function. The result is referred to as the ‘frequency domain’ representation of the original function. An illustration of the frequency domain versus time domain representation is in Fig. 2 below.

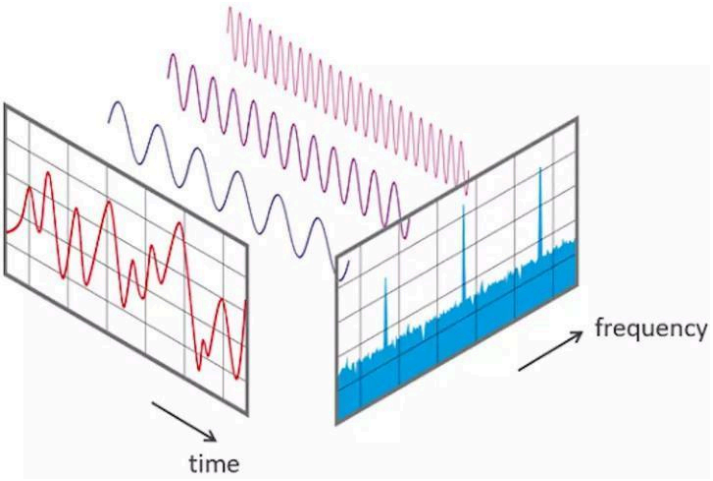


Fig. 2 Illustration of Frequency Domain vs Time Domain representation (Dieleman, 2020)

Short-Time Fourier Transform (STFT) is the most common application of the Fourier Transform mentioned earlier. As the audio signal- or function- has been converted to data, a STFT is calculated on short sections of the full audio file by applying it to a windowing function. This windowing function is applied sequentially across the entire audio file in a series of overlapping windows. The use of a window function creates a tradeoff in accuracy, where a wider window gives better frequency resolution but poorer time resolution, and a narrow window gives poorer frequency resolution but improved time resolution.

Mel- Frequency Cepstral Coefficients (MFCCs) are numerical coefficients derived from spectral properties of audio signals and provide information about the phonetic and vocal attributes. MFCCs allow the data derived from the Spectrogram to be further summarised and reduced in size. Depending on the audio content and the purpose of the model, anything from 12 to 40 coefficients may be used. The Librosa library offers many options to extract and make use of MFCCs.

2.3 CNN Architecture and Audio Classification

CNN architectures generally employ multiple convolutional layers with a Rectified Linear Unit (ReLU) activation function, pooling layers, fully connected layers and a Softmax output layer. A sample diagram of a CNN architecture is shown in Fig. 3 below.

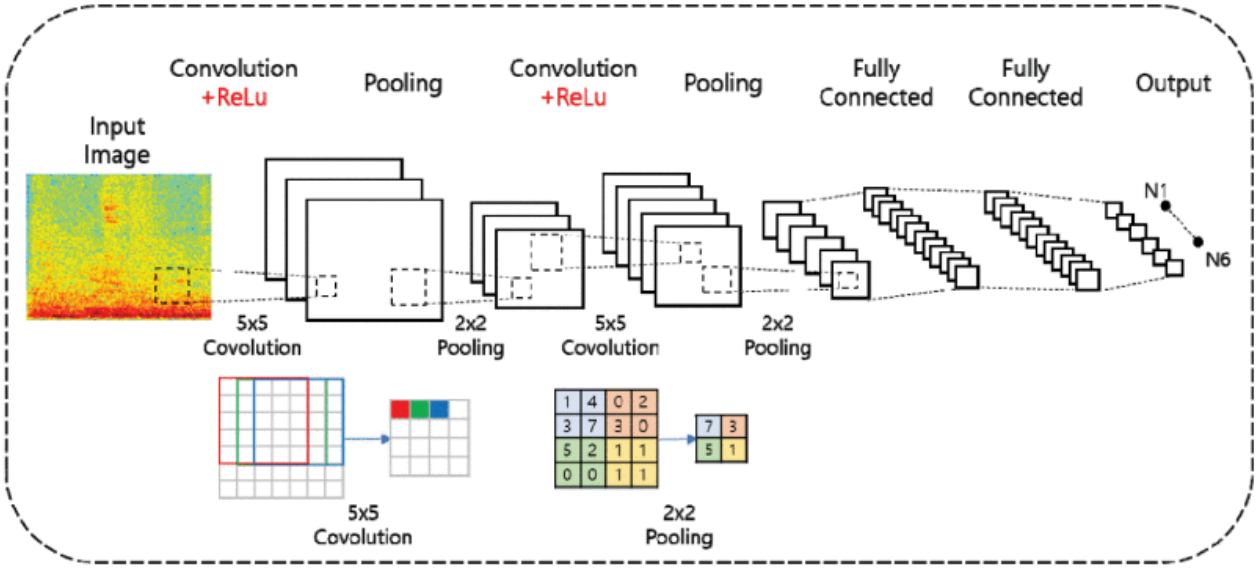


Fig. 3 Sample CNN network architecture for audio classification (G. Park and S. Lee, 2020)

CNN architectures like the one shown are commonly used to process image data, and in audio classification applications, they are often trained using spectrogram images of the audio data. In this example, each convolutional layer uses a convolution grid to analyse the spectrogram image in sections and extract feature information from that. This is followed by a pooling layer to reduce the dimensions of the feature map. This improves efficiency, and also allows the model to focus on the most important features, improving the accuracy. The reduced feature map is then passed to the ReLU function, which is used to extract the features from the map. This function effectively removes all negative values and passes positive values through. This enhances the non-linearity of the model and prevents overfitting. The fully connected layers are used to process the output from the convolutional layers after the data is converted from 2-D to a 1-D format, in a process known as flattening. The Softmax layer uses a logistic sigmoid function which produces a probability distribution over a range of possible output classifications. It is very useful for multi-class classification, as it produces a range of probabilities for each output class that sums to 1.

In "Efficient Music Genre Classification with Deep Convolutional Neural Networks" (W.Suo, 2022) a CNN architecture developed for image processing known as VGG16 was repurposed to perform a Music genre Classification task. A diagram of the VGG16 model is shown in Fig. 4 below.

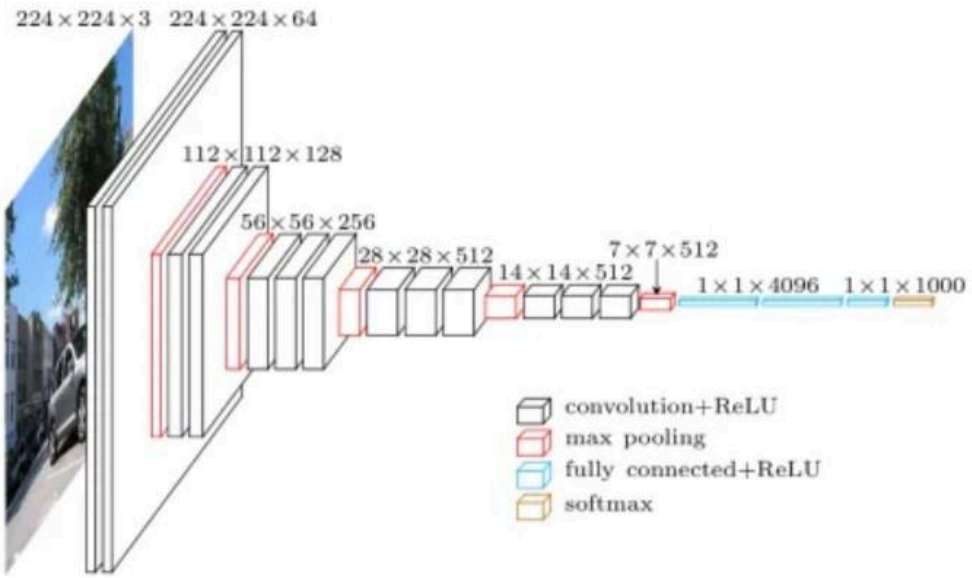


Fig. 4 VGG16 Model Architecture (Simonyan, K. and Zisserman, A. 2015)

This implementation uses four stacks of convolutional layers with ReLU activation function, with a Max pooling layer between each layer to reduce the dimensions. To improve performance on the chosen dataset, an additional three dense (fully connected) layers are added, each with ReLU activation. The loss function used is Categorical Cross Entropy, together with the Stochastic Gradient Descent (SDG) optimiser.

This VGG16 network was trained to identify 10 music genres from the commonly used GTZAN dataset (Tzanetakis et al, 2001). This contains 1000 audio files in 22050 Hz, 16-bit Mono .wav format, covering 10 music genres with 100 examples of each genre. The audio files were converted to 2-dimensional image representations using a Short Time Fourier Transform (STFT) and a Mel-Frequency Spectrogram. The model‘s accuracy and speed was compared to a human baseline model. This CNN gave an accuracy of 98% on the training set and 68.7% on the test set. While the test accuracy shows room for improvement, it vastly outperformed the human baseline test, which only scored 43.3% - 14 correct results from a given set of 90 samples.

2.4.1 Source Separation using Autoencoder Structure

Stoller, Ewert and Dixon (2018) describes an example of a source separation application designed to separate individual elements- vocals, drums, bass, and other accompaniment- from an already mixed piece of music. They begin by identifying that existing models for audio source separation operate on the magnitude spectrum, discarding phase information which can adversely affect model performance but also has an appreciable negative effect on audio quality. To improve this, they developed the ‘Wave-U-Net’, an updated version of the ‘U-Net’ a convolutional model used for image separation applied to biomedical data. The ‘Wave-U-Net’ technique implements source separation entirely in the time domain. A diagram of the Wav-U-Net architecture is shown in Fig. 5 below.

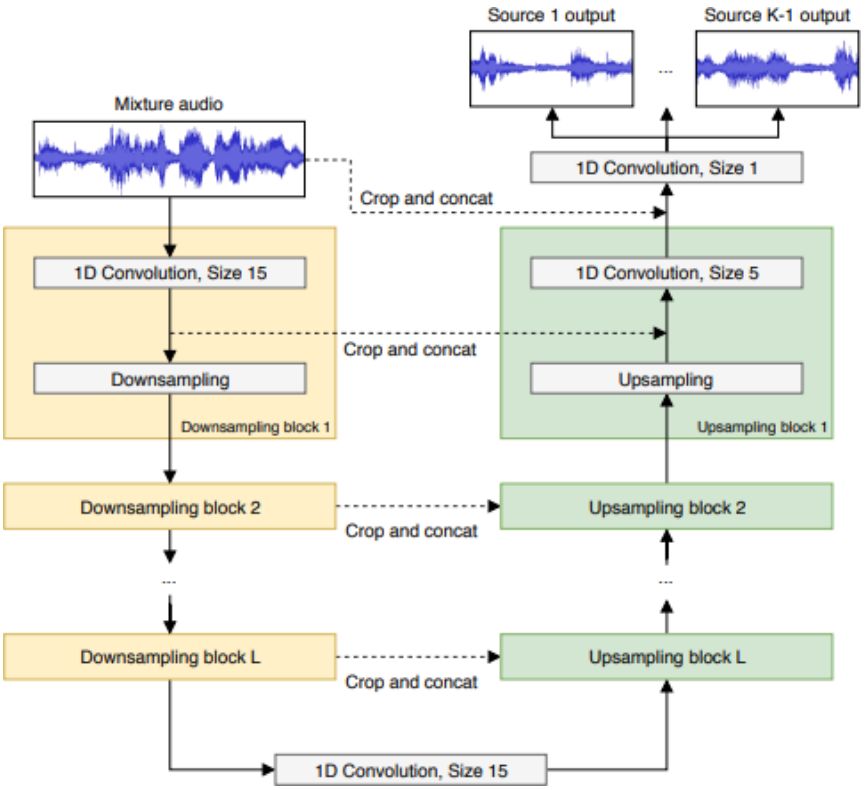


Fig. 5 Diagram of Wave-U-Net architecture with K sources and L layers

This model takes as input an audio waveform which contains a mix of musical elements. The encoder uses convolutional layers followed by downsampling blocks using max pooling. This extracts high-level features from the input, gradually reducing the dimensions of the inputs while increasing the number of feature channels. This creates the 'bottleneck' which is a lower resolution to the input, but contains a higher number of features. The 'crop and concatenate' connections serve as skip connections which allow the decoder to access some information from the encoding path at multiple levels of abstraction. These are combined using upsampling blocks which provide multi-scale features used to make predictions. Linear interpolation is used in the upsampling to avoid spurious frequencies in the output audio known as 'aliasing'. The network contains L levels, with each layer operating at half the time resolution of the previous one. For each of the K sources to be estimated, the model returns predictions for each source audio sample.

The model is trained on the MUSDB18 multi-track music database (Rafii et al., 2018). This contains 150 full length songs in a specific STEM file format- each file contains a 'mixture' track, a 'vocals' track, a 'drums' track, a 'bass' track and a 'rest of the accompaniment' track. The data set is partitioned with 100 songs in the 'training' partition and 50 songs in the 'test' partition. For this application, 75 songs from the 'training' partition are used to form a training set, and the remaining 25 are used as a validation set, which facilitates early stopping. The training process uses pairs of mixed audio tracks and separated 'stem' tracks. These provide a 'ground truth' to compare model predictions to, and the

training process applies a loss function which measures the discrepancy between the predictions and the stems from the dataset. The loss function used is Mean Squared Error (MSE) in conjunction with the ADAM optimiser. Final performance is evaluated using all 50 songs from the 'test' partition. Some data augmentation was performed at both stages, where source signals were reweighted by a scalar of 0.7 or 1.0 and then recombined with the new weighting.

In testing, this model outperformed the spectrogram-based U-Net architecture it is based on. The authors suggest that the data was limited in size and that a better assessment could be made with a more complete data set. The paper states some issues with the signal-to-distortion (SDR) metric normally applied to evaluate source separation performance. Specifically, outliers are produced when the SDR is applied to tracks containing long sections of silence, which skews the overall score. This would occur very often with isolated vocal tracks for example.

2.4.2 Source Separation using LSTM

Open-Unmix (Stöter et al., 2019).was based on a bi-directional LSTM model from Uhlich et al. (2017) using two deep neural networks- a feed-forward and a recurrent neural network- which applied data augmentation during training to avoid overfitting. A diagram of the specific Open-Unmix architecture is shown in Fig. 6 below.

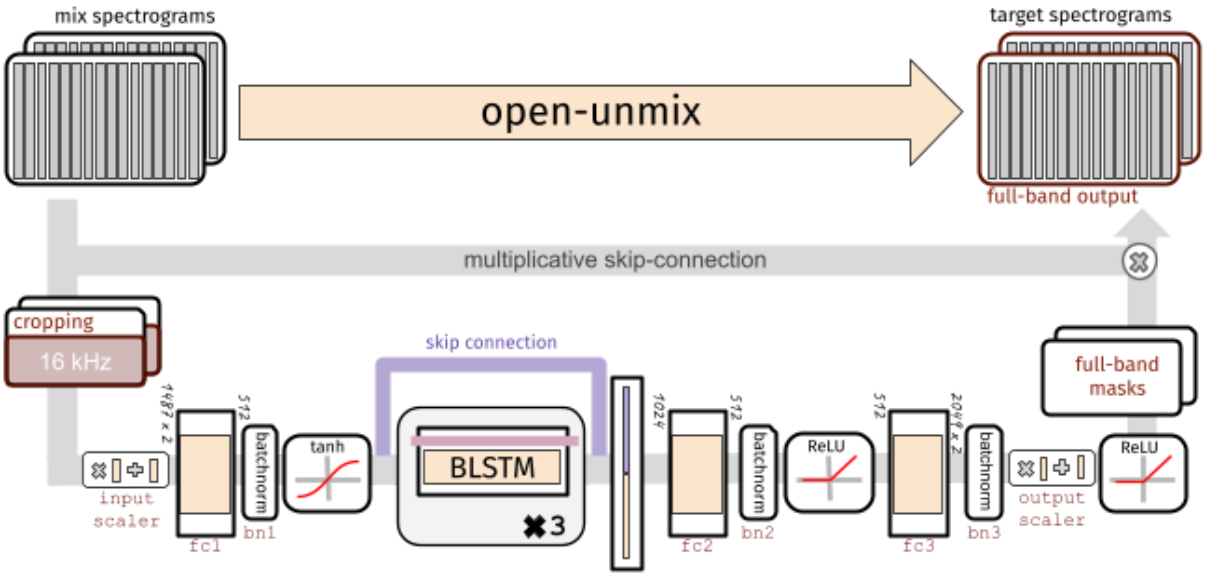


Fig. 6 Diagram of Open-Unmix BLSTM source separation architecture (one source)

For separating multiple sources, Open-Unmix uses a model trained for each specific source- vocal, drums, bass, and 'other' (a remainder after the other separations are complete). Each model uses a three-layer bi-directional deep LSTM. The input to the network is a spectrogram of the mixed audio signal. This is fed into the bi-directional LSTM, which processes the input spectrogram in both forward and backward directions. The LSTM layer outputs a sequence of hidden states which serve as an encoded version of the input spectrogram, which contain temporal dependencies which allow the model to better understand the audio structure. The model learns to predict the magnitude spectrogram of the target source from the magnitude spectrogram of the mixed input- the prediction is derived from applying a mask on the input. The model output consists of separated spectrograms for each of the musical stems used in the training set, which are then used to encode the audio for each stem.

The model can accept as input either a sample stream or a pre-computed magnitude spectrogram, but if using a sample stream as input, it computes the STFT using Torch or Asteroid Filterbanks (Pariente et al., 2020). The spectrogram is then standardised to a standard deviation across all frequency bins and batch normalisation is applied at several stages to make the model more immune to signal gain variation. Skip connections are employed to preserve the flow of low-level features through the network. The loss function used is SGD, and the optimiser is ADAM. The model does not operate on the input spectrogram resolution- after normalisation, the network learns to compress the frequency and channel axis of the model to make it converge faster. The recurrent nature of the bidirectional LSTM network allows the model to be trained on audio samples of any duration.

Open-Unmix showed state of the art performance compared to other similar separation models evaluated for SiSec 2018 (Fabian-Robert et al., 2018), the community-based Signal Separation Campaign sponsored by Native Instruments.

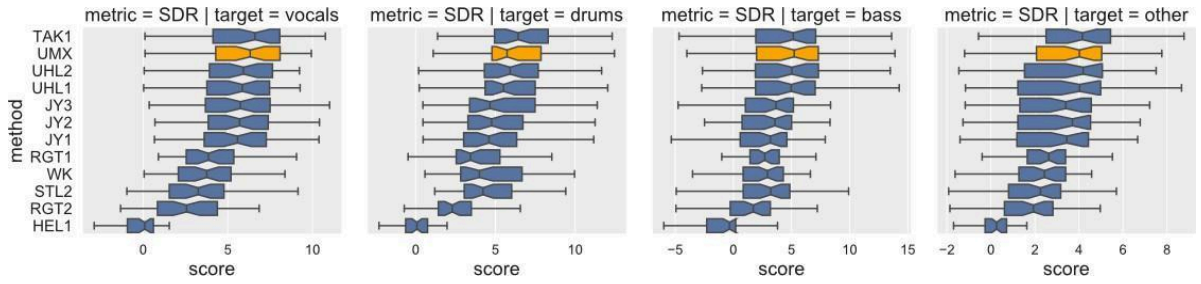


Fig. 7 Performance of Open-Unmix compared to participants in SiSec 2018

In the background to the original Open-Unmix paper, the authors note that there is no 'traditional' signal processing solution available to achieve source separation, and that any attempts in that domain benefited from some prior knowledge of how the recording was made, or were effective on a limited range of recordings. They argue that the only successful approach is through use of deep

learning. With this in mind, they point out the disparity between the amounts of suitable data available for the purpose of source separation compared to, say, those available for voice recognition applications. This, they suggest, is due to the copyright implications of releasing isolated vocal or instrument stems from commercially available works. The oft-cited MUSDB18 dataset relies on musicians operating under the Creative Commons licence, and they credit these as being essential to the continued development in this data-driven area of research. Their implementation of source separation, Open-Unmix, is designed to be open-source and adaptable by the wider community. In addition, they have shared their resources- code, datasets and other resources- with the SiSec community, with the intention of providing something towards the development of an industry standard for source separation. Many current advances in this area are based on research using the Open-Unmix architecture and code resources.

2.5 Audio Effects Modelling using Temporal Convolutional Networks

In 'Randomised Overdrive Neural Networks' (Steinmetz et al., 2021) a novel approach is taken in attempting to model an audio effect unit. In this implementation, a Temporal Convolutional Network (TCN) architecture is employed, but with a very different approach. The fundamental operation of a TCN is that it performs convolution not in the spatial domain, but in the time domain. This is very useful for audio, as the most important features in audio are observed in the time domain. Because convolution is occurring in the time domain, the receptive field is increased by using dilated convolutions, which simply insert zeros in between the extracted feature values. This allows long-range dependencies to be captured without significantly affecting the computational load. These convolutional layers are stacked on top of one another, allowing the network to capture different levels of abstraction in the audio signal- with the higher layers learning the more complex features. A PReLU (Programmable Rectified Linear Unit) activation function is used, which allows the amount of non-linearity to be controlled.

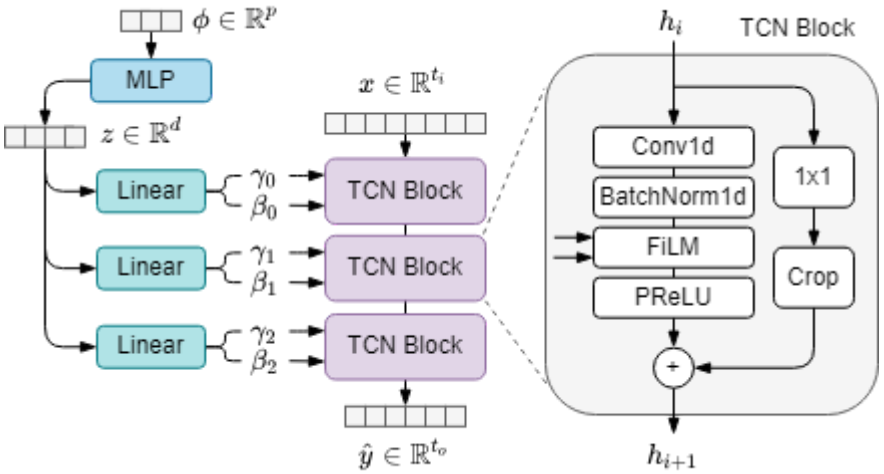


Fig. 8 Temporal Convolutional Network architecture used to emulate an audio compressor

In this experiment, the model is not trained on source data in the traditional sense. This architecture operates on the principle that a very different audio effect can be achieved by simply changing the weights applied to the same network- here, the network weights become parameters which are set by the user in real-time. This is facilitated by the Feature-wise Linear Modulation, or FiLM block which applies an affine transformation to conditioning parameters γ and β (scale and bias). A Multilayer Perceptron (MLP) block takes input from the user and maps the parameters to the FiLM blocks.

This research built an audio effect plugin to demonstrate the capabilities of this approach, a screenshot of the GUI is shown in Fig. 9 below.

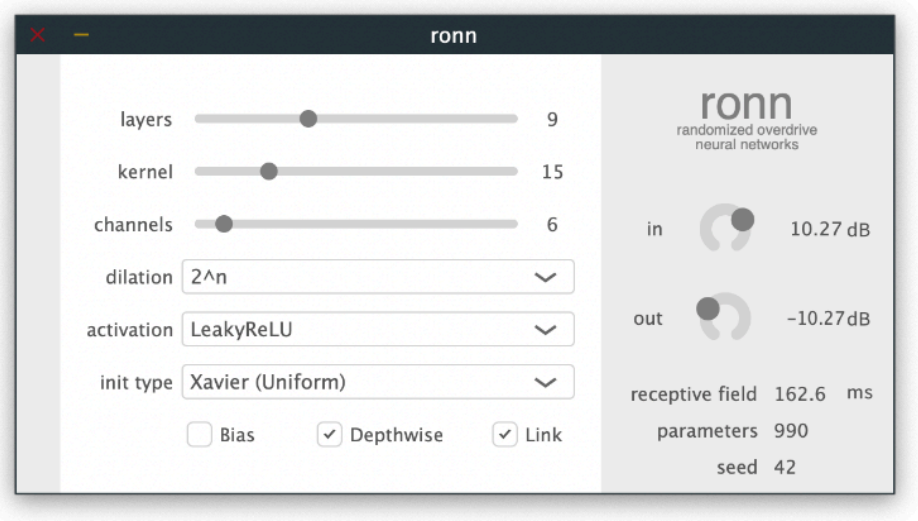


Fig. 9 Control Interface for Randomised Overdrive Neural Network Plugin

The results from this plugin are often unpredictable, but they are certainly interesting and not easily achievable using traditional effects devices.

3. Conclusions

Neural Networks are being applied to Music Production and Audio Engineering with excellent results. CNN architectures are capable of carrying out some tasks more efficiently, such as music genre classification. Autoencoder and LSTM architectures achieve previously impractical results in extracting source elements from mixed audio tracks. Some recent experimental research using a TCM architecture explores using a neural network as a user-controllable audio effect device.

These successes are bringing AI techniques into the same workspace as existing audio software and hardware, where they are becoming essential to the modern process of composing and producing music. Applications in Music Generation and Sound Synthesis are also in development.

Collaboration and sharing of work in this area is vital to the success of AI techniques in this field. Luckily, there appears to be a trend towards open-sourcing of code and models, and sharing of valuable datasets for audio applications.

The future sounds good.

References

Davis, S. and Mermelstein, P. (1980) Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Transactions on Acoustics, Speech and Signal Processing, 28, 357-366.

Dieleman, S. (2020). *Generating Music in the Waveform Domain*. [online] Available at: http://https://sander.ai/2020/03/24/audio-generation.html[Accessed 5th May 2023].

Fabian-Robert and Liutkus, Antoine and Ito, Nobutaka, (2018) "The 2018 Signal Separation Evaluation Campaign", "Latent Variable Analysis and Signal Separation: 14th International Conference, LVA/ICA 2018, Surrey, UK", pages="293--305"

Pariente, M., Cornell, S., Cosentino, J., Sivasankaran, S., Tzinis, E., Heitkaemper, J., Olvera, M., Stöter, F.R., Hu, M., Martín-Doñas, J.M. and Ditter, D., (2020). Asteroid: the PyTorch-based audio source separation toolkit for researchers. *arXiv preprint arXiv:2005.04132*.

Park, G and Lee, S. (2020) "Environmental noise classification using convolutional neural networks with input transform for hearing aids", Int. J. Environ. Res. Public Health, vol. 17, no. 7, pp. 2270

Rafii, Z., Liutkus, A., Stöter, F.R., Mimilakis, S.I. and Bittner, R., (2017). The MUSDB18 corpus for music separation.

Simonyan, K. and Zisserman, A. (2015) Very Deep Convolutional Networks for Large-Scale Image Recognition, April 2015, [online] Available: https://arxiv.org/abs/1409.1556.

Steinmetz, Christian & Reiss, Joshua. (2021). Randomised Overdrive Neural Networks

Stoller, Ewert, Dixon. (2018) “WaveU-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation”, 19th International Society for Music Information Retrieval Conference, Paris, France, 2018.

Stöter, F.R., Uhlich, S., Liutkus, A. and Mitsufuji, Y., (2019). Open-unmix-a reference implementation for music source separation. *Journal of Open Source Software*, *4*(41), p.1667.

Suo, W. (2022) "Efficient Music Genre Classification with Deep Convolutional Neural Networks," 2022 5th International Conference on Data Science and Information Technology (DSIT), Shanghai, China, 2022, pp. 01-05

Tzanetakis, George and Essl, Georg and Cook, Perry (2001), Automatic Musical Genre Classification Of Audio Signals, The International Society for Music Information Retrieval