DreamSound: Deep Activation Layer Sonification

# Introduction

Deep learning (DL) in audio signal processing has received much attention in the last four years, and it is still a growing field.[1](#fn1) The Music Information Retrieval (MIR) community incorporated DL via large-scale audio datasets that came as early as 2015 (Engel et al. 2017, Gemmeke et al. 2017, Piczak 2015) and most MIR problems were outperformed by DL thenceforth.[2](#fn2)

There exists some work at the intersection of DL and sonification (Herrmann 2020, R. Michael et al. 2019), and more work in musical contexts applying DL to audio synthesis. Raw audio synthesis with DL has its origins in speech synthesis, first with the deep convolutional neural network WaveNet (Oord et al. 2016), subsequently with recurrent neural networks (RNN) (Kalchbrenner et al. 2018, Mehri et al. 2017). Their heavy computational requirements were a disadvantage for both training and performance. Further optimizations with parallel computations emerged (Hantrakul et al. 2019, Oord et al. 2017, Song et al. 2021, Yamamoto et al. 2020), but it was not until Generative Adversarial Networks (GANs) that music benefited from raw audio synthesized with DL (Bollepalli et al. 2017, Donahue et al. 2018, Kaneko et al. 2017, Liu et al. 2020, Pascual et al. 2017, Prenger et al. 2019, Tian et al. 2020).[3](#fn3)

These ground-breaking deep networks were tailored for short-length raw audio files of around one second, with a sample rate accurate enough for speech comprehension (16kHz). Longer or higher sample rate audio files were difficult to address due to high computation demands. A major breakthrough comes from the WaveNET-like autoencoder in (Engel et al. 2017), which was created as a new, data-driven synthesis technique for longer length audio files, extending GANs towards musical application. Several hybrid synthesizer models have then been trained (McCarthy & Ahmed 2020), and the first GAN synthesizers appeared: GANSynth (Engel et al. 2019), trained on the musical instrument sound dataset NSynth (Engel et al. 2017), and EnvGAN (Madhu & K 2021), trained on the environmental sound generation ESC dataset (Piczak 2015). In sum, the evolution of audio synthesis has been fruitful, to the point that Google’s Magenta Lab has developed tools and plugins accessible to a wider musical audience (Roberts et al. 2019).

Historically, most DL development appeared first on images and then on sound, namely because of the availability of large-scale datasets, namely, ImageNet (Russakovsky et al. 2015). The literature reflects that one inspiration for sonic GANs came from its deep convolutional version DCGAN (Radford et al. 2015), an adversarial image generator. The problem of translating networks from images to sounds is mentioned in (Rothman) and (Purwins et al. 2019), and an interesting discussion can be read in (Briot et al. 2017). Of interest here is Deep Dream (DeepDream 2015), an image generating architecture using layer *activation maximization*[4](#fn4) of a deep pre-trained model for image classification (Szegedy et al. 2014).

In the present paper, we present DreamSound, a creative adaptation of (DeepDream 2015) to sound using *activation maximization* as well as *timbre style transfer*. Audio examples can be found in the code repository.[5](#fn5) DreamSound advances work on DL and audio synthesis, because it is aimed at sonifying YAMNet layers (Plakal & Ellis 2020), a novel network previously trained on sound. Further, DreamSound proposes several creative approaches in relation to layer *activation maximization*. In section 2, we present our approach as input manipulation and in section 3 as a sonification design. Both sections discuss our implementation and findings in relation to previous work, as well and the limitations we have found. We conclude in section 4 that there is much work to be done in both the adaptation of image to audio algorithms and in the navigable feature space of deep models.

The research in DreamSound can be understood from two approaches: *input manipulation* and *sonification design*. In turn, these approaches are described with examples of previous work and our implementation is briefly discussed.

# Previous Work

## Deep Dream

Deep Dream is a project described in a blog post (DeepDream 2015) and a tutorial (DeepDream). It is geared towards image generation using deep layer *activation maximization*, on a deep network (Szegedy et al. 2014) that was trained for image classification with the ImageNet (Russakovsky et al. 2015) dataset. At the heart of the algorithm, a *gradient ascent* (see [3.2](#subsec:gradients)) takes place between a *loss* and an input. Loss, in this context, refers to the activations of a layer given some input. The *gradient* is said to ascend because there is incremental manipulation on the input to a next iteration of the same routine. Thus, in time, the input is ‘steered’ in an additive way towards a particular *activation* or class of the model’s class space. Briot et al (Briot et al. 2017) refer to this type of increments *input manipulation*, since the manipulation occurs at the input, not the output of the architecture. In DreamSound, there is an incremental manipulation of an initially random or specific sound content towards a targeted feature space.

The output of Deep Dream can be understood as a psychedelic image generator that creates the *pareidolia* effect, i.e., the psychological phenomenon in which the mind, in response to a stimulus, perceives similar patterns where none exist (Briot et al. 2017). Essentially, the effect itself directly depends on the activations of a layer within a deep neural network. Thus, the combination of the neural background and the psychedelic aspect gave way for the *dream* in the name, raising further questions about the nature of artificial networks and their otherness that extends the limits of this paper.

## Deep Dream in Sound

There is existing work in the application of the Deep Dream effect (DeepDream 2015) to sound. In (Dittmar & Balke 2015), Balke and Dittmar use a magnitude spectrogram as the color channels of an image (RGB) and apply *gradient ascent* between the spectrogram and a deep network trained on images. Subsequently, the output images were resynthesized into sound with Griffin and Lim’s method.[6](#fn6) Stamenovic adapted a similar approach in the python library tensorflow (Stamenovic 2016). Finally, an important antecedent that steers away from Deep Dream is Herrmann’s recent work in the visualization and sonification of neural networks (Herrmann 2020). In contrast, Herrmann uses a network previously trained on sound to sonify the features that activate a selected layer yielding impressive results. His research further proves that while lower layers focalize on input localities, higher ones produce more upper-level sonic activity, e.g., rhythmic or harmonic patterns. In this sense, the results obtained from sonifying layer activations are informative on both input and excitations of a layer.

(a) Gradients (10 steps)

(a) “Speech”

## Images as Sound

The creative potential of these projects lives in their experimental approach towards spectral transformation. In all these cases, the authors perform layer *activation maximization* on a deep network trained on images, taking the gradient between the activations triggered by an image and the input. The input, in this case, is the spectrogram of a sound, either magnitude (Dittmar & Balke 2015, Stamenovic 2016) or the scaleogram (constant-Q transform) (Herrmann 2020). Therefore, some issues arise due to the usage of images as sound, namely the sequential aspect of audio. For example, Purwins et al (Purwins et al. 2019) note the difference between raw audio (one-dimensional time series signal) and two-dimensional images, and thus claim that audio signals have to be studied sequentially and audio-specific solutions need to be addressed. More specifically, Briot et al (Briot et al. 2017) borrow from crystallography the term *anisotropy* to describe one of the central difficulties when transcoding architectures dealing with images to sounds within spectrogram representations. *Anisotropy* means direction dependence. While the spatial correlation between pixels maintains independently of their direction of an image (*isotropy*), this is not the case when dealing with spectrogram images. In a spectrogram representation the horizontal axis represents one thing (e.g. time) and the vertical, another (e.g. frequency). Therefore, significant information is lost when translating algorithms from isotropic to anisotropic contexts. In DreamSound, we implemented an audio-specific solution using raw audio, with variations on an *activation maximization* function that performs filtering to further adjust the output.

# Design

While visualization has been the main channel to understand and perceptualize deep networks (Simonyan et al. 2014), there exist work on their sonification like Herman’s work mentioned above. *Feature inversion* is a technique that has been a proposed solution for understanding these architectures (Dosovitskiy & Brox 2016, Mahendran & Vedaldi 2014), achieving important advancement in the interpretation of Machine Listening models. In (Mishra et al. 2018), Mishra et al proved that temporal and harmonic structures are preserved in deep convolutional layers. Winters et al (R. Michael et al. 2019) advanced scientific and participant study-based work by sonifying the penultimate layer of a deep convolutional neural network previously trained for image classification. An interesting aspect of their results is the concept of a *sonification layer* that can further reduce the signal-to-noise ratio within a deep network.

Following the line of these examples, DreamSound can be understood as artistic sonification geared to output sounds using a deep convolutional neural network in an intrinsically coherent way. Three design approaches are mentioned, defined by three creative implementations of an *activation maximization* function. The examples here were made with a single-class python package implementation of DreamSound, available for import via pip install dreamsound.[7](#fn7). The full details of the implementation can be expanded in a more verbose documentation and extend the limits of this paper.

## The YAMNet Model

While there are many available trained models for sound classification, we have chosen a general sound source classifier YAMNet (Plakal & Ellis 2020). YAMNet is a pretrained deep neural network classifier that uses MobileNets (Howard et al. 2017), a depthwise-separable convolution architecture with a final activation layer that contains most relevant features for classification. The model was previously trained with 521 audio classes based on the Audio Set corpus (Gemmeke et al. 2017), which are mostly non-musical, short fragments.

## Sonifying the gradients

In most machine learning models, *gradient descent* is the process by which the loss, that is, a function describing the activations of a layer, is minimized with respect to the input. The inverse happens in *gradient ascent*: the loss is maximized so that the input image increasingly excites the layers (DeepDream), arriving at *activation maximization*. The ‘gradients’ are referred to here as the gradient vector between a loss and its inputs in a model. We have found similarities in our sonifications of the gradients and in Herrmann’s sonification of the activations (Herrmann 2020), which sound like filtered noise with dynamic spectral envelopes. Given our choice of the last layer of YAMNet, we find, like Herrmann, that the dynamic aspect of these envelopes resembles the rhythmic elements of the input sound. Visually, the spectrogram of the gradients looks like the inverted image of the spectrogram of the original sound, see Figure [[fig:melspecs]](#fig:melspecs) (a).

(a) DeepDream Activation

(b) Hard-cut Filter

(c) Hard-cut Filter Add

(d) Targeted Filter

## Deep Dream function

Following the Deep Dream function, the gradients obtained between the loss and the input are added with some attenuation to the input and fed back to the loop, as can be seen in Figure [[fig:maxfun]](#fig:maxfun) (a). When maximizing the activation of a certain layer, the result is said to point to a direction in the class space of the classifier. In this case, the *activation maximization* is ‘steered’ in the *same* direction of the original sound’s class. In order to steer towards a different class of the model, we use a target. Therefore, the sound would sound more like itself or what the model considers its class to be.

However, while this function gives interesting results in the image domain, in the sound domain the additive aspect takes precedence. That is to say, the original sound and the gradients (Figure [[fig:melspecs]](#fig:melspecs) b.3) are perceived as two superimposed independent layers. We recommend listening to the audio files for a better idea.

## Filter-based function

X\_mag, X\_pha = MAGPHASE(STFT(x))  
Y\_mag = ABS(STFT(y))  
Y\_mag\_offset = NORMALIZE(Y\_mag) - threshold  
# hard cut based on sign  
hard\_cut = (SIGN(Y\_mag\_offset)+1) / 2  
# apply the hard-cut filter  
X\_mag\_cut = hard\_cut \* X\_mag  
# rephase  
X\_mag\_rephased = COMPLEX\_MUL(X\_pha, X\_mag\_cut)  
# inverse stft  
x\_new = ISTFT(X\_mag\_rephased)  
# add  
output = x\_new \* step\_size + y  
return output

[function]

Therefore, a hard-cut filter (Figure [[fig:melspecs]](#fig:melspecs) b.3) was designed to remove magnitudes that fall below a defined threshold and to let the remaining magnitudes through. Thus, by constructing this filter with the gradients and applying it to the original sound, the regions of the gradients where energy is present will let the original sound pass to the following iteration. In other words, the original sound is convolved with the gradients, and the order of the inputs for the convolution can be easily flipped to construct the filter with the original sound and convolve the gradients with it [[fig:maxfun]](#fig:maxfun) (b).

The use of this function yielded more interesting sonic outcomes than the Deep Dream function, as can be seen in Figure [[fig:melspecs]](#fig:melspecs) (b.1). The original sound and its gradients at that step seem to merge better than with the Deep Dream function. The difference between the original sound and the output can be seen in Figure [[fig:melspecs]](#fig:melspecs) (b.2), and it evidences the elements that were removed from the original sound. Better results are achieved by an extension to this design (see [[function]](#function)). The additive element of (a) occurs between the output of the filter and the original sound, which only then is fed back into the loop, shown in Figure [[fig:maxfun]](#fig:maxfun) (c). The audio examples using this function seem to be more expressive.

## Target-based function

In Figure [[fig:maxfun]](#fig:maxfun) (d), however, the filter is constructed with a new sound that we call ‘target.’ This technique is a creative interpretation of *style transfer*, that was originally designed for images in (Gatys et al. 2015). Gatys et al used a deep network to obtain content information (i.e., features) from one image and style (or feature correlations) from another. The notion of style represented then the style of an artist. In the case of audio transfer, style does not refer to musical style, but to timbre. There are other examples dealing with *timbre style transfer* (Foote et al. 2016, Ulyanov & Lebedev 2016, Wyse 2017), which are dealt with in (Briot et al. 2017), but in general, a gradient is used on an input and a target in order to synthesize a third, hybrid sound. In (Verma & Smith 2018), Verma and Smith treat audio synthesis as a *style transfer* problem, and they use back-propagation to optimize the sound to conform to filter-output. With our target-based function, the gradients of the layer are convolved to the filter constructed by the target sound and then fed back into the loop.

# Conclusions and Future Work

We have presented a prototype adaptation of the Deep Dream project into sound using Tensorflow 2 and YAMNet. Our project began as a rapid and remote collaborative work within the Google Colab environment, and it developed into a python package. We have contributed some research at the intersection of deep learning and audio, and presented our results. The most interesting results occur with a filter-based *activation maximization* function built from the gradients obtained from an input and its loss. Future work consists of (1) adapting DreamSound to intake different length YAMNet layers, as well as other models. (2) Further explorations with an adversarial model such as Kwgan is expected.[8](#fn8) An interesting use of DreamSound is to further evaluate the classification capacity of YAMNet. Therefore, we intend to (3) use it as a data augmentation tool for this and other models. Finally, since DreamSound is a sound generator, we intend to (4) play it as a musical instrument in an artistic work.

Bollepalli B, Juvela L, Alku P. 2017. Generative adversarial network-based glottal waveform model for statistical parametric speech synthesis. *Interspeech 2017*

Briot J, Silva Chear AC da, Manzelli R, Thakkar V, Siahkamari A, et al. 2018. An end to end model for automatic music generation: Combining deep raw and symbolic audio networks

Briot J-P, Hadjeres G, Pachet F. 2017. Deep learning techniques for music generation - A survey. *CoRR*. abs/1709.01620:

Chandna P, Blaauw M, Bonada J, Gomez E. 2019. WGANSing: A multi-voice singing voice synthesizer based on the wasserstein-GAN. *2019 27th European Signal Processing Conference (EUSIPCO)*

Choi K, Fazekas G, Cho K, Sandler M. 2017. A tutorial on deep learning for music information retrieval

Deep learning for music (DL4M). 2017

Deep learning with audio. 2018

DeepDream

DeepDream. 2015

Dittmar C, Balke S. 2015. DeepDreamEffect

Donahue C, McAuley J, Puckette M. 2018. Adversarial audio synthesis

Dong H-W, Yang Y-H. 2018. Convolutional Generative Adversarial Networks with Binary Neurons for Polyphonic Music Generation. *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pp. 190–96. Paris, France: ISMIR

Dosovitskiy A, Brox T. 2016. Inverting visual representations with convolutional networks

Engel J, Agrawal KK, Chen S, Gulrajani I, Donahue C, Roberts A. 2019. GANSynth: Adversarial neural audio synthesis

Engel J, Resnick C, Roberts A, Dieleman S, Eck D, et al. 2017. Neural audio synthesis of musical notes with WaveNet autoencoders

Foote D, Yang D, Rohaninejad M. 2016. Do androids dream of electric beats?

Gatys LA, Ecker AS, Bethge M. 2015. A neural algorithm of artistic style. *CoRR*. abs/1508.06576:

Gemmeke JF, Ellis DPW, Freedman D, Jansen A, Lawrence W, et al. 2017. Audio set: An ontology and human-labeled dataset for audio events. *Proc. IEEE ICASSP 2017*

Griffin D, Lim J. 1983. Signal estimation from modified short-time fourier transform. *ICASSP ’83. IEEE International Conference on Acoustics, Speech, and Signal Processing*. 8:804–7

Hantrakul L, Engel J, Roberts A, Gu C, Hantrakul L. 2019. Fast and flexible neural audio synthesis. *Proceedings of the 20th International Society for Music Information Retrieval Conference*, pp. 524–30. Delft, The Netherlands: ISMIR

Herremans D, Chuan C-H. 2017. Proceedings of the first international workshop on deep learning and music

Herrmann V. 2020. Visualizing and sonifying how an artificial ear hears music. *Proceedings of the NeurIPS 2019 Competition and Demonstration Track*. 123:192–202

Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, et al. 2017. MobileNets: Efficient convolutional neural networks for mobile vision applications

Jang W, Lim D, Yoon J. 2021. Universal MelGAN: A robust neural vocoder for high-fidelity waveform generation in multiple domains

Kalchbrenner N, Elsen E, Simonyan K, Noury S, Casagrande N, et al. 2018. Efficient neural audio synthesis

Kaneko T, Kameoka H, Hojo N, Ijima Y, Hiramatsu K, Kashino K. 2017. Generative adversarial network-based postfilter for statistical parametric speech synthesis. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4910–14

Kumar K, Kumar R, Boissiere T de, Gestin L, Teoh WZ, et al. 2019. MelGAN: Generative adversarial networks for conditional waveform synthesis. *Advances in Neural Information Processing Systems*. 32: <https://proceedings.neurips.cc/paper/2019/file/6804c9bca0a615bdb9374d00a9fcba59-Paper.pdf>

Lee CY, Toffy A, Jung GJ, Han W-J. 2018. Conditional WaveGAN. *CoRR*. abs/1809.10636:

Liu J-Y, Chen Y-H, Yeh Y-C, Yang Y-H. 2020. Unconditional audio generation with generative adversarial networks and cycle regularization. *Interspeech 2020*

Madhu A, K S. 2021. EnvGAN: Adversarial synthesis of environmental sounds for data augmentation

Mahendran A, Vedaldi A. 2014. Understanding deep image representations by inverting them

Manzelli R, Thakkar V, Siahkamari A, Kulis B. 2018. Conditioning Deep Generative Raw Audio Models for Structured Automatic Music. *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pp. 182–89. Paris, France: ISMIR

McCarthy O, Ahmed Z. 2020. HooliGAN: Robust, high quality neural vocoding

Mehri S, Kumar K, Gulrajani I, Kumar R, Jain S, et al. 2017. SampleRNN: An unconditional end-to-end neural audio generation model

Mishra S, Sturm BL, Dixon S. 2018. Understanding a Deep Machine Listening Model Through Feature Inversion. *Proceedings of the 19th International Society for Music Information Retrieval Conference*, pp. 755–62. Paris, France: ISMIR

Mittal G, Engel J, Hawthorne C, Simon I. 2021. Symbolic music generation with diffusion models

MuseGAN papers

Mustafa A, Pia N, Fuchs G. 2021. StyleMelGAN: An efficient high-fidelity adversarial vocoder with temporal adaptive normalization

Nistal J, Lattner S, Richard G. 2020. DrumGAN: Synthesis of drum sounds with timbral feature conditioning using generative adversarial networks. *Proceedings of the 21st International Society for Music Information Retrieval Conference*, pp. 590–97. Montreal, Canada: ISMIR

Oord A van den, Dieleman S, Zen H, Simonyan K, Vinyals O, et al. 2016. WaveNet: A generative model for raw audio

Oord A van den, Li Y, Babuschkin I, Simonyan K, Vinyals O, et al. 2017. Parallel WaveNet: Fast high-fidelity speech synthesis

Oramas S, Nieto O, Barbieri F, Serra X. 2017. Multi-Label Music Genre Classification from Audio, Text and Images Using Deep Features. *Proceedings of the 18th International Society for Music Information Retrieval Conference*, pp. 23–30. Suzhou, China: ISMIR

Pascual S, Bonafonte A, Serrà J. 2017. SEGAN: Speech enhancement generative adversarial network

Piczak KJ. 2015. ESC: Dataset for environmental sound classification. *Proceedings of the 23rd ACM International Conference on Multimedia*, pp. 1015–18. New York, NY, USA: Association for Computing Machinery

Plakal M, Ellis D. 2020. YAMNet

Prenger R, Valle R, Catanzaro B. 2019. Waveglow: A flow-based generative network for speech synthesis. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3617–21

Purwins H, Li B, Virtanen T, Schlüter J, Chang S-Y, Sainath T. 2019. Deep learning for audio signal processing. *IEEE Journal of Selected Topics in Signal Processing*. 13(2):206–19

R. Michael W, Kalra A, Walker BN. 2019. Hearing artificial intelligence: Sonification guidelines and results from a case-study in melanoma diagnosis

Radford A, Metz L, Chintala S. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks

Roberts A, Engel J, Mann Y, Gillick J, Kayacik C, et al. 2019. Magenta Studio: Augmenting Creativity with Deep Learning in Ableton Live. *Proceedings of the 6th International Workshop on Musical Metacreation*, p. 7. Charlotte, United States: MUME

Rothman D. What’s wrong with CNNs and spectrograms for audio processing?

Russakovsky O, Deng J, Su H, Krause J, Satheesh S, et al. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*. 115(3):211–52

Simonyan K, Vedaldi A, Zisserman A. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps

Song E, Yamamoto R, Hwang M-J, Kim J-S, Kwon O, Kim J-M. 2021. Improved parallel WaveGAN vocoder with perceptually weighted spectrogram loss

Stamenovic M. 2016. Deep dreaming on audio spectrograms with tensorflow

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, et al. 2014. Going deeper with convolutions

Tian Q, Chen Y, Zhang Z, Lu H, Chen L, et al. 2020. TFGAN: Time and frequency domain based generative adversarial network for high-fidelity speech synthesis

Ulyanov D, Lebedev V. 2016. Audio texture synthesis and style transfer

Verma H, Thickstun J. 2019. Convolutional composer classification. *Proceedings of the 20th International Society for Music Information Retrieval Conference*, pp. 549–56. Delft, The Netherlands: ISMIR

Verma P, Smith JO. 2018. Neural style transfer for audio spectograms

Wyse L. 2017. Audio spectrogram representations for processing with convolutional neural networks. *Proceedings of the 1st International Workshop on Deep Learning for Music*, pp. 37–41

Yamamoto R, Song E, Kim J-M. 2020. Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram

Yang L-C, Chou S-Y, Yang Y-H. 2017. MidiNet: A convolutional generative adversarial network for symbolic-domain music generation

1. See (Purwins et al. 2019) for a complete review, (Briot et al. 2017) for a published book, as well as(Herremans & Chuan 2017) for the first dedicated workshop, and (Choi et al. 2017) and the repositories for DL4M (Deep learning for music (DL4M) 2017) and the Aalto courses (Deep learning with audio 2018) for introductory tutorials. For the use of DL for symbolic music representations, see (Briot et al. 2018, Dong & Yang 2018, Manzelli et al. 2018, Mittal et al. 2021, MuseGAN papers, Yang et al. 2017)[↩︎](#fnref1)

1. Despite the usefulness of DL in raw audio, e.g. reducing the gap between symbolic and audio classification (Oramas et al. 2017), it can be argued that not all MIR problems benefit from it (Verma & Thickstun 2019).[↩︎](#fnref2)

1. Multiple variants to the original WaveGAN have appeared in the literature. See for example: conditional GAN (Lee et al. 2018), MelGAN (Jang et al. 2021, Kumar et al. 2019), WGANSing (Chandna et al. 2019), TFGAN (Tian et al. 2020), DRUMGAN (Nistal et al. 2020), SyleMelGAN (Mustafa et al. 2021), among others. In our previous work Kwgan (<https://github.com/fdch/kwgan>), we extended WaveGan with conditionals to tailor an artistically relevant context.[↩︎](#fnref3)

1. Activation maximization is a process that returns the inputs that with most confidence would cause a certain output.[↩︎](#fnref4)

1. <https://github.com/fdch/dreamsound>[↩︎](#fnref5)

1. For the Griffin and Lim’s method, see (Griffin & Lim 1983)[↩︎](#fnref6)

1. <https://pypi.org/project/dreamsound>[↩︎](#fnref7)

1. <https://github.com/fdch/kwgan>[↩︎](#fnref8)