Deep Dream for Sound Synthesis

Daniel Bisig
Instituto Stocos, Spain
e-mail: daniel@stocos.com
Zurich University of the Arts, Switzerland
e-mail: daniel.bisig@zhdk.ch

Ephraim Wegner
Offenburg University, Germany
e-mail: ephraim.wegner@hs-offenburg.de

Abstract

In 2015, Google engineer Alexander Mordvintsev presented *DeepDream* as technique to visualise the feature analysis capabilities of deep neural networks that have been trained on image classification tasks. For a brief moment, this technique enjoyed some popularity among scientists, artists, and the general public because of its capability to create seemingly hallucinatory synthetic images. But soon after, research moved on to generative models capable of producing more diverse and more realistic synthetic images. At the same time, the means of interaction with these models have shifted away from a direct manipulation of algorithmic properties towards a predominance of high level controls that obscure the model's internal working.

In this paper, we present research that returns to *DeepDream* to assess its suitability as method for sound synthesis. We consider this research to be necessary for two reasons: it tackles a perceived lack of research on musical applications of *DeepDream*, and it addresses *DeepDream*'s potential to combine data driven and algorithmic approaches.

Our research includes a study of how the model architecture, choice of audio datasets, and method of audio processing influence the acoustic characteristics of the synthesised sounds. We also look into the potential application of *DeepDream* in a live-performance setting. For this reason, the study limits itself to models consisting of small neural networks that process time-domain representations of audio. These models are resource-friendly enough to operate in real time.

We hope that the results obtained so far highlight the attractiveness of *DeepDream* for musical approaches that combine algorithmic investigation with curiosity driven and open ended exploration.

1. Introduction

Large scale deep-learning models such as Chat-GPT[1], MidJourney[2], and MusicLM[3] have recently been made available to a wider audience. The capability of these models to generate a large variety of high quality text, images, or sounds and the ease with which these models can be used by non-experts has brought the field of generative machine learning to the forefront of public attention.

Unfortunately, these models don't lend themselves very well to generative approaches that situate the ideation and development of algorithms at the core of their practice. The reason for this involves both core aspects of the models' functioning and practical aspects of their usage. A user's influence on the model's behaviour is limited to the provision of high level controls often in the form of text prompts that steer the content of the generated output. This comes at the cost of the user's understanding of the models' architectural design and operational functioning, both of which would be a prerequisite to adapt the models for one's own artistic goals. Furthermore, the sheer size of these models renders it unfeasible for regular users to train these models on material that they might have created and/or collected on their own. This prevents these models from generating output that is representative of the potentially highly idiosyncratic aesthetic and thematic interests of a specific user.

The DeepDream technique (DD) represents one of the earliest deep learning based approaches to generate synthetic images. While this technique is clearly inferior to state of the art generative models in terms of diversity and quality of generated media, it excels at exposing the relationships between the algorithmic properties of the model and the characteristics of the generated media. Furthermore, DD can be used in combination with models that possess a modest number of parameters. These models can be trained on relatively small datasets and are thereby able to capture some of the idiosyncratic properties inherent in these datasets. Furthermore, these models are able to generate media very quickly and can therefore be used in real-time.

This publication focuses on the use of DD for generating audio. The work described is divided into two parts. The main part consists of an analytical study of the relationships between training data, model architecture, DD parameters on the one hand and the acoustic characteristics of the generated audio on the other hand. The second part represents an example application of DD for music composition.

2. Background

In 2015, Alexander Mordvintsev presented DD as a technique to improve the understanding of the feature extraction capabilities of deep convolutional neural networks (DCNN) that have been trained on image classification tasks [1]. The technique employs a visualisation procedure that functions by running a training process on an input image instead of the network. During this procedure, the input image is iteratively modified through a feature inversion process that maximises the activity of one or several chosen network layers and feature maps. After several iterations, the input image increasingly exhibits those features that are recognised by the chosen network layers and feature maps. With the aid of this visualisation procedure, it can be shown that lower network layers recognise basic

1 ChatGPT: chat.openai.com
2 MidJourney: docs.midjourney.com
3 MusicLM: aitestkitchen.withgoogle.com

patterns, while higher network layers recognise more complex composite patterns. For a more exhaustive introduction into DD, the reader is referred to a recently published systematic review [2].

While much of the interest and popularity around DD has focused on its application in the image domain, a few researchers and artists have also explored its usefulness for audio.

Ardila and colleagues applied DD on a DCNN that has been trained with raw waveforms to predict collaborative filtering track embeddings [3]. The authors studied the filters learned by the first layer and the corresponding spectra. Mishra and collegues employed DD on a DCNN that has been trained with Mel Spectrograms on a singing voice detection task [4]. Their findings suggest that at the deepest level, convolution layers preserve temporal and harmonic structures while fully connected layers do not.

In a more artistically motivated research project, Galac and Delgadino employed creative variations on DD [5]. These variations are based on the integration of audio filters into the activation maximisation function. The authors used the pretrained YAMNETI [6] model. They conclude that the use of a hard-cut filter in the maximisation function produces more interesting sonic results than the original DD. Another artistic research project has been conducted by Herrmann [7]. The author trained a DCCN on a scaleogram representations of audio clips. What mainly distinguishes this project from previous endeavours is that training is self-supervised instead of supervised. Self-supervised training causes the network to perceptually discriminate between different audio features instead of learning a mapping from input to explicit labels.

The work presented in this publication complements and extends this prior research in two directions. Similarly to [3] and [4], it conducts a systematic analysis of the feature detection capabilities of different network layers and feature maps. But in addition, it also conducts a comparison between different audio training sets and network architectures. Similarly to [5] and [7], it highlights the creative possibilities of DD for musical applications. But in addition, it exemplifies how DD can be adopted for creating a musical composition.

3. Implementation

The study presented in this publication is based on our own implementation and training of neural networks and the subsequent application of DD for generating audio. The implementation involves a selection of three different publicly available audio datasets, the design of two different DCCN, their training on audio classification tasks, the modification of an existing DD algorithm, and the use of audio feature analysis to quantify the characteristics of the generated audio. Each of these implementation aspects is described in more detail in the subsequent sections.

3.1 Datasets

We decided to work with several different audio datasets as training data to compare their respective influence on the audio generated with DD. Our criteria for selecting the datasets were as follows: the audio sampling rate is at least 44100 Hz, each audio recording is at least a few seconds long, and the number of classes is at least 100. The last criteria was based on our assumption that a model that recognises a large number of classes discriminates between more nuanced audio characteristics than a model that recognises a small number of

classes. We chose three audio datasets that full-fill these criteria but that differ from each other with regards to the diversity of the recorded sounds. The Freesound dataset[4] exhibits the largest acoustic diversity. It contains 200 classes of sound events that were drawn from the AudioSet Ontology. The Musical Instruments dataset[5] exhibits intermediate acoustic diversity. It contains 255 classes of instruments played at different loudness levels and with different tone types. The AISHELL-3 dataset[6] exhibits small acoustic diversity. It contains 218 classes of different Chinese subjects speaking short sentences in Madarin. Throughout the remainder of the text, the three datasets will be referred to as *Events* for the Freesound Dataset, *Instruments* for the Musical Instruments dataset, and *Speech* for the AISHELL-3 dataset.

3.2 Models

Two different convolutional models have been designed using the PyTorch[7] deep learning framework. Both models take as input normalised waveforms of raw audio of one second duration and a sample rate of 44100 Hz and produce as output the log-probabilities for each class. In both models, the neural network consists solely of 1D convolution layers that perform convolution in the temporal domain. This design is informed by the finding that fully connected layers fail to capture crucial aspects of audio [4]. The model architecture is very simple and differs between the two models only in terms of the number of convolution layers. One model (*Model1*) consists of four convolution layers and possesses roughly 400000 trainable parameters. The other model (*Model2*) consists of eight convolution layers and possesses roughly 300000 trainable parameters.

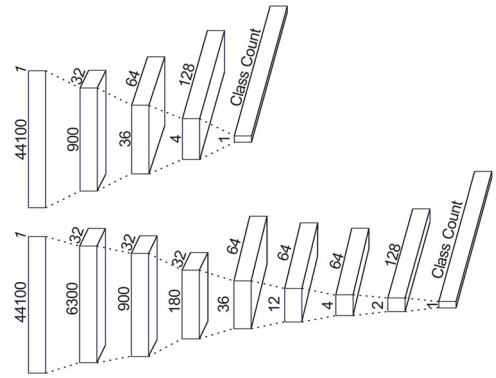

Figure 1: Architectures of Model1 and Model2. The deepness of the layers increases from left to right. For each layer, the vertical number represents the length of the input vector and the diagonal number represents the number of channels.

model	layer	ksize	stride	padding
1	0	85	49	42
	1	43	25	21
	2	17	9	8
	3	7	4	3
2	0	129	7	64
	1	65	7	32
	2	33	5	16
	3	17	5	8
	4	9	3	4
	5	5	3	2
	6	3	2	1
	7	3	2	1

Table 1: Layer Properties. The abbreviations stand for: layer for layer index, and ksize for kernel size.

The layout of the architecture and the properties of the convolution layers are depicted in figure 1 and table 1, respectively. Other elements of the model architecture are the use of the Swish activation function [8] and Batch Normalisation between each layer, and the Softmax ac-

4 Freesound Dataset 50k: zenodo.org
5 Microphone Array Measurements of Musical Instruments: depositonce.tu-berlin.de
6 AISHELL-3 Open Source HI-FI Mandarin Speech Corpus: www.aishelltech.com
7 PyTorch: pytorch.org

tivation function after the output layer. The motivation to design two models that differ from each other with regards to the number of convolution layers was informed by the well known finding that layers at different depth discriminate features at different levels of complexity [1]. By performing DD on models that possess a different number of layers but are otherwise identical, insights can be gained whether the layer count affects the complexity of acoustic properties that generated audio.

3.3 Training

The two models were trained on an audio classification task. Training was performed independently with each audio dataset. Training was run for 200 epochs with a test-train split of 20%/80% and a batch size of 128. The learning rate was initially set to 10^{-3} and subsequently stepwise reduced every 100 epochs with a decay factor of 0.1. The loss function was based on negative log likelihood. For parameter optimisation, an Adam optimiser with default parameter settings was employed. Table 2 lists the classification accuracy achieved on the test set for each of the models and datasets.

model	dataset	accuracy
1	*Events*	27%
	Instruments	28%
	Speech	89%
2	*Events*	37%
	Instruments	41%
	Speech	95%

Table 2: Audio Classification Accuracy.

3.4 DeepDream

The implementation of DD follows standard practice but includes two additional processing steps: a shift by a random time offset and a low-pass filter. When employing DD on images, it has been shown that a random offset in the image coordinate system increases the diversity of the generated visual features. For this reason, this technique has been adopted for audio. The addition of a low-pass filter was inspired by the work by Galac and Delgadino [5] and has proven crucial to reduce the amount of noise present in the audio that is generated in deep layers.

In the chosen implementation, DD conducts three processing steps: 1) the waveform is modified through gradient ascent 2) the waveform is randomly offset in time 3) the waveform is low-pass filtered. The gradient ascent algorithm conducts seven processing steps: 1) the waveform is passed into the model 2) the activations of the chosen layers and feature maps are obtained 3) for each layer and activation, the mean square error loss between the activations and a zero vector is calculated 4) the error gradient is computed by back-propagation 5) the gradient is smoothed using Cascade Gaussian Smoothing 6) the waveform is modified based on the gradient 7) the waveform is normalised.

In this DD implementation, the following parameters can be varied: the content of the initial audio waveform, the maximum size of the random temporal offset, the number of iterations over which the waveform is modified, the coefficient and kernel size used for gradient smoothing, the learning rate with which the gradient is applied to the waveform, and the cut-off frequency of the low-pass filter.

3.5 Audio Feature Analysis

To assess of the acoustic characteristics of the generated audio, several audio feature are analysed: Waveform Root Mean Square, Spectral Centroid, Spectral Flatness, and Spectral Flux. The Root Mean Square (RMS) of a waveform corresponds to the *Loudness* of a sound.

The Spectral Centroid indicates where the centre of mass is located in an audio spectrum. This corresponds to the *Brightness* of a sound. The Spectral Flatness is obtained as ratio between geometric and arithmetic mean of an audio spectrum. This corresponds to the *Noisiness* of a sound. The Spectral Flux is obtained as difference of the audio spectra between two consecutive frames. This corresponds to the *Roughness* of a sound.

4. Results

Two experiments have been conducted to study the acoustic results obtained using DD. The first experiment served to assess the influence of a chosen audio dataset, model architecture, convolution layer, and feature map on the generated audio. The second experiment served to evaluate the effect of different parameters of DD on the generated audio. These two experiments and the results obtained through them are discussed in more detail in the following sections.

4.1 Experiment 1

In this experiment, a total of six different models (two model architectures, each trained with one of the three audio datasets) were systematically tested by maximising the activity of each convolution layer and feature map in turn. In each of these tests, the same parameter values for DD were used: white noise as initial audio, a maximum temporal offset size of 1, a number of iterations of 2000, a gradient smoothing coefficient of 0.5 and kernel size of 9, a learning rate of 0.01, and a low-pass cut-off frequency of 12 kHz. The audio generated in each of these tests was subsequently analysed using the selected audio features. The results of this experiment are presented as sound examples and graphs.

Since the number of feature maps and correspondingly the number of generated sounds is very large, only a small representative subset of these sounds has been made available online. For each combination of model architecture and audio dataset, this selection consists of two sounds per convolution layer. The sounds have been uploaded to the SoundCloud audio streaming service and are organised as one Playlist for each model and dataset[8] [9] [10] [11] [12] [13].

To quantify the acoustic characteristics of the generated sounds, the mean and standard deviation of each audio feature have been computed for all sounds that were generated by all the feature maps in a single convolution layer. These statistics are depicted as graphs, with one graph for each combination of model architecture and audio feature (see figures 2, 3, 4, 5, 6, 7, 8, 9). In each of these graphs, the mean and standard deviations of the corresponding audio feature are depicted for each dataset and model layer. The indices of the model layers belong to the x-axis and the audio feature values to the y-axis. The mean values of the audio features are plotted as fully opaque lines whereas the shaded bands indicate a range of one standard deviation. The primary colours of the lines

8 Experiment 1 - Audio examples generated by *Model1* trained on the *Events* dataset: soundcloud.com

9 Experiment 1 - Audio examples generated by *Model2* trained on the *Events* dataset: soundcloud.com

10 Experiment 1 - Audio examples generated by *Model1* trained on the *Instruments* dataset: soundcloud.com

11 Experiment 1 - Audio examples generated by *Model2* trained on the *Instruments* dataset: soundcloud.com

12 Experiment 1 - Audio examples generated by *Model1* trained on the *Speech* dataset: soundcloud.com

13 Experiment 1 - Audio examples generated by *Model2* trained on the *Speech* dataset: soundcloud.com

and bands represent the different datasets: Red for *Events*, Blue for *Instruments*, and Green for *Speech*. When bands overlap, the resulting mixed colours represent several datasets: Yellow for *Events* and *Speech*, Violet for *Events* and *Instruments*, Cyan for *Instruments* and *Speech*, and Gray for all datasets.

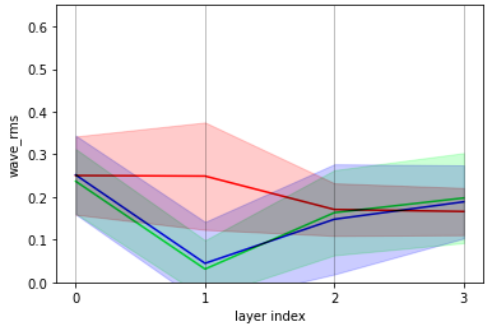

Figure 2: Model1 - Layer Effects on Loudness.

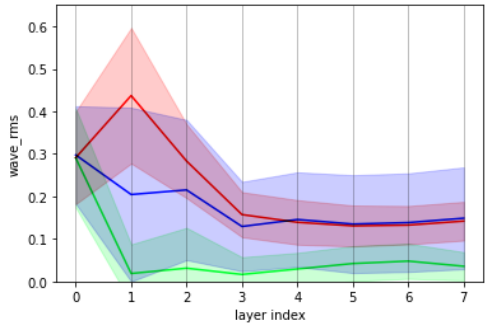

Figure 3: Model2 - Layer Effects on Loudness.

Based on this quantitative analysis and in combination with qualitative listening, several observations can be made about the effect of each layer in *Model1* or *Model2* on the acoustic characteristics of the generated sounds. These observations as summarised for each model architecture and audio feature in tables 3, 4, 5, 6, 7, 8, 9, 10. The following abbreviations are used in these tables. If the effect applies only for one or two datasets, then these dataset(s) are indicated with the single letters in brackets: (e) refers to *Events*, (i) to *Instruments*, and (s) to *Speech*. If the effect applies to all audio datasets, then the letter (a) is used.

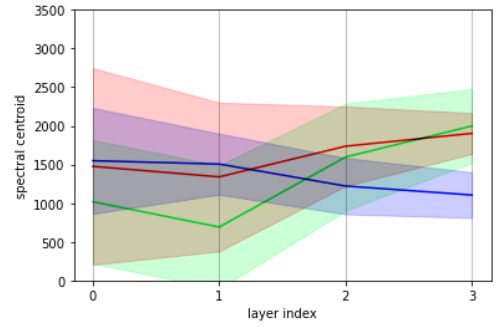

Figure 4: Model1 - Layer Effects on Brightness.

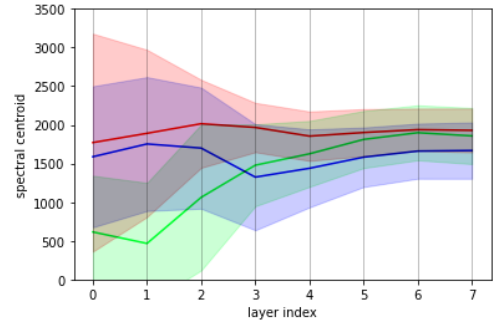

Figure 5: Model2 - Layer Effects on Brightness.

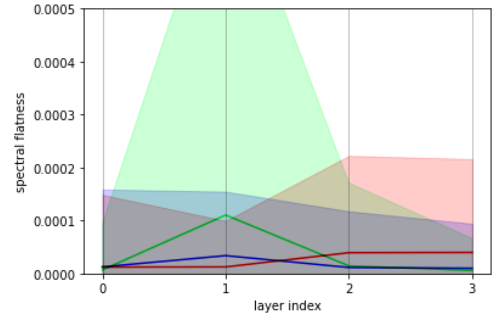

Figure 6: Model1 - Layer Effects on Noisiness.

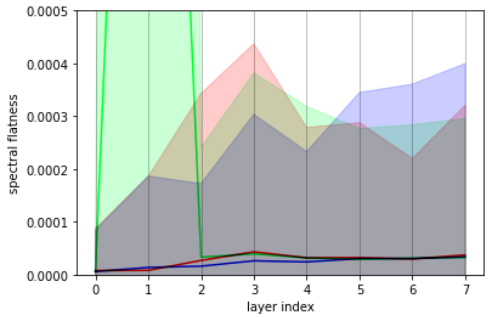

Figure 7: Model2 - Layer Effects on Noisiness.

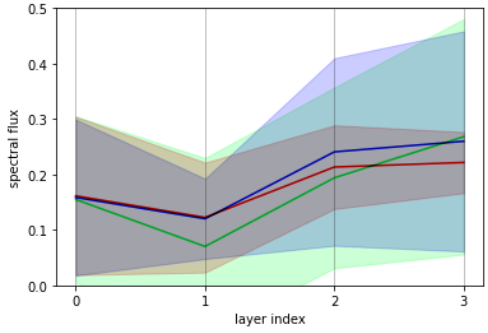

Figure 8: Model1 - Layer Effects on Roughness.

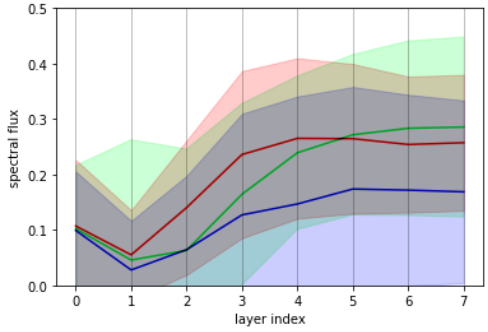

Figure 9: Model2 - Layer Effects on Roughness.

layer	loudness characteristics
0	medium loudness and small dynamic range (a)
1	low (i, s) or medium (e) loudness often concentrated in short loudness bursts (a)
2	medium loudness (a) with small (e, i) to medium (s) dynamic range
3	medium loudness (a) with small to medium dynamic range (a)

Table 3: Model1 - Layer Effects on Loudness

layer	loudness characteristics
0	medium loudness and small dynamic range (a)
1	high (e), medium (i) or low (s) loudness with small dynamic range (e) or short bursts (s) or both (i)
2	medium (e, i) or very low (s) loudness, which is constant or exhibits repetitive variations (e), short or extended bursts (s), or all of this (i)
3	low (e, i) or very low (s) loudness with repetitive (e) or irregular (i) variations, or extended bursts (s)
4	low (e, i) or very low (s) loudness, with more varied repetitions (e), in combination irregular variations (i), or extended bursts (s)
5	low (e, i) or very low (s) loudness with highly varied patterns (e, i) or combinations of burst repetitions and constant loudness (s)
6	similar to layer 5 (a)
7	similar to layer 5 (a)

Table 4: Model2 - Layer Effects on Loudness

layer	brightness characteristics
0	constant for one feature map but strongly varied across feature maps (a)
1	constant (i) or sightly varied (e, s) for one feature map and less strongly varied across feature maps (a)
2	slightly (e, i) or strongly (s) varied for one feature map and less strongly varied across feature maps (a)
3	similar to layer 2 (a)

Table 5: Model1 - Layer Effects on Brightness

layer	brightness characteristics
0	constant for one feature map but varies strongly across maps (a)
1	slightly varied (e, i) or constant (s) for one feature map and less (e, i) or similarly (s) strongly varied across feature maps
2	similar to layer 1 (a)
3	more strongly varied for one feature map (a) and less across feature maps (s)
4	varies even more strongly for one feature map (a) and less across feature maps (a)
5	varies very strongly for one feature map (a) and little across feature maps (a)
6	similar to layer 5 (a)
7	similar to layer 5 (a)

Table 6: Model2 - Layer Effects on Brightness

layer	noisyness characteristics
0	little noise (a)
1	slightly more noise that is mostly concentrated in loudness bursts (e, i) or evently distributed (s)
2	considerably (e, i) or little (s) more noise
3	considerably (e, i) or slightly more noise (s)

Table 7: Model1 - Layer Effects on Noisiness

layer	noisyness characteristics
0	little noise (a)
1	slight (e, i) or significant (s) noise
2	some noise mostly concentrated in loudness bursts (a)
3	some noise in entire sound (a)
4	similar to layer 3 (a)
5	significant noise in entire signal (a)
6	similar to layer 5 (a)
7	similar to layer 5 (a)

Table 8: Model2 - Layer Effects on Noisiness

layer	roughness characteristics
0	minimal (a)
1	similar to layer 0 (a)
2	increased spectral variations always (e, s) or sometimes (i) in sync with loudness variations
3	increased spectral variations mostly (s) or sometimes (e, i) in sync with loudness variations

Table 9: Model1 - Layer Effects on Roughness

layer	roughness characteristics
0	minimal (a)
1	even less (a)
2	increased and in sync with loudness variations (a)
3	increased and always (e, i) or sometimes in sync with loudness variations (s)
4	increased and partially in sync with and partially independent of loudness variations (a)
5	similar to layer 4 (a)
6	similar to layer 4 (a)
7	similar to layer 4 (a)

Table 10: Model2 - Layer Effects on Roughness

4.2 Experiment 2

In the second experiment, the influence of different DD parameter values on the generated sounds were evaluated with a *Model2* architecture that has been trained on the *Events* dataset. This combination of dataset and model has been chosen because it generates sounds with an acoustic characteristics that varies widely both in the temporal and spectral domain. In this experiment, audio was generated in real-time. This was achieved by conducting several DD iterations on a short excerpt of a full waveform. The excerpt was repeatedly shifted in a round-robin manner to eventually subject the entire waveform to feature inversion. In each DD run, only one DD parameter was changed at a time while the others were fixed. Fixed parameters were assigned the following values: a maximum temporal offset size of 1, a number of iterations of 10 per audio excerpt, a gradient smoothing coefficient of 0.5 and a kernel size of 9, a learning rate of 0.04, and a low-pass cut-off frequency of 12 kHz. Together with the parameter changes, the selection of convolution layers and feature maps was also changed. The variations of parameter values, convolution layers, and feature maps followed a strict sequence which is shown in table 11. In each DD run, the audio waveform was initialised with white noise and then modified by stepping through the sequence of settings. A 20 seconds delay was applied before applying the step. Between each subsequent step, a 20 seconds pause was applied. These durations were chosen to allow the acoustic characteristics of the generated sound to stabilise before the next change. Changes in layers and feature maps were offset by 10 seconds to changes in DD parameters. This allowed to observe the effect of each of these changes separately.

layer indices	0, 1, 3, 8
feat. map indices	3, 3, 4, 8
temporal offset	1, 16, 64, 128, 512
iteration count	1, 2, 4, 7, 10
grad. coeff.	0.1, 0.3, 0.5, 0.7, 0.9
grad. kernel-size	3, 5, 9, 13, 17
learning rate	0.02, 0.04, 0.08, 0.12, 0.16
cutoff freq. (kHz)	2, 4, 8, 12, 16

Table 11: Experiment 2 - DD Parameter Sequences

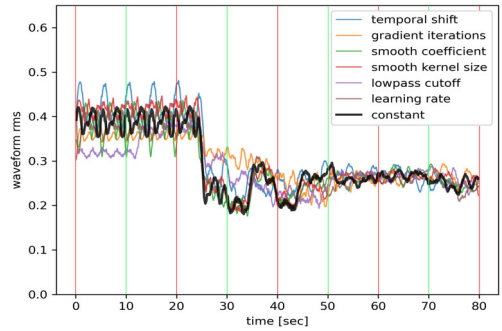

Figure 10: Model2 - DD Parameter Effects on Loudness.

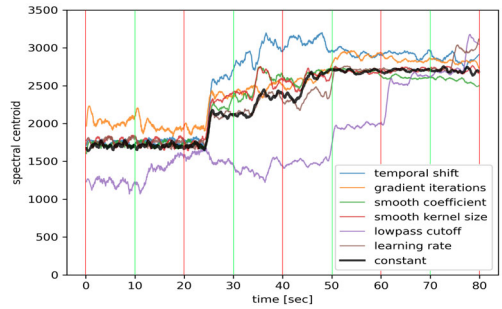

Figure 11: Model2 - DD Parameter Effects on Brightness.

The sounds that have been generated in Experiment 2 are available on SoundCloud[14]. For these sounds, the same acoustic features have been analysed as in Experiment 1. The graphs in figures 10, 11, 12, 13 depict how the mean value of a specific audio feature changes in re-

14 Experiment 2 - Audio examples generated by *Model2* trained on the *Events* dataset: soundcloud.com

sponse to different DD parameter variations and selections of layers and feature maps. The line colours in these graphs represent the parameter that were changed. The bold black line represents feature values that were obtained when keeping all DD parameters fixed and only selecting different layers and feature maps. In each of these plots, the time in seconds belongs to the x-axis and the audio feature value belongs to the y-axis. The red vertical lines indicate the moments of parameter change. The green vertical lines indicate moments of layer and feature map change.

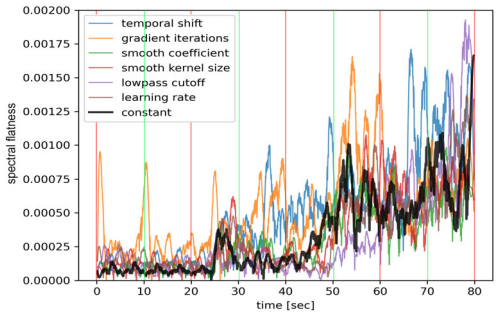

Figure 12: Model2 - DD Parameter Effects on Noisiness.

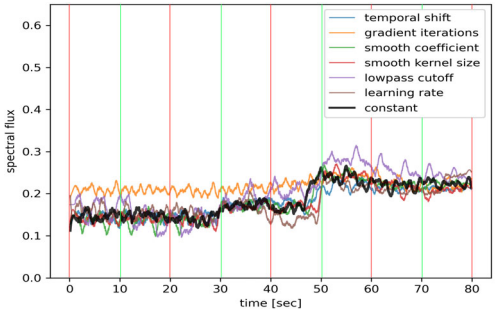

Figure 13: Model2 - DD Parameter Effects on Roughness.

This experiment shows that the influence of most DD parameters on the acoustic characteristics of the generated sounds is small. The gaussian smoothing coefficient and kernel size have no acoustic effect at all. The interaction count and learning rate exhibit a small acoustic effect. These two parameters exert their influence mainly by affecting the duration over which audio features change when a layer and/or feature map is switched. Other than that, an increase in iteration count also leads to a slight increase in *Roughness* if this feature is small otherwise, and a large learning rate slightly increases *Noisiness*, *Brightness* and *Roughness*. The randomised temporal offset mainly affects *Brightness* and *Noisiness* which both increase significantly for high offset sizes. The cut-off frequency of the low-pass filter has the strongest effect on *Brightness* which directly correlates with the cut-off frequency. The cut-off frequency also affects *Loudness* which is reduced for low frequencies. The acoustic properties that are affected the least by the cut-off frequency are *Noisiness* and *Roughness*. Both properties increase only slightly with increasing cut-off frequency.

5. Discussion

The quantitative and qualitative evaluations conducted as part of the two experiments proved extremely useful to gain a better understanding for the influence of the training data, network architecture, and DD parameter settings on the acoustic characteristics of the generated sounds.

It has been found that the choice of layer has by far the strongest influence on the acoustic characteristics of the generated sounds. This effect is particularly strong for the first two layers and supersedes the influence stemming the dataset or DD parameters. Choosing a feature map in the first layer always results in strongly pitched sounds with simple spectra, sustained loudness, and little noise. Choos-

ing a feature map in the second layer always leads to sounds whose loudness is concentrated in short bursts. From then on, feature maps in increasingly deeper layers tend to generated sounds that exhibit an increased complexity in terms of *Loudness* and spectral properties at the cost of an increase in noise. This tendency towards more acoustic complexity is counteracted by *Brightness*, whose variability across feature maps decreases significantly at higher layers.

The higher number of layers in *Model2* compared to *Model1* has no dramatic effect. Feature maps in layers 4 and 5 tends to generate sounds that have a slightly more complex and diverse *Loudness* and spectral dynamics than more shallow layers. Feature maps in layers 6 and 7 generate mostly the same acoustic results at layer 5. Accordingly, with the current choice of a layer architecture, an additional increase in the number of layers will likely not produce more diverse acoustic results.

It is a surprising and somewhat disappointing finding that the choice of audio dataset influences the characteristics of the generated audio only in nuances. No combination of feature map and DD parameter settings has generated sounds that clearly reproduce the characteristics of the audio material in the datasets. Nevertheless, the nuanced influences of the datasets are worth mentioning. In case of the *Events* dataset, the generated sounds are fairly varied with regards to the dynamics of their loudness and spectral properties, with the loudness and spectral dynamics being strongly correlated. In case of the *Instruments* dataset, the generated sounds exhibit a high loudness dynamics and low spectral dynamics, with both of them being strongly correlated. In case of the *Speech* dataset, the generated sounds exhibit a low loudness dynamics and high spectral dynamics, with both of them exhibiting little or no correlation.

Also small was the influence of most DD parameters on the generated sounds. The only exceptions are the size of the randomised temporal offset which affects *Brightness* and *Noisiness* and the cut-off frequency of the low pass filter which affects *Brightness* of the generated sounds. Due to a lack of acoustic impact of most DD parameters, the focus of attention should clearly be placed on the choice of feature maps when aiming for a large variety of generated sounds.

6. Composition

The findings described previously have helped with the creation of a music piece entitled "Analog Zombies in Deep Dreams"[15]. This piece will be released on CD Nr. 23 of the Deutsche Gesellschaft für ElektroakustischeMusik)[16]. The piece has been composed by the musician Thomas Wenk. For the composition, he recorded the mechanical and electrical noises produced by old cassette recorders. These recordings were assigned to one of three classes: isolated click sounds with low brightness and strong noise, isolated click sounds with a clear pitch and intermediate brightness, rapid click repetitions with high brightness and a clear pitch. A version of *Model1* with a smaller number of feature maps was trained to classify these recordings and subsequently used in combination with DD to generate new audio material. The generated sounds that were obtained in this manner vary in their acoustic characteristics between undifferentiated col-

15 Analog Zombies in Deep Dreams: www.e-wegner.net

16 Deutsche Gesellschaft für Elektroakustische Musik: www.degem.de

oured noise and complex rhythmical patterns.

The composer created a piece that combines the original recordings and generated sounds into a collage that highlights the causal connection between the two types of sounds. At the beginning of the piece, the original and generated sounds are juxtaposed in a manner that emphasises their differences. In this section, the original sounds are clearly recognisable as being produced by a technical apparatus whereas the generated sounds possess a more abstract sonic quality. Later on, the generated sounds dominate and draw the listeners' attention to the nuanced differences between the generated sounds. This section is occasionally interrupted by recordings of key-presses. The recordings appear as percussive punctuations that separate the generated sounds from each other.

7. Outlook

We plan to continue our work on musical applications of DD in both artistic and scientific directions.

For the creation of new artistic works, we plan to adopt the approach employed in Experiment 2, i.e. the generation of audio in real-time while simultaneously switching between convolution layers and feature maps. The new works could take the form of live music performances or interactive audio installations. As part of this artistic direction, we also intend to explore the acoustic effects of combining multiple feature maps at the same time. In parallel to this, we also intend to collaborate with addititonal composers who extensively work with large pre-recorded audio collections. We are curious to see what other strategies these composers might come up with when using DD to expand and enrich their musical vocabulary.

On the scientific side, our most immediate next step involves the conduction of additional systematic experiments that deal with varying aspects of DD that have so far been kept fixed. This includes initialising audio waveforms with other content than white noise and varying the number of classes when training models on an audio classification task. Other future work involves the design of model architectures that incorporate more sophisticated layers such as different variants of residual layers. Finally, and most importantly, we would like to follow up on the work conducted by Hermann [7] and abandon supervised training in favour of self-supervised approaches.

8. Acknowledgements

The research presented in this publication has been conducted in the context of the Horizon Europe project entitled Premiere[17] and supported by the European Union.

9. References

[1] Alexander Mordvintsev, Christopher Olah, and Mike Tyka. "Inceptionism: Going deeper into neural networks". In: (2015).

[2] LRA Al-Khazraji, Ayad R Abbas, and Abeer S Jamil. "A Systematic Review of Deep Dream". In: IRAQI J. Comput. Commun. Control Syst. Eng 23 (2023), pp. 192–209.

[3] Diego Ardila et al. "Audio deepdream: Optimizing raw audio with convolutional networks". In: Proceedings of the Interna-

17 Premiere - Performing arts in a new era: AI and XR tools for better understanding, preservation, enjoyment and accessibility: premiere-project.eu

tional Society for Music Information Retrieval Conference, New York, NY, USA. 2016, pp. 7–11.

[4] Saumitra Mishra, Bob L Sturm, and Simon Dixon. “Understanding a Deep Machine Listening Model Through Feature Inversion.” In: ISMIR. 2018, pp. 755–762.

[5] Federico Nicolás Cámara Halac and Matıas Delgadino. “DreamSound: Deep activation layer sonification”. In: Proceedings of the International Conference on Auditory Display. 2021.

[6] Manoj Plakal and Dan Ellis. YAMNet. 2021. URL: https ://github.com/tensorflow /models/tree/master/research/audioset/ yamnet.

[7] Vincent Herrmann, HJ Escalante, and R Hadsell. “Visualizing and sonifying how an artificial ear hears music.” In: NeurIPS (Competition and Demos). 2019, pp. 192–202.

[8] Prajit Ramachandran, Barret Zoph, and Quoc V Le. “Searching for activation functions”. In: arXiv preprint arXiv:1710.05941 (2017).