UNSUPERVISED REPRESENTATION LEARNING FOR FINE-GRAINED SOUND SOURCE PERCEPTION

Etienne Bost
Aix-Marseille Université
etienne.bost@etu.univ-amu.fr

Thomas Schatz
Aix-Marseille Université
thomas.schatz@univ-amu.fr

ABSTRACT

Humans show an exquisite ability to perceive the world around them that has proven difficult to replicate in machines. For example we can tell a lot about what caused a sound from just hearing it, without any visual information. Our objective in this work is to build a model that, like humans, is able to infer detailed properties of sound-generating objects. Our approach infers object properties from an observed sound by inverting a simple but faithful source-filter generative model for natural sounds. To invert the generative model, we consider different methods based on artificial neural networks.

***K*eywords** Unsupervised · Sound sources · Generative model · Inference · Latent representation

1 Introduction

Interpreting a sound in order to infer detailed properties of the source of the sound is a task humans are able to solve with ease, while machines are still very far from reaching such a performance level [1]. Take for instance a glass falling and shattering on a hard surface. Humans can tell approximately how far it fell, from which height, the size of the glass whether it was empty or full and much more even if they haven't seen this glass before. In contrast, while state of the art systems are able to differentiate a glass falling on he floor and a basketball bouncing, they are typically unable to gather more information on the sounds they heard. These systems are considered successful at identifying sounds but they can't interpret them.[1]

Here, we are going to consider a simple model that is able to generate realistic sounds from object properties and invert it so we can predict these object properties from a sound. Interpreting audio signals in itself is a hard task but we know how to synthesize sounds, for instance through physics simulation engines. These methods take object properties as input like their shapes, weight, material, motion... and output the audio signal they should produce or perhaps a whole probability distribution over possible resulting sounds. Inverting such a generative model then means taking a sound as input and outputting the object properties most likely to have synthesised it according to the generative model. We pay particular attention to the balance between simplicity and faithfulness for the generative model, as we believe that accurate model inversion is important for this project and we expect in general simpler models to be simpler to invert as well.

Source-filter generative models for natural sounds recently introduced in the acoustic and psycho-acoustic literature [2, 3, 4] appear particularly appropriate for our purposes, as they are of a simple form and use fully differentiable equations to generate plausible sounds of a great variety. These models are based on the idea that most if not all sounds are produced by the contact between two objects [5]. They express the sound resulting from the contact as the convolution between the contact force (the source) and impulse responses associated with each of the involved objects (the filters). These impulse responses characterize the vibrational properties of those objects, i.e. how they convert mechanical stimulation into acoustic signal. Let us note that what we call object here is quite general and may include fluids like the air flow acting on the vocal chords during speech production1.

To formulate a source filter model capable of generating the widest possible variety of natural sounds, the source component appears to be the limiting factor. Indeed, while general approaches are available to model the filter

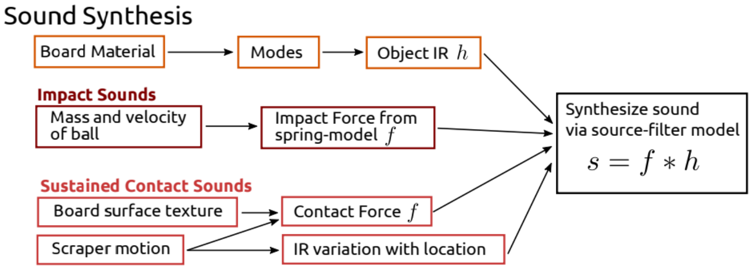

Figure 1: *Representation of the source-filter model used to generate impact and scraping sounds presented by James Traer and Josh Mc Dermott [2]*

components (impulse responses), modeling the source component requires a way to compute the contact force for every sort of contact likely to generate sound, and we have not found a direct way to achieve this so far. It is possible, however, to classify sonic event or sounds based on the type of contact that generated them [5] (see Figure 2). One possible approach would therefore be to develop a model of the contact force separately for each type of contact and use a probabilistic mixture of those models to generate generic sounds.

Because of this restriction on sound sources, we will focus our attention on scraping sounds. Scraping sounds are sounds generated by the action of rubbing a solid object over a surface. In his paper [3], Josh Mc Dermott explained in details how to estimate the contact force in the case of scraping sounds. In this work, we focus exclusively on modeling the source component for contacts between vibrating objects and more specifically for scraping sounds.

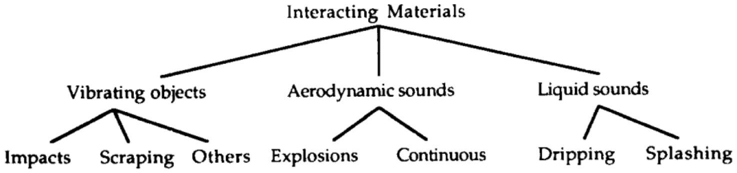

Figure 2: *A hierarchical description of simple sonic events by W. Gaver 1993 [5]*

We have identified two possible approaches to inverting the generative model, i.e. to retrieving likely values for the parameters of the source and filter components (the model's *latent variables*) given an observed sound. The first approach consists of generating data with the source-filter model for set values of the latent variables and use it to train a neural net with explicit supervision to retrieve the latent variables from the signal they generate. In other word, we are trying to approximate the inverse function of the source-filter generative model in a supervised learning setting. We might also explore a second approach that can be understood as a form of analysis by synthesis [6]. In this approach, we use the generative model to fit the parameters of a variational autoencoders (VAE) [7]. A variational encoder is comprised of an encoder and a decoder neural net. The decoder neural net generates a sound given values of the latent variables. We plan to use our generative model directly as the decoder since it is differentiable. The encoder neural net generates values for the latent variables given an observed sound. It will be implemented as a generic neural net whose parameters will be chosen so as to produce values for the generative model latent variables that result in a generated sound as similar as possible to the observed sound when the decoder is applied to them.

We expect our inverted model to be able to perform well in at least two tasks: detecting scraping sounds in common auditory scenes and inferring object properties when scraping is detected in these auditory scenes. We will evaluate our model's detection abilities by testing it on increasingly difficult annotated auditory scenes from simple isolated scraping sounds up to real world recordings containing embedded scraping sounds. To evaluate our model's inference abilities, we will consider a set of sounds for which qualitative knowledge regarding the sound production process is available—for example, the sound might be produced by scrapping a metal bar back and forth against a wooden table with a regular motion. We will assess whether the values of the latent variables associated to each of these sounds by our model—which describe the inferred vibrational properties of the filters and the inferred motion and other properties of the source—are consistent with this qualitative knowledge.

We believe the approach we propose in this paper may help building more intelligent artificial agents with finer perceptual skills than has been possible so far. It might also help understand human intelligence better by providing the means to build concrete computational models of humans' sound source perception abilities, with applications in the fields of neuroscience and cognitive science.

2 Related work

Sound sources perception
The problem we are tackling is not new. Several articles have already proposed solutions based on precise physics simulations [8] or supervised learning methods [9]. The sounds on which it is currently possible to use these methods are limited in their diversity, however, and extending them to a larger set of sounds would require a large effort. This encouraged us to search for a solution using unsupervised learning with a simpler generative model than what has been used so far, which might perform accurately under a wider variety of conditions and might be less expensive to extend in the future.

Auditory synthesis
A few phenomenological generative models able to synthesize a variety of sounds from a description of the physical interaction producing those sounds have been developed [3, 2]. They are far simpler than detailed physical models that use numerical methods to solve the wave equation [10]. To the best of our knowledge, however, there has been no attempt to use modern machine learning techniques to invert this new generation of simpler generative model.

3 Methods

In this part, we are first going to detail the source-filter model we use to synthesize scraping sounds and then we will present two plausible solutions to reverse this model and perform the inference task.

3.1 Source-filter model

As we have said in our introduction, the first step of solution we propose is to build a simple but faithful generative model for scraping sounds. The model we describe here is able to generate scraping sounds using the source-filter model from probability distributions and variables describing the sound-producing objects and their interaction. The sound-filter model states that any audio signal can be expressed as the convolution of a contact force with the sum of the impulse responses corresponding respectively to the scraping objects and the scraped surface.

$$s(t) = f(t) * [h_{scrap}(t) + h_{surf}(t)]$$

The product of convolution operator is defined as follows:

$$f * h : t \mapsto (f * h)(t) = \int f(\tau)h(t-\tau)d\tau$$

where h corresponds to the object's impulse response

It has been argued that an object's impulse response (IR) doesn't need to be precisely computed using cutting-edge physics engines and can be approximated with a linear combination of exponentially decaying sinusoidal waveforms [2]. The intuition behind this approximation is that an object submitted to an impulsion, meaning a mechanical excitement of their rigid bodies will enter a vibrating state producing a sound from which comes the sinusoidal signals [11]. Because of the rigidity of the objects their vibrations will decrease exponentially at a rate depending on their material. Instead of precisely computing the IR with a physics engine, this approximation allows us to greatly reduce calculations. Experiments conveyed on human perception of impulse responses show that this approximation works extremely well for rigid objects [2]. This exponentially decaying sinusoidal approximation is computed from 3M + 2N parameters and is given by the following formulas:

$$h_{surf}(t) = \sum_{n}^{N} 10^{(\alpha_{surf\,n} - \beta_{surf\,n}.t)/20} + \sum_{m}^{M} 10^{(a_{surf\,m} - b_{surf\,m}.t)/20} cos(w_{surf\,m}.t)$$

$$h_{scrap}(t) = \sum_{n}^{N} 10^{(\alpha_{scrap\,n} - \beta_{scrap\,n}.t)/20} + \sum_{m}^{M} 10^{(a_{scrap\,m} - b_{scrap\,m}.t)/20} cos(w_{scrap\,m}.t)$$

Parameters a, b, ω, α and β are sampled from the following distributions:
$(a_{scrap}, b_{scrap}, \omega_{scrap}) \sim \mathcal{N}(\mu_{M\,scrap}, \Sigma_{M\,scrap})$
$(a_{surf}, b_{surf}, \omega_{surf}) \sim \mathcal{N}(\mu_{M\,surf}, \Sigma_{M\,surf})$
$(\alpha_{scrap}, \beta_{scrap}) \sim \mathcal{N}(\mu_{N\,scrap}, \Sigma_{N\,scrap})$
$(\alpha_{surf}, \beta_{surf}) \sim \mathcal{N}(\mu_{N\,surf}, \Sigma_{N\,surf})$
where
a_i, b_i, c_i are vectors of dimension M
α_i, β_i are vectors of dimension N
with $i \in \{scrap, surf\}$
According to the paper by James Traer [2], M=15 and N=30 are able to produce realistic scraping sounds. In order to keep our model as simple as possible, we will try to find empirically the minimum values for N and M to generate plausible scraping sounds.

The other component that is required in the source filter model is the source, which corresponds to the contact force between two objects responsible for a sound. The contact force in the case of scraping is defined as the sum of two force components: vertical and horizontal force. The vertical force results from the action of pressing the scraper down on the surface (related to the scraper mass in absence of active pressing). The horizontal force originates from the collision between the scraper and steep asperities (hollows and bumps) on the surface texture.

$$f(t) = f_h(t) + f_v(t)$$

Following James Traer's approach [2], we compute the force components according to the following equations:

$$f_v(t) = m(\frac{\partial^2 S(x,y)}{\partial x^2}|v_x(t)|^2 + \frac{\partial^2 S(x,y)}{\partial y^2}|v_y(t)|^2)$$

$$f_h(t) = \beta_1|\frac{\partial z(x,y)}{\partial x}v_x(t) + \frac{\partial z(x,y)}{\partial y}v_y(t)|^{\beta_2}$$

These formulas introduce more latent variables of the model. These variables represent objects' properties such as the mass and position of the scraper on the surface described by $x(t)$ and $y(t)$ as well as the surface depth profile $z(x, y)$. The velocities on the x and y axes are computed from the trajectories of the scraper. Also, we will consider the constants $\beta_1 = 0.05$ and $\beta_2 = 1$, following [2].

Notice that we also introduce the variable $S(x, y)$ in our source-filter model which corresponds to the vertical trajectory of the scraper. Where most models have assumed that the vertical trajectory of the scraper follows precisely the surface depth profile, Agarwal and colleagues argue that it is unrealistic [3] and presents a method to estimate S from z, which we adopt here in a simplified form:

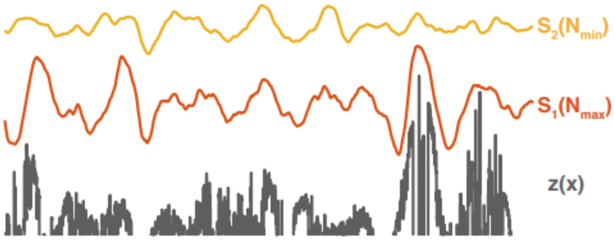

Figure 3: *Trajectory S of the scraper point. The trajectory is determined by the surface depth profile ($z(x)$, shown in gray) and the applied normal force N. Larger normal force (red) produce a more extreme scraper trajectories than smaller normal forces (yellow). Figure by Agarwal and colleagues [3]*

$$\frac{\partial^2 S(x,y)}{\partial x^2} = \frac{1}{\alpha_x} tanh(\alpha_x \frac{\partial^2 z(x,y)}{\partial x^2})$$

$$\frac{\partial^2 S(x,y)}{\partial y^2} = \frac{1}{\alpha_y} tanh(\alpha_y \frac{\partial^2 z(x,y)}{\partial y^2})$$

where $\alpha_x = \alpha_y = \frac{1}{2}(0.01 + 0.05) = 0.03$

Note that these equations are a simplification of the full model presented by Agarwal and colleagues. However, according to his results, this simplification still allows us to synthesize plausible sounds (see Figure 4). We will perform qualitative listening tests with the generative model and if it turns out we cannot generate realistic-sounding scraping sounds with the simplified version, we will use the full model.

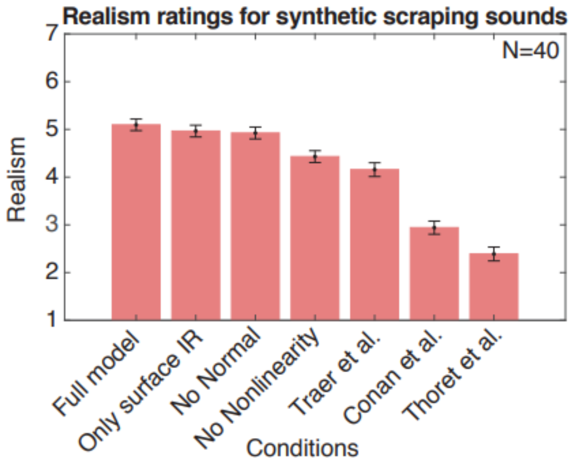

Figure 4: *Realism score of scraping sounds synthesised using the 7 different simplifications of the source-filter model judged by 40 participants [3]. The simplification we use corresponds to the "No Normal" condition.*

We use a probability distribution to describe the variable $z(x, y)$ which designates the relative height of the surface profile at the point of coordinate (x, y). We argue that it is really difficult to find the equation for $z(x, y)$ without a full physical modelling of the surface or a microscopic observation of it. Nevertheless, we only need $z(x, y)$ at the scraper position (x, y) at each instant. Unlike the approach in Agarwal and colleagues, we use a probability distribution $\mathcal{N}(\mu_z,\, \sigma_z^2)$ from which we sample the values for z we care about.

To sum up, the latent variables of our model are μ_z, $\sigma_z^2, m, x, y, \mu_{M\,scrap}$, $\Sigma_{M\,scrap}, \mu_{M\,surf}$, $\Sigma_{M\,surf}, \mu_{N\,surf}$, $\Sigma_{N\,surf}, \mu_{N\,surf}$, $\Sigma_{N\,surf}$. The source-filter model we have described has no free-parameters and appears fully differentiable.

3.2 Inference

The main goal of the inference task is to assign values to the latent variables so the signal generated using our model with this values is as similar as possible to a certain observed signal. As mentioned in the introduction we have thought of two main approaches to estimate this distribution. The first one is based on supervised learning and the second one on variational autoencoders.

3.2.1 Supervised learning

The first method consists of using supervised learning methods to solve the inference problem. In this approach we start by randomly generating values for the latent variables in a appropriate range for generating plausible-sounding natural sounds (as determined by qualitative listening tests with the generative model). We then use the generative model to obtain the associated waveforms and we use this dataset as training data in a supervised learning task where the goal is to predict the latent variables from the observed sound. Since the labels are auto-generated, this is a form of self-supervised (unsupervised) learning, even though it assumes the form of a supervised learning problem.

To solve what we have now defined as a supervised learning task, we will start by using a simple feed-forward deep neural network without any specific architecture such as recurrences or convolutions. The input to this network is an audio signal waveform presented as a vector of fixed length D (representing the last $D \times f$ seconds of input, where f is the signal's sampling frequency in Hertz) and the output is going to be the latent variables of the generative model predicted to have generated the input signal. We could also try to give the spectrogram of the audio signal as input to the network.

The generative model we are trying to reverse is nothing more than a function $f(z) : \mathbb{R}^d \to \mathbb{R}^n$ where z are the latent variables of the model, d the number of latent variables and n the dimension of the vector coding the generated audio signal (conditions by the sampling rate and signal duration). Reversing the model means finding the function $g(s) : \mathbb{R}^n \to \mathbb{R}^d$ which gives an inverse-image of s by function f where s is a dimension n vector encoding an audio signal. Considering the formula described before for f, finding an exact formula for g is probably extremely difficult if even possible. Instead, based on the observation that any function, no matter how complicated can be evaluated with a sequence of elementary operations[12] involving one or two arguments at a time, we are going to evaluate g with a compositions of sum and products between simple coefficients. The neural network architecture solves this very task. In this part we will describe a basic deep neural network architecture we could use in our problem to emphasize how it can estimate latent variables. In a few words, a neural network is a sequence of operations performed on the inputs to get the outputs. At each iteration, the coefficients involved in these operations can be tweaked in order to optimize a loss function computed from the outputs. A neural network can be represented as a graph where the operations are performed by neurons (vertexes of the graph) distributed in different layers.
In our case, the input layer will be the vector encoding the signal waveform and the output layer latent variables predicted by the network to have generated the input signal according to our generative model.

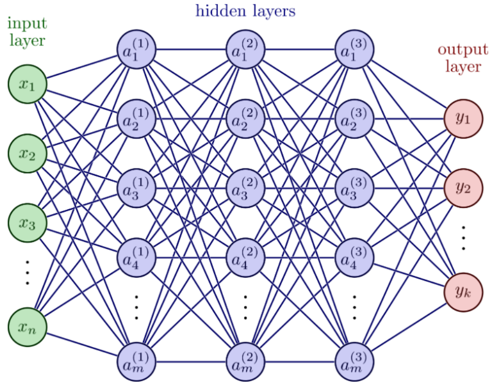

Figure 5: *Representation of a neural network with three hidden layers[13]*

The output of each neuron depends on the output of every neurons in the previous layer, its synaptic weights as well as a bias coefficient. The neuron output is obtained by taking a linear combination of the output of the neuron in the preceding layer, weighted by the neuron's synaptic weights and passing the result through an activation function σ.

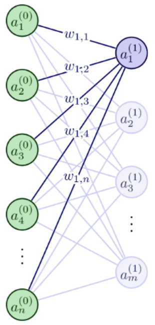

Figure 6: *Activation function of neuron a_1 on a hidden layer [13]*

In this example, the output of neuron a_1 on a hidden layer will be $\sigma_{a_1^{(1)}}(w_{1,1}a_1^{(0)} + w_{1,2}a_2^{(0)} + ... + w_{1,n}a_n^{(0)} + b_1^{(0)})$

where b is the bias of the neuron and w its weight. Activation functions σ can take different forms. The most common is called Rectified Linear Unit (ReLU) and is defined as follows: $ReLU(x) = max(0, x)$.

Loss function and optimization. In order to choose network parameters that lead to a good estimation of g, we need to specify a loss function measuring how far the latent variables predicted by the network are from the latent variables that were actually used to produce the sound. We plan to start our investigations by using a simple linear combination of square euclidean distances as the loss, except maybe for the latent variables corresponding to impulse responses, for which it is not clear that an euclidean distance would be meaningful. We are considering using the earth mover's distance [14] if we can find a way to efficiently propagate gradients through it. The earth-mover's distance is a metric to measure the distance between two probability distribution. Informally, this distance is like seeing the probability distributions as two different piles of dirt over a certain space and measuring the minimal effort to turn one pile into the other. The effort or cost is defined as the amount of dirt moved time the distance on which it has been moved. Formally, the earth mover's distance between two distribution probabilities P and Q is the solution to the following optimisation problem. Assume that P has m clusters with $P = \{(p_1, w_{p1}), (p_2, w_{p2}), ..., (p_m, w_{pm})\}$ and similarly Q has n clusters $Q = \{(q_1, w_{q1}), (q_2, w_{q2}), ..., (q_n, w_{qn})\}$. We are trying to find $F = [f_{i,j}]$ and $D = [d_{i,j}]$ to minimise the following quantity:
$min \sum_{i=1}^{m} \sum_{j=1}^{n} f_{i,j} d_{i,j}$
subject to the constraints:
$f_{i,j} \geq 0, 1 \leq i \leq m, 1 \leq j \leq n$
$\sum_{j=1}^{n} f_{i,j} \leq w_{pi}, 1 \leq i \leq m$
$\sum_{i=1}^{n} f_{i,j} \leq w_{qj}, 1 \leq j \leq m$
$\sum_{i=1}^{m} \sum_{j=1}^{n} f_{i,j} = \sum_{i=1}^{m} w_{pi} = \sum_{j=1}^{n} w_{qj}$
We then define the earth mover's distance using the optimal solution to this problem, as:

$$EMD(P, Q) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{i,j} d_{i,j}}{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{i,j}}$$

Our loss function is a sum of euclidean or earth mover's distances for each latent variable we are trying to predict. We could add a coefficient to each term of the sum to emphasize or deemphasize certain latent variables.

Once we have this loss function, we can optimize the neural net parameters by taking stochastic gradients of the loss function, as is usual when training neural networks. More specifically, the stochastic gradients are computed through back-propagation. This step effectively tweaks the weights and biases of each neuron in order to minimize the loss function.

Such an approach appears as a good solution in order to invert our model since we can generate our own data for the supervised learning task. We might experience a transfer problem, however, since our final goal is to use the reverse model on real word acoustic scenes which have not been directly generated by our generative model, unlike all the data seen during training.

3.2.2 Variational Autoencoders

A second solution we consider consists of formulating the problem of inferring latent variables from observed sounds within a variational autoencoder framework. In this setting, an encoder neural network is used to predict the latent variables from an observed sound as before, but the predicted latent variables are then used to resynthesize a sound using the generative model and a loss is defined in the sound space between the original sound and the re-synthesized sound instead of being defined in the latent variable space. The encoder neural net parameters are optimized by backpropagating the gradients of the loss through the generative model.

With this method it becomes possible to optimize the model without having access to a target value for the latent variables and one option would be to use the supervised learning approach first to initialize an encoder network and then use the variational autoencoder approach on actual sound recordings (for which target latent variables are not available) to fine-tune the encoder parameters. This might help address the potential transfer issues that we mentioned in the previous section.

Let us now present the variational autoencoder framework in more details. First we need to formulate our generative model in a probabilistic form. From the waveform equation $f(Z)$ obtained with the source-filter model where Z is the set of latent variables we have listed earlier, we build the corresponding generative model using a Gaussian distribution. In general, a generative model is a probability distribution of the generated data P_{XZ} where X denotes the observed variable (i.e the audio signal) and Z denotes the latent variables (or more generally a family of such distributions, indexed by a parameter set Θ). The simplest distribution we can use to approach our generated data is a Gaussian distribution. $X \sim \mathcal{N}(\mu, \Sigma)$. μ is the vector describing the average energy levels at the instants considered according to the sampling rate and the D parameter. Σ is the standard deviation of the generated sounds with this mean.

To introduce a bit of formalism, let z be the set of latent variables of the model and x the observed data (signal waveform). Generally, generative models are presented as a parameterized probability distribution over the data $p_\theta(x, z)$ where θ denotes the parameters of the model. However, our model doesn't have any free parameter, so we will denote it as $p(x, z)$. By integrating the latent variables, we can obtain the distribution of data x generated by this model: $p(x) = \int p(x, z)dz$

From these notations, the inference step is about finding $p(z|x)$ knowing we already have access to $p(x|z)$ with the previously described Gaussian distribution.

This problem fits within the variational autoencoder (VAE) framework by replacing the decoder neural net with our parameter-free, differentiable generative model. Variational autoencoders are a form of probabilistic autoencoders aiming to model the distribution of an observed data x by using a latent representation z of the data such that $p(x) = \int p(x, z)dz$.
$p(x) = \int p(x|z)p(z)dz$
It is possible to estimate $p(x|z)$ with a neural network called a decoder. Here, we make the hypotheses that $p(z)$ is a Gaussian distribution with the identity as covariance matrix. The main difficulty is that we can't integrate these distributions in order to find $p(x)$. The method used to get around this problem is based on Bayes rule: $p(x) = \frac{p(x|z)p(z)}{p(z|x)}$

Similarly as the method to find $p(x|z)$, VAE estimate distribution $p(z|x)$with a neural network called an encoder. This distribution will be approached by $q_\phi(z|x)$ where ϕ are the parameters of the distribution. VAE train both networks (encoder and decoder) in order to maximise the likelihood between generated data and real observed data. This process is equivalent to maximising the following function: $L_\phi = \mathbb{E}_{q_\phi(z|x)}[log(p(x|z)) - log(q_\phi(z|x))]$ called ELBO (evidence lower bound) which is a lower bound on the maximum likelihood. The gradient of this function can be obtain by performing a reparametrization trick which allows us to optimize parameters ϕ by doing a stochastic gradient descent in order to maximize ELBO. With enough iterations, data sampled from $p(x|z)$ learned with the decoder can generate data resembling the training set and $q(z|x)$ gives us the latent variables z used to represent data x.

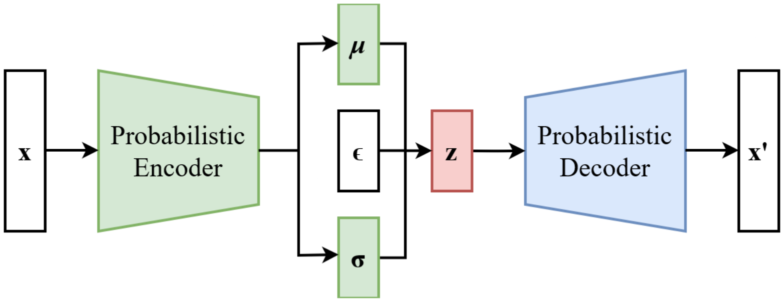

Figure 7: *Structure of a Variational Autoencoder using the reparameterization trick,from Wikipedia*

In our case, the decoder part is actually the generative model based on the source-filter model that can accurately generate data x from latent variables z so there is no need to learn it using a neural network. The distribution $p(x|z)$ is then fixed. Since we are trying to invert the decoder, the training data we will use is a set scraping sounds generated with the model. In order to learn an encoder from a fixed decoder, we need to perform a stochastic gradient descent on the parameterized ELBO function. Since our source-filter model appears fully differentiable, we should be able to compute the gradient of $p(x|z)$. The learned encoder giving us the estimated distribution of the latent variables according to an observed piece of data allows us to solve the inference problem.

4 Experiments and results

We will evaluate our approach in several ways. First, we will get familiar with the generative model and its latent variables space by performing qualitative listening tests on synthesized sounds. Second, we will evaluate the performance of the reverse model in a series of scraping sound detection tasks of systematically varying difficulty. Third, we will evaluate the performance of the reverse model in scraping sound analysis tasks, in which a scraping sound is given and the goal is to infer qualitative properties of the scraper and of the surface on which it scrapes.

4.1 Scraping sounds synthesis using the source-filter model

The first step of the testing process will consist in a qualitative test of the source filter model. The goal is here to simply generate sound samples from latent variables we give to the model. There is no clear way to properly evaluate the realism of a synthesized sound other than asking humans to do it knowing the latent variables from which it was generated. Even though this step does not constitute a formal evaluation of our model, it is still a vital part of this work for two reasons. The first reason is to know which simplifications of the model we are able to do and still obtain realistic sounds. The source filter model used to synthesize scraping sounds that we described in the second part of this report is already a simplification of what Agarwal and colleagues [3] proposed. For instance, we do not take into account the normal forced applied on the scraper causing it to not precisely follow the profile of the surface on which it scrapes. According to Agarwal and colleagues' results, this kind of simplification of their model doesn't change much the realism of the generated sounds, which is why we did it in the first place. We still need to confirm this on sounds that we have generated ourselves however. We believe the simpler our model is, the fewer latent variables we have and so the better the inference will be once we invert the model.

The second reason for this qualitative test, and perhaps the most important is to comprehend the space in which the latent variables evolve. Our generative model may be able to generate sounds that do not constitute plausible scraping sounds. That means there may be value ranges for the latent variables that do not result in scraping sounds. A very simplistic example for that would be the mass of the scraper taking negative values, but there are also many value ranges for the expectation of the probability distribution giving the parameters for the impulse responses that may give IR not corresponding to any physically possible material. Knowing the approximate value ranges for the latent variables to generate plausible scraping sounds could allow us to insert helpful priors into the inference method, which might substantially impact the system's performance.

4.2 Detection of scraping sounds

We propose 4 increasingly difficult experiments in order to test our model's ability to detect scraping sounds in auditory scenes. In each experiment, the goal is for the system to predict which intervals of a given auditory scene contain scraping. The predicted intervals are compared to ground truth intervals, so as to produce an error score.

The first experiment is on scenes that have no sound for a certain period, then a brief scraping sound sample lasting for a few seconds and then no sound again until the end of the scene. These artificial auditory scenes should last about 20 seconds each. For this experiment in particular, the scraping sounds are samples generated by our source-filter model. The goal here is to test the reversing of the generative model.

The second experiment is very much like the first except the scraping samples will be real scraping samples that are either from free audio samples libraries or from the recorded scraping sounds used by Agarwal and colleagues in their experiments [3]. Here we are testing how our model transfers to real world sounds.

The third experiment is performed on hand-crafted auditory scenes from various sounds samples. The length of these scenes is still about twenty seconds, and they are composed of different sound effects, each lasting a few seconds and put one after the other in a random order. Like before, these sound effects are from online free sound libraries or from recorded sounds by Agarwal and colleagues. Some scenes will contain one or more scraping samples but some will not include any scraping at all. The goal of this experiment is to test how well the model can differentiate scraping from other sounds.

The last experiment is conveyed on recorded auditory scenes from the real world that may contain scraping sounds. Unfortunately, we couldn't find existing datasets containing this type of auditory scene with the moments of scraping precisely annotated. Instead, we collect sounds from video streaming websites or free audio libraries. We seek auditory scenes that have a high chance of containing scraping based on our intuition. These scenes can include factory noises, ice skating sounds... We annotate by hand the times when there are scraping sounds in these scenes. This test asseses the accuracy of our model on real world sounds and is both the most interesting and the most challenging.

Note that each experiment answers a different question about the model performance and can guide its improvement.

4.3 Inferring object properties from scraping sounds

Once we have detected a scraping sound, we can try and infer latent variables associated with this sound in our generative model. This will provides model predictions regarding the properties of the scraper and the surface on which it scrapes. More specifically, in the case of scraping we are aiming to produce a small description of the event similar to how Agarwal and colleagues describes the scraping sounds they generates with the source-filter model. That means, for example, giving the material of the scraper, the material of the surface and the motion of the scraper on the surface. The material of the scraper is obtained from the probability distributions giving the parameters to estimate the impulse response of the scraper. The material of the surface is given by its IR similarly to the scraper but also by its depth which is given by the latent variable z. From the depth we can estimate the roughness of the surface and guess the material from there using preexisting classification tables of materials by roughness. The motion of the scraper is given by variables x and y. We will try to infer general descriptions of the motion like "circle-like" or "back and forth movement repeated 4 times" for example.

To test the inference in our model, we use the very same auditory scenes as in the four detection experiments. In order to have consistent results, we are going to test the inference only on the auditory passages when scraping is detected by the model.

5 Discussion and perspectives

In this final part we will come back to justify some of the points in our model and try to give a perspective on how some of them could have been done differently. Finally we will sum up the contribution this paper brings and talk about how it could be improved.

5.1 Discussion

Nowadays the classical approach to solving any machine learning problem is to use supervised learning and neural networks but such methods cannot easily be applied in this case. Studies on the subject suggest it is very hard to build machine learning model that can interpret sounds (over identifying them) [1]. We believe the main difficulty resides in the lack of data annotations describing precisely the properties of each object causing the sounds in an auditory scene at every moment. This type of labels that could be used for training supervised models would be particularly difficult and expensive to obtain in this case because of the diversity of the properties we can infer from sound sources (there is no complete list). Furthermore object properties of interest may not be readily observable in video data (e.g. objects mass or material, speed...). This means that collecting the type of labels needed for training standard supervised models to infer detailed properties of sound sources might require recording custom auditory scenes with controlled elements whose properties have been specially measured.

Instead, we have taken the non-standard approach of drawing inspiration from how we think humans perform inference about sound source properties, i.e. without explicit supervision. It seems likely that humans would be able to develop this ability without relying on explicitly supervised training. One source of evidence that may support this view is the literature on animal's ability to infer properties of sound sources, which they would presumably develop without explicit supervision. We plan to investigate this literature in the future.

The generative model we have presented based on a source-filter approach can only synthesize scraping sounds. The reason for that was that we didn't find a simple way to estimate the contact force between objects for all sonic event categories at once, although it seems to be possible to model the contact force separately for each category. We note that this restriction to a particular sound category does not affect the generality of our approach, which could be applied in a similar way to other classes of sounds, provided a model for the contact force is supplied. In future work, we will investigate whether it might be possible to spell out a unified formula for contact force. If we consider sounds produced by vibrating objects, for example, there are similarities between scraping contact force and impact contact force, that suggest that a unified model might be possible.

We could also attempt to progressively cover all possible type of sound-producing contacts to form a mixture model that could handle all natural sounds. If we can combine our model for scraping sounds with similar models that can infer sound source properties for other sound categories, the detection task becomes a multi-class classification problem where we try to identify to which category a certain sound belongs. Because with this approach, the contact force is different for each sound category, the latent variables we infer might not always be the same in the inference task.

For the second inference method we have described where we use variational autoencoders, we express the decoder as a probability distribution we said to be estimate by a Gaussian distribution but it may not be most accurate. Other probably distributions might be more adapted. We have made a similar approximation for the depth of the surface profile (denoted z in the generative model) we sampled from a Gaussian distribution. However, even if a Gaussian distribution may be too simple to describe a lot of profile surfaces. There exists more precise statistical texture models [15] we could try to use. The use of a probability distribution is a departure from the approach of Agarwal and colleagues and we will investigate whether it leads to realistic sounds and if not, we may use texture scanning data as they did.

We could obtain the material of the surface from its profile depth. Once the model is inverted, one of our main goals is to get the material of the scraper and of the surface. For the scraper, this information is contained in both its impulse response and its profile depth surface (denoted $z(x, y)$). A profile depth surface characterises a texture and textures are proper to material classes. For instance a surface texture can be characterised by its roughness. A roughness coefficient like R_a can be calculated from the depth of the surface profile z. For example, in one dimension:
$R_a = \frac{1}{L} \int_0^L |z(x)| dx$ where L is the length of the surface. There are analog formulas for a two dimensional surface. The advantage a roughness coefficient gives us is that there are tables giving the correspondence between roughness coefficient ranges and usual materials. Combined with the impulse response of the surface, we hope to obtain a good estimation of the surface material.

We have thought about making our model robust to background noise. Since in most auditory scenes, there is a background noise on top of the scraping sounds, we can try to emulate that by adding a Gaussian white noise on top of our generated signal. That way, during the inference process our model will learn to retrieve latent variables from a signal with a background noise and might transfer better to real world auditory scenes.

Depending on the results of the first generative model, we might want to improve it it order to better discriminate scraping sounds from other sounds (detection task). To do this we want to combine our model with a model that can generate about any audible sound creating a mixture of models. A generative model based on all audible sounds can be a Gaussian distribution which parameters have been tweaked. From an audio signal, we can have the probability that this sample is scraping and the probability this sample is another type of sound. We might get better results for the detection task than the single generative model we have described before.

5.2 Conclusion and perspectives

In this report we have described an approach to make a machine infer detailed object properties from an audio signal inspired by human abilities. We mainly focused our efforts on scraping sounds, proposing a source-filter model capable of generating scraping sounds from latent variables that represent objects properties. We then discussed two ways to reverse this model in order to find the latent variables from an observed audio signal: (self-)supervised training of an artificial neural network and variational autoencoders. Using supervised learning methods to reverse the generative model is perhaps the most direct approach and the one we will try first. The experiments we have described will allow us to test our solution's ability to generate realistic sounds, detect certain type of sounds in controlled auditory scenes and infer object properties from the latent variables obtained from the same auditory scenes.

At the time this report was written, only the generative model based on the source filter-model was implemented, we are currently in the process of exploring latent variable ranges which correspond to the synthesis of realistic-sounding scraping sounds. Next, we are aiming to implement the training of the inverse model in order to be able to perform the detection and inference experiments we have described before the end of the internship. Since a simple approach to computing impact sounds is available [2], we are also hoping to implement and reverse a generative model similarly to what we have done for scraping but for impact sounds this time. That way we can try to combine the two models and use them on a more diverse set of example auditory scenes including both scraping and impact sounds.

In order to design a more complete model, we will need to find ways to compute the contact force for additional types of contact responsible for naturally occurring sounds, beyond scraping and impact sounds. The source-filter model, the impulse responses and the method to reverse the model might stay the same however. It is conceivable that, in the end, the system built may be able to infer almost as many object properties on as many different sort of sounds as a human can.

There are also a lot of potential improvements we could bring to the model itself. The first one is to find the right priors for the inference process. We already mentioned this in the section on generative model evaluation, but we expect that finding plausible values or value ranges for latent variables may speed-up and improve the inference process a lot (independently of the method used). The other improvement will be about selecting the right model for the supervised learning task. We have mentioned in this report the use of a standard deep neural network but we have no guarantee that it is the best model architecture for this task. We also don't have any notion of the optimal amount of layers and neurons per layer either and as we have seen from previous work on deep learning, these parameters can greatly affect the performances of the system. The inference process may be helped by the selection of optimal parameters on the neural network for this task. Once this is done, a very interesting experiment to perform would be to compare the results of the optimized deep learning model we just described with the results of other inference methods such as the use of the variational autoencoders architecture we have previously described. Hopefully, the same experiments we have already described are enough to compare the different methods, models and architectures. Overall, we are hopeful that with the right methods and optimizations, a task as difficult as sound sources perception may be achievable without any supervision and using only existing approaches to machine learning.

References

[1] Michael J Bianco, Peter Gerstoft, James Traer, Emma Ozanich, Marie A Roch, Sharon Gannot, and Charles-Alban Deledalle. Machine learning in acoustics: Theory and applications. *The Journal of the Acoustical Society of America*, 146(5):3590–3628, 2019.

[2] James Traer, Maddie Cusimano, and Josh H McDermott. A perceptually inspired generative model of rigid-body contact sounds. In *The 22nd International Conference on Digital Audio Effects (DAFx-19)*, 2019.

[3] Vinayak Agarwal, Maddie Cusimano, James Traer, and Josh McDermott. Object-based synthesis of scraping and rolling sounds based on non-linear physical constraints. *arXiv preprint arXiv:2112.08984*, 2021.

[4] Mitsuko Aramaki, Mireille Besson, Richard Kronland-Martinet, and Sølvi Ystad. Controlling the perceived material in an impact sound synthesizer. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(2):301–314, 2010.

[5] William W Gaver. What in the world do we hear?: An ecological approach to auditory event perception. *Ecological psychology*, 5(1):1–29, 1993.

[6] Stan Z. Li and Anil Jain, editors. *Analysis-by-Synthesis*, pages 35–36. Springer US, Boston, MA, 2009.

[7] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[8] Zhoutong Zhang, Qiujia Li, Zhengjia Huang, Jiajun Wu, Josh Tenenbaum, and Bill Freeman. Shape and material from sound. *Advances in Neural Information Processing Systems*, 30, 2017.

[9] Yunyun Wang, Chuang Gan, Max H Siegel, Zhoutong Zhang, Jiajun Wu, and Joshua B Tenenbaum. A computational model for combinatorial generalization in physical auditory perception, 2017.

[10] V Abrol, A Abtahi, P Agrawal, Y Ai, X Alameda-Pineda, P Alku, A Amar, J Amini, and A Ando. 2020 index ieee/acm transactions on audio, speech, and language processing vol. 28. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28, 2020.

[11] Guy-Bart Stan, Jean-Jacques Embrechts, and Dominique Archambeau. Comparison of different impulse response measurement techniques. *Journal of the Audio engineering society*, 50(4):249–262, 2002.

[12] Stephen Wright, Jorge Nocedal, et al. Numerical optimization. *Springer Science*, 35(67-68):7, 1999.

[13] Neural networks by izaak neutelings. `https://tikz.net/neural_networks/`. Accessed: 2021-09-12.

[14] Frank L Hitchcock. The distribution of a product from several sources to numerous localities. *Journal of mathematics and physics*, 20(1-4):224–230, 1941.

[15] Josh H McDermott and Eero P Simoncelli. Sound texture perception via statistics of the auditory periphery: evidence from sound synthesis. *Neuron*, 71(5):926–940, 2011.