DREAMSOUND: DEEP ACTIVATION LAYER SONIFICATION

Federico Nicolás Cámara Halac

Ohio State University
1813 North High Street
Columbus, Ohio
camarahalac.1@osu.edu

Matías Delgadino

Oxford University
Woodstock Road
Oxford, OX2 6GG
matias.delgadino@maths.ox.ac.uk

ABSTRACT

Deep learning (DL) in audio signal processing has received much attention in the last four years, and it is still a growing field. Some attempts were made to translate deep networks from image to audio applications, and we discuss some of the issues that arise. In the present paper, we present DreamSound, a creative adaptation of Deep Dream to sound addressed from two approaches: *input manipulation*, and *sonification design*. Our chosen model is YAMNet, a pre-trained deep network for sound classification. We present the original Deep Dream *activation maximization* function in relation to three filter-based creative variations that were implemented. We find that more interesting results are achieved with this filter-based approach, but that there is still room for experimentation.

1. 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] The Music Information Retrieval (MIR) community incorporated DL via large-scale audio datasets that came as early as 2015 [**2015piczak**, **2017audioset**, **engel2017neural**] and most MIR problems were outperformed by DL thenceforth.[2]

There exists some work at the intersection of DL and sonification [**Winters2019**, **pmlr-v123-herrmann20a**], 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 [**oord2016wavenet**], subsequently with recurrent neural networks (RNN) [**mehri2017samplernn**, **kalchbrenner2018efficient**]. Their heavy computational requirements were a disadvantage for both training and performance. Further optimizations with parallel computations emerged [**oord2017parallel**, **lamtharn˙hantrakul˙2019˙3527860**, **yamamoto2020parallel**, **song2021improved**], but it was not until Generative Adversarial Networks (GANs) that music benefited from raw audio synthesized with DL [**Bollepalli˙2017**, **2017Kaneko**, **pascual2017segan**, **donahue2018adversarial**, **2019waveglow**, **tian2020tfgan**, **Liu˙2020**].[3]

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 [**engel2017neural**], 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 [**mccarthy2020hooligan**], and the first GAN synthesizers appeared: GANSynth [**engel2019gansynth**], trained on the musical instrument sound dataset NSynth [**engel2017neural**], and EnvGAN [**madhu2021envgan**], trained on the environmental sound generation ESC dataset [**2015piczak**]. 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 [**adam˙roberts˙2019˙4285266**].

Historically, most DL development appeared first on images and then on sound, namely because of the availability of large-scale datasets, namely, ImageNet [**ILSVRC15**]. The literature reflects that one inspiration for sonic GANs came from its deep convolutional version DCGAN [**radford2015unsupervised**], an adversarial image generator. The problem of translating networks from images to sounds is mentioned in [**RothmanBlog**] and [**2019Purwins**], and an interesting discussion can be read in [**Briot2017**]. Of interest here is Deep Dream [**Mordvintsev2015**], an image generating architecture using layer *activation maximization*[4] of a deep pre-trained model for image classification [**szegedy2014going**].

[1] See [**2019Purwins**] for a complete review, [**Briot2017**] for a published book, as well as [**herremans2017proceedings**] for the first dedicated workshop, and [**choi2017tutorial**] and the repositories for DL4M [**Bayle2017**] and the Aalto courses [**Koray2018**] for introductory tutorials. For the use of DL for symbolic music representations, see [**yang2017midinet**, **Briot2018AnET**, **rachel˙manzelli˙2018˙1492375**, **hao˙wen˙dong˙2018˙1492377**, **mittal2021symbolic**, **MuseGanPapers**]

[2] Despite the usefulness of DL in raw audio, e.g. reducing the gap between symbolic and audio classification [**sergio˙oramas˙2017˙1417427**], it can be argued that not all MIR problems benefit from it [**harsh˙verma˙2019˙3527866**].

[3] Multiple variants to the original WaveGAN have appeared in the literature. See for example: conditional GAN [**2018Lee**], MelGAN [**NEURIPS2019˙6804c9bc**, **jang2021universal**], WGANSing [**Chandna˙2019**], TFGAN [**tian2020tfgan**], DRUMGAN [**javier˙nistal˙2020˙4245504**], SyleMelGAN [**mustafa2021stylemelgan**], among others. In our previous work Kwgan (https://github.com/fdch/kwgan), we extended WaveGan with conditionals to tailor an artistically relevant context.

[4] Activation maximization is a process that returns the inputs that with most confidence would cause a certain output.

In the present paper, we present DreamSound, a creative adaptation of [**Mordvintsev2015**] to sound using *activation maximization* as well as *timbre style transfer*. Audio examples can be found in the code repository.[5] DreamSound advances work on DL and audio synthesis, because it is aimed at sonifying YAMNet layers [**YamNet2020**], 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.

2. PREVIOUS WORK

2.1. Deep Dream

Deep Dream is a project described in a blog post [**Mordvintsev2015**] and a tutorial [**DeepDreamTutorial**]. It is geared towards image generation using deep layer *activation maximization*, on a deep network [**szegedy2014going**] that was trained for image classification with the ImageNet [**ILSVRC15**] dataset. At the heart of the algorithm, a *gradient ascent* (see 3.2) 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 [**Briot2017**] 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.

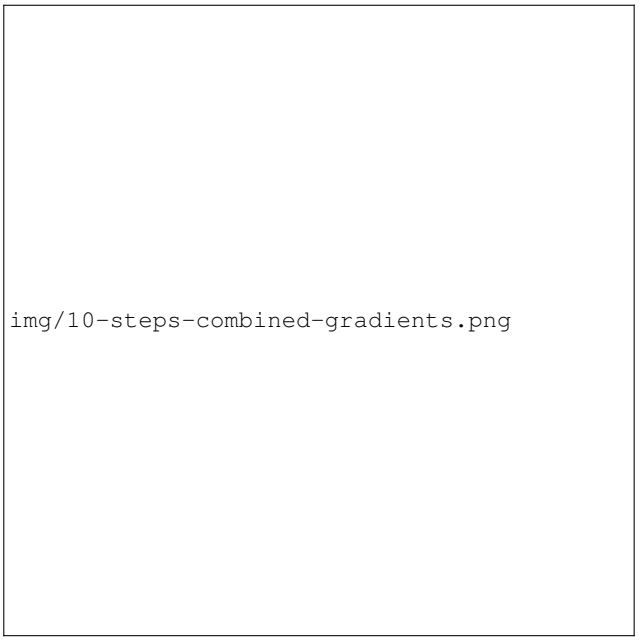
The output of Deep Dream can be understood as a psychedelic image generator that creates the *pareidolia* effect, i.e., the psychological phenomenon in which the mind, in response to a stimulus, perceives similar patterns where none exist [**Briot2017**]. Essentially, the effect itself directly depends 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.

2.2. Deep Dream in Sound

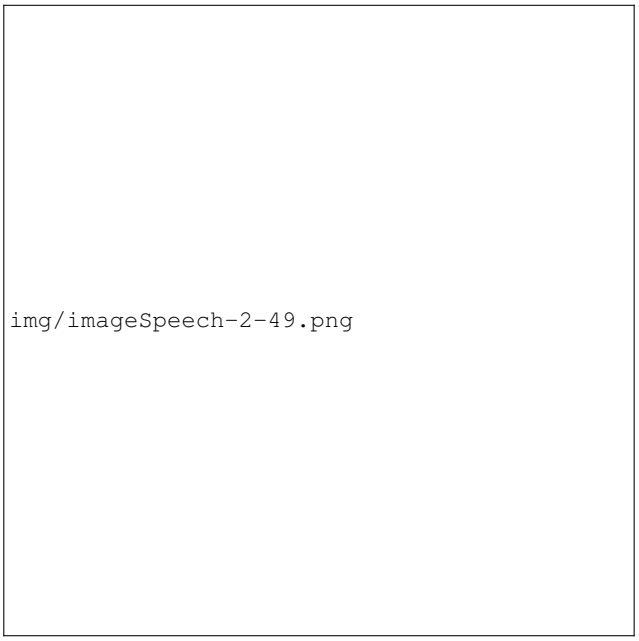
There is existing work in the application of the Deep Dream effect [**Mordvintsev2015**] to sound. In [**Balke2015**], Balke and Dittmar use a magnitude spectrogram as the color channels of an image (RGB) and apply *gradient ascent* between the spectrogram and a deep network trained on images. Subsequently, the output images were resynthesized into sound with Griffin and Lim's method.[6] Stamenovic adapted a similar approach in the python library tensorflow [**Stamenovic2016**]. Finally, an important antecedent that steers away from Deep Dream is Herrmann's recent work in the visualization and sonification of neural networks [**pmlr-v123-herrmann20a**]. 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.

[5] `https://github.com/fdch/dreamsound`

[6] For the Griffin and Lim's method, see [**Lim1983**]



(a) Gradients (10 steps)



(a) "Speech"

Figure 1: Spectrograms of some examples. (a) Gradients output during a first after 10 steps. This was done with an initial implementation of DreamSound. (b) The Mel-scaled spectrograms displayed here belong to the 49th iteration, and are in order: (1) the output of DreamSound, (2) the difference between the original sound and the output, (3) the gradients, and (4) the inverse STFT of the phase-reconstructed hard-cut filter. Audio representations for all examples are available in the the code repository.

2.3. 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 [**Balke2015**, **Stamenovic2016**] or the scaleogram (constant-Q transform) [**pmlr-v123-herrmann20a**]. Therefore, some issues arise due to the usage of images as sound, namely the sequential aspect of audio. For example, Purwins et al [**2019Purwins**] 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 [**Briot2017**] 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.

3. DESIGN

While visualization has been the main channel to understand and perceptualize deep networks [**simonyan2014deep**], 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 [**mahendran2014understanding**, **dosovitskiy2016inverting**], achieving important advancement in the interpretation of Machine Listening models. In [**saumitra˙mishra˙2018˙1492527**], Mishra et al proved that temporal and harmonic structures are preserved in deep convolutional layers. Winters et al [**Winters2019**] 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]. The full details of the implementation can be expanded in a more verbose documentation and extend the limits of this paper.

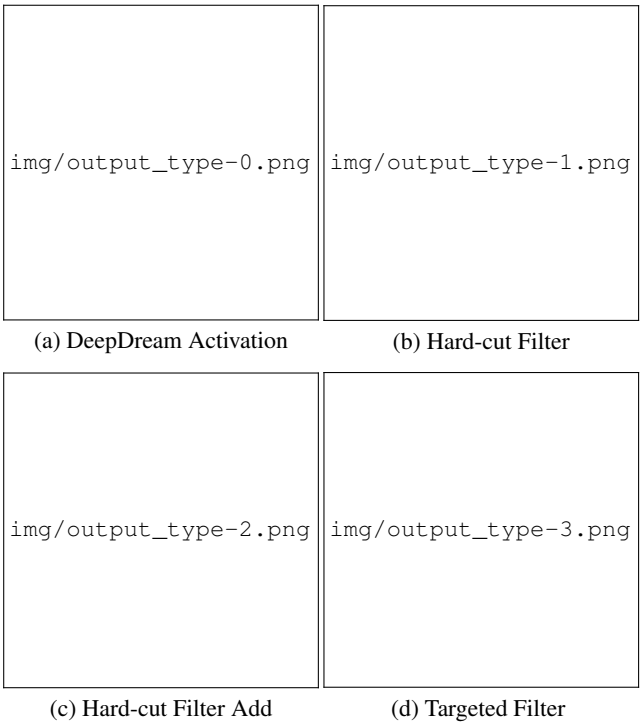
[7] `https://pypi.org/project/dreamsound`

3.1. The YAMNet Model

While there are many available trained models for sound classification, we have chosen a general sound source classifier YAMNet [**YamNet2020**]. YAMNet is a pretrained deep neural network classifier that uses MobileNets [**howard2017mobilenets**], 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 [**2017audioset**], which are mostly non-musical, short fragments.

3.2. 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 [**DeepDreamTutorial**], 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 [**pmlr-v123-herrmann20a**], 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 1 (a).



(a) DeepDream Activation (b) Hard-cut Filter

(c) Hard-cut Filter Add (d) Targeted Filter

Figure 2: Activation Maximization Function types. The first type is directly adapted from DeepDream (a). From (b) to (d), a hard-cut filter is constructed from the magnitude spectra of one of the inputs, so that in (b) the gradients are filtered (convolved) by the original sound, and in (c) the output of this convolution is attenuated and added to the original sound. The difference in (d) comes with the use of another audio used as 'target' to construct the filters.

3.3. 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 2 (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 1 b.3) are perceived as two superimposed independent layers. We recommend listening to the audio files for a better idea.

3.4. Filter-based function

Listing 1: Filter-based function with inputs x and y

```
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
```

Therefore, a hard-cut filter (Figure 1 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 2 (b).

The use of this function yielded more interesting sonic outcomes than the Deep Dream function, as can be seen in Figure 1 (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 1 (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 3.4). 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 2 (c). The audio examples using this function seem to be more expressive.

3.5. Target-based function

In Figure 2 (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 [**GatysEB15a**]. 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* [**Foote2016**, **Ulyanov2016**, **Wyse2017**], which are dealt with in [**Briot2017**], but in general, a gradient is used on an input and a target in order to synthesize a third, hybrid sound. In [**verma2018neural**], 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.

4. 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] 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.

dreamsound

[8] https://github.com/fdch/kwgan