Problems using deep generative models for probabilistic audio source separation

Maurice Frank
Amsterdam Machine Learning Lab
University of Amsterdam
maurice.frank@posteo.de

Maximilian Ilse
Amsterdam Machine Learning Lab
University of Amsterdam
m.ilse@uva.nl

Abstract

Recent advancements in deep generative modeling make it possible to learn prior distributions from complex data that subsequently can be used for Bayesian inference. However, we find that distributions learned by deep generative models for audio signals do not exhibit the right properties that are necessary for tasks like audio source separation using a probabilistic approach. We observe that the learned prior distributions are either discriminative and extremely peaked or smooth and non-discriminative. We quantify this behavior for two types of deep generative models on two audio datasets.

1 Introduction: Langevin dynamics for source separation

Our initial goal was to use Langevin dynamics [11] in combination with deep generative priors to perform source separation of audio mixes. Our approach closely follows the work of [3], where mixed images are successfully separated. The biggest advantage of the approach used by [3] is that it does not rely on pairs of source signals and mixes as required by SOTA audio source separation models [10, 4]. Last, in contrast to [9], we are interested in performing the source separation in the time domain, which has multiple advantages like decreased computational complexity as well as the preservation of the phase of the signals. As common in the source separation literature, we assume that the mix $m \in \mathcal{X}$ is a linear combination of $N$ source signals $s_1, \ldots, s_N \in \mathcal{X}$

$$m = g(s) = \sum_{k=1}^{N} \alpha_k s_k.$$  (1)

As seen in [3], we take a probabilistic approach in order to solve the source separation problem. According to Bayes rule we can compute the posterior distribution using

$$p(s|m) = \frac{p(s)p(m|s)}{p(m)},$$  (2)

where we use a Gaussian approximation $p(m) = \mathcal{N}(g(s), \gamma^2 I)$ with noise parameter $\gamma$. Stochastic Gradient Langevin Dynamics (SGLD) [21] enables us to sample from the posterior distribution $p(s|m)$ without the need for evaluating $p(m)$. A new sample of the source signals can be generated by

$$s_{t+1} = s_t + \eta \nabla_s \left( \log p(s') + \frac{1}{2\gamma^2} \|m - g(s')\|^2 \right) + \sqrt{2\eta} \epsilon,$$  (3)

where $\epsilon = \mathcal{N}(0, 1)$. Last, we assume that the prior of the source signals factorizes as follows

$$p(s) = p(s_1, \ldots, s_N) = p(s_1) \ldots p(s_N).$$  (4)

We choose to parameterize the priors $p(s_1) \ldots p(s_N)$ with deep generative models. This allows us to easily compute $\nabla_s \log p(s')$ using the automatic differentiation tools from [12]. As noted in [3],
in order to successfully recover the original source signals using Equation 3, we require the priors \( p(s_1) \ldots p(s_N) \) to be discriminative as well as sufficiently smooth. After our initial experiments failed, we observed that the learned prior distributions are either discriminative and extremely peaked or smooth and non-discriminative. In the following, we quantify this behavior for two types of deep generative models on two audio datasets.

2 Method: Modeling raw audio with generative models

Deep learning models as used for image applications are unsuitable for raw audio signals (signals in the time-domain). Digital audio is sampled at high sample rates, commonly 16kHz up to 44kHz. The features of interest lie at scales of strongly different magnitudes. Therefore, generative models need to model the complete range of frequencies containing high-frequency features like timbre and slow frequency features like song structure. We will be using two types of generative models to learn the likelihood \( p(s) \) of audio signals, namely, WaveNet [18] and FloWaveNet [5].

**WaveNet** The WaveNet is an autoregressive generative model for raw audio. The generation of a new sample \( x_t \) is conditioned on all previous samples \( x_1, \ldots, x_{t-1} \). The likelihood of the entire signal is given by

\[
p(x) = \prod_{t=1}^{T} p(x_t | x_1, \ldots, x_{t-1}).
\]  

After training, starting from a single sample the model is able to iteratively generate a coherent time signal. The distribution \( p(x_t | x_1, \ldots, x_{t-1}) \) is modeled as a multinomial logistic regression, therefore the continuous signal \( x \) is discretized using a so-called \( \mu \)-law encoding [18], resulting in 256 classes. Due to the autoregressive nature of the Wavenet the generative process is difficult to parallelize and generally slow.

The WaveNet adapts the PixelCNN [19] architecture to the audio domain. It is a fully-convolutional network where dilated causal convolutions [23] are used. Using a stack of dilated convolutions increases the receptive field of the deep features without increasing the computational complexity. Further, the convolutions are gated [2] and the output is constructed as the sum of skip connections from each layer. The skip connections fuse information from multiple time-scales.

**FloWaveNet** An alternative to using an autoregressive model to model \( p(x) \) are normalizing flows [15]. Normalizing flows are a class of exact likelihood models, which are amenable to gradient-based optimisation and efficient in inference and sampling. A standard normalizing flow in continuous space, is based on the simple change of variables formula. Given an observed data variable \( x \in X \); a prior probability distribution \( p_Z(z) \) on a latent variable \( z \in Z \), and a differentiable, bijective function \( z = f(x) \), we can model a probability distribution \( p_X(x) \) as

\[
p_X(x) = p_Z(f(x)) \left| \det \left( \frac{\partial f(x)}{\partial x} \right) \right|.
\]  

Since Equation 6 requires the computation of a determinant, a special architecture is necessary to reduce the computational cost. One such architecture was introduced in [11]. [11] is using so-called coupling layers in which an input \( x \) is masked into two equally sized parts \( x_a \) and \( x_b \). One part is fed into a function to provide the weights of an affine transformation \( (s(\cdot), t(\cdot)) \) of the other

\[
\hat{x}_b = s(x_a) \cdot x_b + t(x_a).
\]  

The resulting Jacobian is a triangular matrix, whose determinant can be easily computed. There exist multiple approaches to combine the dilated convolutions used in the WaveNet and normalizing flows. In the case of FloWaveNet [5] a WaveNet encoder is used to predict the weights of the affine transformation within every coupling layer. This architecture enables fast, parallel generation of audio samples.

2
3 Experiments

3.1 Datasets

**Toy data**  We create a toy-like dataset consisting of four distinct waveforms, as shown in Figure 1: a sine, a sawtooth, a square and a triangle wave. We generate the waveforms using a sampling frequency of 16kHz. For each waveform we sample a random frequency $f \in [27Hz, 4186Hz]$, a random amplitude $A \in [0.8, 1.0]$ and a random phase $\phi \in [0, 2\pi]$.

$$m = \frac{1}{4} \sum_i s_i \quad (8)$$

The mix $m$ is equal to the mean of the four source signals. We create 5000 mixes of one second length for training and a test set of 1500 mixes.

**musdb18**  The musdb18 [13] dataset, created for the 2018 Signal Separation Evaluation Campaign [16], is a benchmark dataset used to evaluate audio source separation algorithms. The dataset consists of 150 songs from various artists and genres, split into train and test sets sized 100 and 50, respectively. For each song, the full mix $m$ and four separate sources $s_1, s_2, s_3, s_4$ are given: drums, bass, vocals and others. The others source contains any instruments not contained in the first three. Note that the mix $m$ does not strictly follow Equation 8 since it involves audio effects like compression. The song files are encoded with a sampling rate of 44.1kHz which we down-sample to 16kHz. We extract 150 fixed-length frames of one second from each song.

3.2 Discriminative power of the priors

In the following, we train a separate WaveNet and FloWaveNet model for each signal source type of both datasets, in total eight separate generative models for each dataset. Details about architecture choices and training schedules can be found in the Appendix 5.

Using Langevin dynamics for separation we optimize the separated source frames under each prior model. During training of the deep generative priors, they explicitly contract the density for positive, in-class examples. During separation, the priors encounter negative out-of-distribution samples for the first time. To be useful for separation, the priors have to give a low likelihood to samples from the other possible sources.
Following the results in [8], we test the out-of-distribution detection performance of the deep generative priors by evaluating the mean log-likelihood of the test data of each source under each source prior. In Figure 3 we show that only for the FloWaveNet model trained with the Toy data behaves as anticipated. The in-class samples have a high likelihood while for all prior models the out-of-class samples have a low likelihood.

For the musdb18 dataset, neither the WaveNet- nor the FloWaveNet-based priors can discriminate between in-class and out-of-class samples. We hypothesize that this stems from the fact that the real musical data is severely more complicated compared than the Toy data. The in-class variability of the real sources is that high, that the models are not able to learn a distribution that would be discriminative. Note that the source other in musdb18 contains an undefined set of instruments making a model of those sounds in general impractical. But even if we are ignoring this subset neither prior model can discriminate the remaining source types. As a result, for the following experiments we focus on the Toy data dataset.

### 3.3 Smoothness of the learned distribution

In the case of the Toy dataset, only the FloWaveNet priors can distinguish between in-class and out-of-class signals. However, when we tried to use those priors for source separation as described in Section 3.2 we failed. We argue that one possible explanation is the peakedness of the learned prior distributions. The probability mass learned by the model is peaked at true samples but quickly decays with more disturbance of the input. The reason for this behavior is that all models are trained with noise-free samples of their respective signal sources (sine, square, saw, triangle).

As proposed in [3], we now approximate the noisy distribution $\log p_\sigma(x)$, which is the convolution of the noiseless distribution with a Gaussian with variance $\sigma$: $\log p_\sigma(x) \ast N(0, \sigma)$ by adding Gaussian noise with the same variance to the input. Figuratively speaking, the Gaussian noise in data space translates to Gaussian smoothing of the peak in the probability distribution of the data.

Instead of retraining the deep generative priors we fine-tune [22] the noise-free models used in Section 3.2. We follow [3] in evaluating the noise-conditional model at different levels of noise $\sigma \in \{0.01, 0.027, 0.077, 0.129, 0.359\}$. In Figure 4 we show the cross mean log-likelihood for increasing noise-conditionals for the Toy data dataset. We find that even with small levels of conditional noise added the discriminative power of the learned generative models decays significantly. While being smooth the noise-conditional distributions cannot be used for source separation as intended.

### 3.4 Random and constant inputs

Previous works [17][20][8] have pointed out that generative models tend to assign high likelihood values to constant inputs. We find the same holds true for generative priors trained on the Toy data.
dataset. Table 1 shows that for the noise-free model a constant zero input is highly likely, except under the square wave prior, which we assume stems from the square wave never having the value 0.0. When fine-tuning the model with a noise-conditioning of $\sigma = 0.359$ the constant zero input becomes less likely for the sources sine and triangle but more likely for saw and square. To test whether the noise-conditioning results in simple constant inputs being unlikely but pure noise input in return becoming likely we evaluate the likelihood of noise drawn from a zero-centered Gaussian with reasonable wide variance for both the noise-free and noise-conditional model.

In Table 1 we see that for the noise-free prior model, a high variance input noise sampled from $\mathcal{N}(0, 0.5)$ is highly unlikely. Evaluating the same input noise under the wider noise-conditioned prior model the input becomes more likely. For the sine and saw waveform the noise input is even as likely as a normal in-class input.

We read these results to support the previous interpretation that even a small amount of noise fine-tuning can have severe effects on the estimated density. The noise-free prior models have sharp likelihood peaks around true data, in which even small amounts of added noise are highly unlikely. The noise-conditioning of the flow models flattens these peaks in so far that noise and out-of-distribution samples become highly likely, even at small levels of noise-conditioning.

### 4 Conclusion

In this work, we show that contemporary generative models for modeling of audio signals also exhibit strong problems with out-of-distribution data as similarly described in [8] for models of image data. Our experiments reinforce a suspicion that was also experimentally found in prior work on image data. Current deep generative models do not learn a density that is discriminative against out-of-distribution samples. We show that in our case the models lose their ability to detect out-of-distribution samples when trained with additive noise which is added to smooth the learned densities. Therefore our work further demonstrates to be cautious when applying current flow-based models to data outside close bounds of their training distribution.

### References


