DREAMSOUND: DEEP ACTIVATION LAYER SONIFICATION

Federico Nicolás Cámara Halac

Ohio State University
1813 North High Street
Columbus, Ohio

Matías Delgadino

Oxford University
Woodstock Road
Oxford, OX2 6GG

ABSTRACT

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 [**?**, **?**, **?**] and most MIR problems were outperformed by DL thenceforth.[2]

There exists some work at the intersection of DL and sonification [**?**], and more work in musical contexts applying DL to audio synthesis. Raw audio synthesis with DL has its origins on speech synthesis, first with the deep convolutional neural network WaveNet [**?**], subsequently with recurrent neural networks (RNN) [**?**, **?**]. Their heavy computational requirements were a disadvantage for both training and performance. Further optimizations with parallel computations emerged [**?**, **?**, **?**, **?**], but it was not until Generative Adversarial Networks (GANs) that music benefited from raw audio synthesized with DL [**?**, **?**, **?**, **?**, **?**, **?**, **?**].[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 samplerate audio files were difficult to address due to high computation demands. A major breakthrough from [**?**] and [**?**] comes with the WaveNET-like autoencoder in [**?**], which 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 [**?**], and the first GAN synthesizers appeared: GANSynth [**?**], trained on the musical instrument sound dataset NSynth [**?**], and EnvGAN [**?**], trained on the environmental sound generation ESC dataset [**?**]. 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 [**?**].

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

In the present paper, we present DreamSound, a creative adaptation of [**?**] to sound using *timbre style transfer* as well as *activation maximization*. 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 [**?**], a novel network previously trained on sound. Further, DreamSound proposes several creative approaches in relation to layer *activation maximization*. In section 2, we present approach in relation to previous work, as well and the limitations we have found. In section 3, we expose our methods and techniques. In section 4, we present our results and briefly discuss them in context. We conclude in section 5 that there is much work to be done on the adaptation of image to audio concepts and in the navigable feature space of deep models.

2. APPROACH

The research in DreamSound can be understood at the intersection of three problems: *input manipulation*, *timbre style transfer*, and *sonification design*. In turn, these problems are described with examples of previous work and their solutions in the present work are presented.

2.1. Input Manipulation

2.1.1. Deep Dream

Deep Dream is a project described in a blog post [**?**] and a tutorial [**?**]. It is geared towards image generation using deep layer *activation maximization*, on a deep network [**?**] that was trained for image classification with the ImageNet [**?**] dataset. At the heart

[1] See [**?**] for a complete review, [**?**] for a published book, as well as[**?**] for the first dedicated workshop, and [**?**] and the repositories for DL4M [**?**] and the Aalto courses [**?**] for introductory tutorials. For the use of DL for symbolic music representations, see [**?**, **?**, **?**, **?**, **?**, **?**]

[2] Despite the usefulness of DL in raw audio, e.g. reducing the gap between symbolic and audio classification [**?**], it can be argued that not all MIR problems benefit from it [**?**].

[3] Multiple variants to the original WaveGAN have appeared in the literature. See for example: conditional GAN [**?**], MelGAN [**?**, **?**], WGANSing [**?**], TFGAN [**?**], DRUMGAN [**?**], SyleMelGAN [**?**], among others. In our previous work 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.

[5]

of the algorithm, a *gradient ascent* (see 2.2.2) 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 (feed-forwad) 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 [**?**] 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 targetted 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 [**?**]. 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 psychodelic aspect gave way for the *dream* in the name, raising further questions about the nature of artificial networks and their otherness that extend the limits of this paper.

2.1.2. Deep Dream in Sound

There is existing work in the application of the Deep Dream effect [**?**] to sound. In [**?**], 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 [**?**]. Finally, an important antecedent that steers away from Deep Dream is Herrmann's recent work in the visualization and sonification of neural networks [**?**]. In contrast to [**?**, **?**], 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.

2.1.3. Images as Sound

The creative potential of these approaches is radicated 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 [**?**, **?**] or the scaleogram (constant-Q transform) [**?**]. Therefore, some issues arise due to the usage of images as sound, namely the sequential aspect of audio. For example, in [**?**] Purwins et al 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 claim in [**?**]that the *asintropy* of spectrogram representations has been one of the central difficulties when transcoding architectures dealing with images to sounds. That is to say, while the spatial correlation between pixels maintains independently of their direction of an image, this is not the case when dealing with spectrogram images because the horizontal axis represents one thing (e.g. time) and the vertical, another (e.g. frequency). In DreamSound, we implemented an audio-specific solution using raw audio, with variations on an *activation maximization* function that perform filtering to further adjust the output.

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

2.2. Sonification Design

While visualization has been the main channel to understand and perceptualize deep networks [**?**], 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 [**?**, **?**], achieving important advancement in the interpretation of Machine Listening models. In [**?**], Mishra et al proved that temporal and harmonic structures are preserved in deep convolutional layers. Winters et al [**?**] advanced cientific and participant study-based work by sonifiying 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 deep convolutional neural network in an intrinsically coherent way. Three design approaches are mentioned, defined by three creative implementations of an *activation maximization* function.

2.2.1. The YAMNet Model

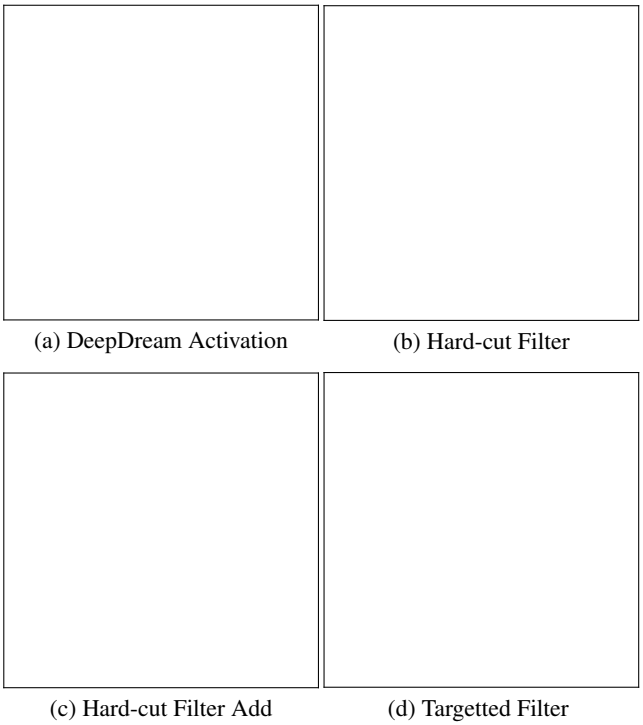
While there are many available trained models for sound classification, we have chosen a general sound source classifier YAMNet [**?**]. YAMNet is a pretrained deep neural network classifier that uses MobileNets [**?**], a depthwise-separable convolution architecture with a final activation layer that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

2.2.2. 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 [**?**], arriving at *activation maximization*. The 'gradients' are refered to here as the gradient vector between a loss and its inputs in a model. We have found similatiries in our sonifications of the gradients and in Herrmann's sonification of the activations [**?**], which sound like filtered noise with dynamic spectral envelopes. Further, given our choice of the last layer of YAMNet, we find like Herrmann the dynamic aspect of these envelopes ressembles the rhythmic elements of the input sound. Visually, the spectrogram of the gradients look like the inverted image of the spectrogram of the original sound.

2.2.3. Deep Dream approach

Following the Deep Dream approach, 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 1 (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



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

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

Figure 1: 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.

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 interesing results in the image domain, in the sound domain the additive aspect takes presedence. That is to say, the original sound and the gradients are perceived as two superimposed independent layers.

2.2.4. Filter-based approach

Therefore, a hard-cut filter 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, resulting in Figure 1 (b). Further, 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. We have proposed three extensions to this filter-based design. Figure 1 (c) extends the previous convolution with the additive element of (a) being performed between the output of the filter and the original sound, which only then is fed back into the loop.

2.2.5. Target-based approach

In Figure 1 (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 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 artists. 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* [**?**, **?**, **?**], which are dealt with in [**?**], but in general, a gradient-based approach is used on an input and a target in order to synthesize a third, hybrid sound. In [**?**], 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 approach, the gradients of the layer are convolved to the filter constructed by the target sound and then fed back into the loop.

3. RESULTS

In what follows, we describe some of the audio examples that accompany this paper. These examples were made with a single-class python package implementation of DreamSound, available for import via .[7]. The details of the implementation can be expanded in a more verbose documentation and extend the limits of this paper.

```
def combine_2(self, x, y):

    X = stft(x)
    Y = stft(y)
    X_mag, X_pha = magphase(X)
    Y_mag = tf.math.abs(Y)

    # normalize
    Y_mag_norm = normalize(Y_mag)
    # offset
    Y_mag_offset = Y_mag_norm - threshold
    # hard cut based on sign
    hard_cut = (tf.math.sign(Y_mag_offset) + 1) * 0
    # apply the hard-cut filter to the
    # magnitude of the gradient
    X_mag_cut = hard_cut * X_mag
    # apply the phase of the original sound to the
    X_mag_rephased = complex_mul(X_pha, X_mag_cut)
    # compute the inverse stft on the cut and repha
    x_new = istft(X_mag_rephased)
    # resize either x_new or y to min length so tha
    x_new, y = hard_resize(x_new, y)
    # add a small amount of the sound to the new (r
    output = tf.math.add(x_new * step_size, y)
    # inverse fft of the hard cut
    hard_cut_real = istft(complex_mul(X_pha,hard_cu

    return output, hard_cut_real
```

[7]

4. CONCLUSIONS AND FUTURE WORK

We have presented a prototype adaptation of the Deep Dream project into an audio-based musical context using tensorflow and YAMNEt. Our project began as a rapid and remote collaborative work within the Google Colab environment using Python, namely with tensorflow, numpy, and librosa, and it developed into a python package. We have contributed some research at the intersection of deep learning and audio-based musical tasks, and presented our results. The most interesting results occur when recursively applying an FFT filter built from the gradients obtained from an input soundfile and its loss. While our approach is tethered to a general sound database and a specific model, the techniques of gradient ascent for loss maximization of layer activations, and the FFT filter exposed in this paper are be portable enought for importing into other existing classification models for audio, such as for genre or musical instrument classification. Further work includes plans to extend this project towards the creation of a database of dreamed sounds, with the capability to change models and other fft filtering techniques, as well as other modifications using GANs.