Maurice Frank

Unsupervised variational source separation with deep priors

Contents

Todo list

Proposal

Abstract

Research Question

Source separation is the task of finding a set of latent sources $\boldsymbol{s} = [\boldsymbol{s}_1, \dots \boldsymbol{s}_k, \dots, \boldsymbol{s}_N]^T \in \mathbb{R}^{N \times T}$ to an observed mix of those sources $\boldsymbol{m} \in \mathbb{R}^{1 \times T}$. The induced model proposes a mixing function $\boldsymbol{m} = f(\boldsymbol{s})$. The task is to find an approximate inverse model $g(\cdot)$ which retrieves $\boldsymbol{s}$:

$$\mathbf{m} = f(\boldsymbol{s}) \tag{1}$$

$$g(\boldsymbol{m}) \approxeq \boldsymbol{s} \tag{2}$$

In this learning setting *supervision* can happen in two ways: First the source signals are identified as being from class k [1]. Second the tuples $(\boldsymbol{m}, \boldsymbol{s})$ are supervised giving us examples of mixes and their corresponding sources.

[1] For the setting of music think of the classes being {*guitar*,*piano*,*voice*,...}

add other supverision: knowing k

1. Can we learn a source separation model $g(\cdot)$ by learning deep priors for the different source classes.
2. Can we reduce this to an unsupervised setting. Unsupervised relating to the missing pairings of sources and mixes.

Related works

In this chapter, we discuss previous research in supervised and semi-supervised source separation.

ICA

Deep Latent-Variable Models

For our process, we have observations from the data space $\boldsymbol{x} \in \mathcal{D}$ for which there exists an unknown data probability distribution $p^*(\mathcal{D})$. We collect a data set $\{\boldsymbol{x}_1 \dots \boldsymbol{x}_N\}$ with N samples. We introduce an approximate model with density[2] $p_{\boldsymbol{\theta}}(\mathcal{D})$ and model parameters $\boldsymbol{\theta}$. Learning or modeling means finding the values for $\boldsymbol{\theta}$ which will give the closest approximation of the true underlying process:

[2] We write density and distribution interchangeably to denote a probability function.

$$p_{\boldsymbol{\theta}}(\mathcal{D}) \approx p^*(\mathcal{D}) \tag{3}$$

The model $p_{\boldsymbol{\theta}}$ has to be complex enough to be able to fit the data density while little enough parameters to be learned. Every choice for the form of the model will *induce* biases[3] about what density we can model, even before we maximize a learning objective using the parameters $\boldsymbol{\theta}$.

[3] called *inductive biases*

In the following described models we assume the sampled data points $\boldsymbol{x}$ to be drawn from $\mathcal{D}$ *independent and identically distributed*[4]. Therefore we can write the data log-likelihood as:

[4] meaning the sample of one datum does not depend on the other data points

$$p_{\boldsymbol{\theta}}(\mathcal{D}) = \prod_{\boldsymbol{x} \in \mathcal{D}} p_{\boldsymbol{\theta}}(\boldsymbol{x}) \tag{4}$$

$$\log p_{\boldsymbol{\theta}}(\mathcal{D}) = \sum_{\boldsymbol{x} \in \mathcal{D}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) \tag{5}$$

The maximum likelihood estimation of our model parameters maximizes this objective.

To form a latent-variable model we introduce a *latent variable*[5]. The data likelihood now is the marginal density of the joint latent density:

[5] Latent variables are part of the directed graphical model but not observed.

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} \tag{6}$$

Typically we introduce a factorization of the joint. Most commonly and simplest:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z} \tag{7}$$

This corresponds to the graphical model in which $\boldsymbol{z}$ is generative parent node of the observed $\boldsymbol{x}$, see Figure 1. The density $p(\boldsymbol{z})$ is called the *prior distribution*.

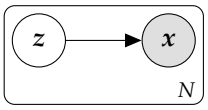


Figure 1: The graphical model with a introduced latent variable $\boldsymbol{z}$. Observed variables are shaded.

If the latent is small, discrete, it might be possible to directly marginalize over it. If for example, $\boldsymbol{z}$ is a discrete random variable and the conditional $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is a Gaussian distribution than the data model density $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ becomes a mixture-of-Gaussians, which we can directly estimate by maximum likelihood estimation of the data likelihood.

For more complicated models the data likelihood $p_\theta(x)$ as well as the model posterior $p_\theta(z|x)$ are intractable because of the integration over the latent z in Equation (7).

To formalize the search for an intractable posterior into a tractable optimization problem we follow the *variational principle*[6] which introduces an approximate posterior distribution $q_\phi(z|x)$, also called the *inference model*. Again the choice of the model here carries inductive biases as such that even in asymptotic expectation we can not obtain the true posterior.

[6] Michael I. Jordan et al. "An Introduction to Variational Methods for Graphical Models". In: *Machine Learning* 37.2 (1999), pp. 183–233.

Following the derivation in[7] we introduce the inference model into the data likelihood [8]:

[7] Diederik P. Kingma and Max Welling. "An Introduction to Variational Autoencoders". In: (2019). arXiv: 1906.02691 [cs, stat], p. 20.

[8] The first step is valid as q_θ is a valid density function and thus integrates to one.

$$\log p_\theta(x) = \mathbb{E}_{q_\theta(z|x)}\left[\log p_\theta(x)\right] \tag{8}$$

$$= \mathbb{E}_{q_\theta(z|x)}\left[\log \frac{p_\theta(x,z)}{p_\theta(z|x)}\right] \tag{9}$$

$$= \mathbb{E}_{q_\theta(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\frac{q_\phi(z|x)}{p_\theta(z|x)}\right] \tag{10}$$

$$= \mathbb{E}_{q_\theta(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\right] + \mathbb{E}_{q_\theta(z|x)}\left[\log \frac{q_\phi(z|x)}{p_\theta(z|x)}\right] \tag{11}$$

$$= \mathbb{E}_{q_\theta(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\right] + \mathbb{D}_{\mathrm{KL}}[q_\phi(z|x)\|p_\theta(z|x)] \tag{12}$$

Note that we separated the likelihood into two parts. The second part is the (positive) Kullback-Leibler divergence of the approximate posterior from the true intractable posterior. This unknown divergence states the 'correctness' of our approximation [9].

[9] More specifically the divergence marries two errors of our approximate model. First, it gives the error of our posterior estimation from the true posterior, by definition of divergence. Second, it specifies the error of our complete model likelihood from the marginal likelihood. This is called the *tightness* of the bound.

The first term is the *variational free energy* or *evidence lower bound* (ELBO):

explain free energy

$$\mathrm{ELBO}_{\theta,\phi}(x) = \mathbb{E}_{q_\theta(z|x)}\left[\log \frac{p_\theta(x,z)}{q_\phi(z|x)}\right] \tag{13}$$

We can introduce the same factorization as in Equation (7):

$$\mathrm{ELBO}_{\theta,\phi}(x) = \mathbb{E}_{q_\theta(z|x)}\left[\log \frac{p_\theta(x|z)p(z)}{q_\phi(z|x)}\right] \tag{14}$$

$$= \mathbb{E}_{q_\theta(z|x)}\left[\log \frac{p(z)}{q_\phi(z|x)}\right] + \mathbb{E}_{q_\theta(z|x)}\left[\log p_\theta(x|z)\right] \tag{15}$$

$$= -\mathbb{D}_{\mathrm{KL}}[q_\phi(z|x)\|p(z)] + \mathbb{E}_{q_\theta(z|x)}\left[\log p_\theta(x|z)\right] \tag{16}$$

Under this factorization, we separated the lower bound into two parts. First, the divergence of the approximate posterior from the latent prior distribution and second the data posterior likelihood from the latent [10].

[10] this will later be the reconstruction error. How well can we return to the data density from latent space

The optimization of the $\text{ELBO}_{\theta,\phi}$ allows us to jointly optimize the parameter sets θ and ϕ. The gradient with respect to θ can be estimated with an unbiased Monte Carlo estimate using data samples [11]. We can *not* though do the same for the variational parameters ϕ, as the expectation of the ELBO is over the approximate posterior which depends on ϕ. By a change of variable of the latent variable we can make this gradient tractable, the so called *reparameterization trick*.[12] We express the $z \sim q_\theta$ as an random sample from a unparametrized source of entropy ϵ and a parametrized transformation:

[11] $\nabla_\theta \text{ELBO}_{\theta,\phi} \approxeq \nabla_\theta \log p_\theta(x, z)$

[12] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: (2014). arXiv: 1312.6114 [cs, stat].

$$z = f_\eta(\epsilon) \tag{17}$$

For example for a Gaussian distribution we can express $z \sim \mathcal{N}(\mu, \sigma)$ as $z = \mu + \sigma \cdot \epsilon$ with $\epsilon \sim \mathcal{N}(0, 1)$ and $\eta = \{\mu, \sigma\}$.

The VAE framework

VAE [12][13]

The β-VAE[14] extends the VAE objective with an β hyperparameter in front of the KL divergence. The value β gives a constraint on the latent space controlling the capacity of it. Adapting β gives a trade-off between the reconstruction quality of the autoencoder and the simplicity of the latent representations[14]. Using such a constraint is similar to the use of in the information bottleneck.[15]

[13] Danilo Jimenez Rezende et al. "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: (2014). arXiv: 1401.4082 [cs, stat].

[14] Irina Higgins et al. "Beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". In: (2016).

[15] Christopher P. Burgess et al. "Understanding Disentangling in Beta-VAE". In: (2018). arXiv: 1804.03599 [cs, stat].

Flow based models

Another class of common deep latent models is based on *normalizing* flows.[16] A normalizing flow is a function $f(x)$ that maps the input density to a fixed, prescribed density $p(\epsilon) = p(f(x))$, in that normalizing the density [17]. They use a flow for the approximate posterior $q_\phi(z|x)$. Again this is commonly set to be a factorized Gaussian distribution.

For a finite normalizing flow, we consider a chain of invertible, smooth mappings.

NICE[18] - volume preserving transformations - coupling layer - triangular shape

Normalizing Flow[19]

RealNVP[20] builds on top of NICE creating a more general, non-volume preserving, normalizing flow.

[16] Esteban Tabak and Cristina V. Turner. "A family of nonparametric density estimation algorithms". In: *Communications on Pure and Applied Mathematics* 66.2 (2013), pp. 145–164.

[17] The extreme of this idea is, of course, an infinitesimal, continuous-time flow with a velocity field.

[18] Laurent Dinh et al. "NICE: Non-Linear Independent Components Estimation". In: (2015). arXiv: 1410.8516 [cs].

[19] Danilo Jimenez Rezende and Shakir Mohamed. "Variational Inference with Normalizing Flows". In: (2016). arXiv: 1505.05770 [cs, stat].

[20] Laurent Dinh et al. "Density Estimation Using Real NVP". In: (2017). arXiv: 1605.08803 [cs, stat].

$$y_{1:d} = x_{1:d} \tag{18}$$

$$y_{d+1:D} = x_{d+1:D} \odot \exp\left(s(x_{1:d})\right) + t(x_{1:d}) \tag{19}$$

$$\frac{\partial y}{\partial x^T} = \begin{bmatrix} \mathbb{1}_d & 0 \\ \frac{\partial y_{d+1:D}}{\partial x_{1:d}^T} & \operatorname{diag}\left(\exp\left(s(x_{1:d})\right)\right) \end{bmatrix} \tag{20}$$

Glow[21] extended the RealNVP by introducing invertible 1×1-convolutions. Instead of having fixed masks and permutations for the computations of the affine parameters in the coupling layer, Glow learns a rotation matrix which mixes the channels. After mixing the input can always be split into the same two parts for the affine transformation. Further, the authors showed that training can be helped by initializing the last layer of each affine parameter network with zeros. This ensures that at the beginning without weight update each coupling layer behaves as an identity.

[21] Diederik P. Kingma and Prafulla Dhariwal. "Glow: Generative Flow with Invertible 1x1 Convolutions". In: (2018). arXiv: 1807.03039 [cs, stat].

Modelling raw audio

Deep learning models as used for image applications are unsuitable for raw audio signals (*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. Recognizing phase, frequency of a wave might require features at low ms intervals but modelling of speech or music features happens at the scale of seconds to minutes. As such a generative model for this domain has to model at these different scales.

The WaveNet[22] introduced an autoregressive generative model for raw audio. It is build upon the similar PixelCNN[23] but adapted for the audio domain. The WaveNet accomplishes this by using dilated causal convolutions a common tool in signal processing.[24] A dilated convolution uses a kernel with an inner stride. Using a stack of dilated convolutions increases the receptive field of the deep features without increasing the computational complexity, see Figure 2. Further, the convolutions are gated and the output is constructed from skip connections, refer to **??** . A gated feature, as known from the LSTM,[25] computes two outputs: one put through an sigmoid $\sigma(\cdot)$ activation and one through an $\tanh(\cdot)$ activation. The idea being that the sigmoid (with an output range of $[0, 1]$) regulates the amount of information, thereby gating it, while the tanh (with a range of $[-1, 1]$) gives the magnitude of the feature. The output of the WaveNet is the sum of outflowing skip connections added after each (gated) hidden convolution. This helps fusing information from multiple time-scales (*low-level* to *high-level*). The original authors tested the model on multiple audio generation tasks. For this, they formulated the reconstruction objective as a multi-class recognition problem. Encoding the sound files with μ-law encoding,[26] discretizes the range $[-1, 1]$ to allow a set of μtargets. Sound generation with a WaveNet is slow as the autoregressiveness

[22] Aäron van den Oord et al. "WaveNet: A Generative Model for Raw Audio". In: (2016). arXiv: 1609.03499 [cs].

[23] Aäron van den Oord et al. "Conditional Image Generation with PixelCNN Decoders". In: (2016). arXiv: 1606.05328 [cs],

[24] P. Dutilleux. "An Implementation of the Algorithme à Trous to Compute the Wavelet Transform". In: *Wavelets*. Ed. by Jean-Michel Combes et al. Inverse Problems and Theoretical Imaging. Berlin, Heidelberg: Springer, 1990, pp. 298–304.

[25] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[26] *Recommendation G. 711. Pulse Code Modulation (PCM) of Voice Frequencies*. 1988.

requires the generation value by value[27]. This can be alleviated by keeping intermediate hidden activations cached.[28] The WaveNet can be conditioned by adding the weighted conditionals in the gate and feature activations of the gated convolutions.[29]

NSynth[30]

In[31]

FloWaveNet[32]

Test quote

Source separation

WaveNet for Speech denoising[33]

WaveNet-VAE unsupervised speech rep learning[34]

[35] were the first to use an U-Net architecture (see Figure 3) for source separation. They used a convolutional U-Net on spectrograms of musical data to extract the singing voice. Input to the network is the spectral magnitude and the output prediction is an equally-sized masked. The voice signal reconstruction is then done by multiplying the input magnitude with the predicted mask and using the original phase information from the mix, unaltered. The training objective is the L_1 loss between the masked input spectrogram and the target signal.

The Wave-U-Net[36] brings this idea into the time-domain. The downstreaming and upstreaming layers are replaced with WaveNet style 1D convolutional layers. Further here the model is predicting multiple source signals.

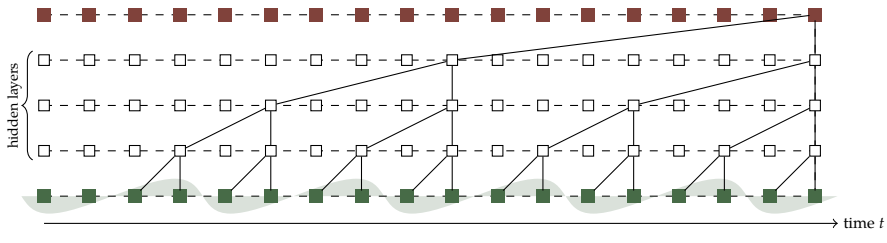
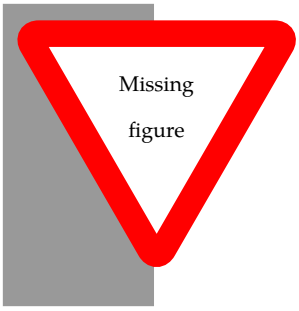


Figure 2: An example of how dilated convolutions are used in the WaveNet. We see three hidden layers with each a kernel size of two. By using the dilations the prediction of the new output element has a receptive field of 18. This convolution is *causal* as the prediction depends only on previous input values.

[27] Why are they doing that then? The WaveNet setting is generating waveforms, by giving the previously generated values as the input and conditioning the process on target classes. Therefore the generation has to happen value-by-value.

[28] Tom Le Paine et al. "Fast Wavenet Generation Algorithm". In: (2016). arXiv: 1611.09482 [cs].

[29] Van den Oord et al., "Conditional Image Generation with PixelCNN Decoders".



Show WaveNet hidden layer

explain PixelCNN++ even if I don't use it know?

[30] Nal Kalchbrenner et al. "Efficient Neural Audio Synthesis". In: (2018). arXiv: 1802.08435 [cs, eess].

[31] Ryan Prenger et al. "WaveGlow: A Flow-Based Generative Network for Speech Synthesis". In: (2018). arXiv: 1811.00002 [cs, eess, stat].

[32] Sungwon Kim et al. "FloWaveNet : A Generative Flow for Raw Audio". In: (2019). arXiv: 1811.02155 [cs, eess].

[33] Dario Rethage et al. "A Wavenet for Speech Denoising". In: (2018). arXiv: 1706.07162 [cs].

[34] Jan Chorowski et al. "Unsupervised Speech Representation Learning Using WaveNet Autoencoders". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27.12 (2019), pp. 2041–2053. arXiv: 1901.08810.

[35] Andreas Jansson et al. "Singing Voice Separation with Deep U-Net Convolutional Networks". In: *ISMIR*. 2017.

[36] Daniel Stoller et al. "Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation" (Paris, France), 2018.

In[37] the authors adapted a time-domain WaveNet for musical source separation. The network is non-causal [38] The WaveNet directly outputs the separated sources and is trained fully supervised. They show this simple setup performing well against spectrogram based baselines. Also, the experiments show a higher performance with a network that is deeper but less wide [39]

Demucs[40] is another extension building on the U-Net idea. Having a similar structure to the Wave-U-Net they introduce a bidirectional LSTM at the bottleneck.[41] The LSTM is responsible for keeping long-term temporal informational by running over the high-level latent along the time dimension. They can outperform spectrogram based approaches.

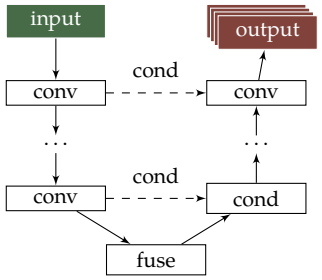


Figure 3: The U-Net

Methodology

Theory

$$\log p_{\boldsymbol{\theta}}(\boldsymbol{m}) = \int \left[\prod_k^N q_{\boldsymbol{\phi}_k}(\boldsymbol{s}_k|\boldsymbol{m})\right] \cdot \log p(\boldsymbol{m})\, d^N\boldsymbol{s} \tag{21}$$

$$= \int \left[\prod_k^N q_{\boldsymbol{\phi}_k}(\boldsymbol{s}_k|\boldsymbol{m})\right] \cdot \log \frac{p(\boldsymbol{m}, \boldsymbol{s}_1, \ldots, \boldsymbol{s}_N)}{p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_N|\boldsymbol{m})}\, d^N\boldsymbol{s} \tag{22}$$

$$= \int \left[\prod_k^N q_{\boldsymbol{\phi}_k}(\boldsymbol{s}_k|\boldsymbol{m})\right] \cdot \left[\log \frac{p(\boldsymbol{m}|\boldsymbol{s}_1, \ldots, \boldsymbol{s}_N) \cdot \prod_k^N p(\boldsymbol{s}_k)}{\prod_k^N q_{\boldsymbol{\phi}_k}(\boldsymbol{s}_k|\boldsymbol{m})} + \log \frac{\prod_k^N q_{\boldsymbol{\phi}_k}(\boldsymbol{s}_k|\boldsymbol{m})}{p(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_N|\boldsymbol{m})}\right] d^N\boldsymbol{s} \tag{23}$$

Datasets

ToyData

MusDB

Planning

[37] Francesc Lluís et al. "End-to-End Music Source Separation: Is It Possible in the Waveform Domain?" In: (2019). arXiv: 1810.12187 [cs, eess].

[38] In this setting the WaveNet can be non-causal because the prediction happens from the given mix and is not autoregressive.

[39] Deepness of a network refers to the number of hidden layers. Wideness refers to the number of kernels/features of each of these hidden layers. Making the WaveNet deeper significantly increases its receptive field.

[40] Alexandre Défossez et al. "Demucs: Deep Extractor for Music Sources with Extra Unlabeled Data Remixed". In: (2019). arXiv: 1909.01174 [cs, eess, stat].

[41] Alexandre Defossez et al. "SING: Symbol-to-Instrument Neural Generator". In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio et al. Curran Associates, Inc., 2018, pp. 9041–9051.

Bibliography

Burgess, Christopher P., Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. "Understanding Disentangling in Beta-VAE". In: (2018). arXiv: 1804.03599 [cs, stat] (cit. on p. 10).

Chorowski, Jan, Ron J. Weiss, Samy Bengio, and Aäron van den Oord. "Unsupervised Speech Representation Learning Using WaveNet Autoencoders". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27.12 (2019), pp. 2041–2053. arXiv: 1901.08810 (cit. on p. 12).

Défossez, Alexandre, Nicolas Usunier, Léon Bottou, and Francis Bach. "Demucs: Deep Extractor for Music Sources with Extra Unlabeled Data Remixed". In: (2019). arXiv: 1909.01174 [cs, eess, stat] (cit. on p. 13).

Defossez, Alexandre, Neil Zeghidour, Nicolas Usunier, Leon Bottou, and Francis Bach. "SING: Symbol-to-Instrument Neural Generator". In: *Advances in Neural Information Processing Systems 31*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Curran Associates, Inc., 2018, pp. 9041–9051 (cit. on p. 13).

Dinh, Laurent, David Krueger, and Yoshua Bengio. "NICE: Non-Linear Independent Components Estimation". In: (2015). arXiv: 1410.8516 [cs] (cit. on p. 10).

Dinh, Laurent, Jascha Sohl-Dickstein, and Samy Bengio. "Density Estimation Using Real NVP". In: (2017). arXiv: 1605.08803 [cs, stat] (cit. on p. 10).

Dutilleux, P. "An Implementation of the Algorithme à Trous to Compute the Wavelet Transform". In: *Wavelets*. Ed. by Jean-Michel Combes, Alexander Grossmann, and Philippe Tchamitchian. Inverse Problems and Theoretical Imaging. Berlin, Heidelberg: Springer, 1990, pp. 298–304 (cit. on p. 11).

Higgins, Irina, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. "Beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". In: (2016) (cit. on p. 10).

Hochreiter, Sepp and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780 (cit. on p. 11).

Jansson, Andreas, Eric J. Humphrey, Nicola Montecchio, Rachel M. Bittner, Aparna Kumar, and Tillman Weyde. "Singing Voice Separation with Deep U-Net Convolutional Networks". In: *ISMIR*. 2017 (cit. on p. 12).

Jordan, Michael I., Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. "An Introduction to Variational Methods for Graphical Models". In: *Machine Learning* 37.2 (1999), pp. 183–233 (cit. on p. 9).

Kalchbrenner, Nal, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aäron van den Oord, Sander Diele-

man, and Koray Kavukcuoglu. “Efficient Neural Audio Synthesis”. In: (2018). arXiv: 1802.08435 [cs, eess] (cit. on p. 12).

Kim, Sungwon, Sang-gil Lee, Jongyoon Song, Jaehyeon Kim, and Sungroh Yoon. “FloWaveNet : A Generative Flow for Raw Audio”. In: (2019). arXiv: 1811.02155 [cs, eess] (cit. on p. 12).

Kingma, Diederik P. and Prafulla Dhariwal. “Glow: Generative Flow with Invertible 1x1 Convolutions”. In: (2018). arXiv: 1807.03039 [cs, stat] (cit. on p. 11).

Kingma, Diederik P. and Max Welling. “Auto-Encoding Variational Bayes”. In: (2014). arXiv: 1312.6114 [cs, stat] (cit. on p. 10).

Kingma, Diederik P. and Max Welling. “An Introduction to Variational Autoencoders”. In: (2019). arXiv: 1906.02691 [cs, stat] (cit. on p. 9).

Lluís, Francesc, Jordi Pons, and Xavier Serra. “End-to-End Music Source Separation: Is It Possible in the Waveform Domain?” In: (2019). arXiv: 1810.12187 [cs, eess] (cit. on p. 13).

Paine, Tom Le, Pooya Khorrami, Shiyu Chang, Yang Zhang, Prajit Ramachandran, Mark A. Hasegawa-Johnson, and Thomas S. Huang. “Fast Wavenet Generation Algorithm”. In: (2016). arXiv: 1611.09482 [cs] (cit. on p. 12).

Prenger, Ryan, Rafael Valle, and Bryan Catanzaro. “WaveGlow: A Flow-Based Generative Network for Speech Synthesis”. In: (2018). arXiv: 1811.00002 [cs, eess, stat] (cit. on p. 12).

Recommendation G. 711. Pulse Code Modulation (PCM) of Voice Frequencies. 1988 (cit. on p. 11).

Rethage, Dario, Jordi Pons, and Xavier Serra. “A Wavenet for Speech Denoising”. In: (2018). arXiv: 1706.07162 [cs] (cit. on p. 12).

Rezende, Danilo Jimenez and Shakir Mohamed. “Variational Inference with Normalizing Flows”. In: (2016). arXiv: 1505.05770 [cs, stat] (cit. on p. 10).

Rezende, Danilo Jimenez, Shakir Mohamed, and Daan Wierstra. “Stochastic Backpropagation and Approximate Inference in Deep Generative Models”. In: (2014). arXiv: 1401.4082 [cs, stat] (cit. on p. 10).

Stoller, Daniel, Sebastian Ewert, and Simon Dixon. “Wave-U-Net: A Multi-Scale Neural Network for End-to-End Audio Source Separation” (Paris, France). 2018 (cit. on p. 12).

Tabak, Esteban and Cristina V. Turner. “A family of nonparametric density estimation algorithms”. In: *Communications on Pure and Applied Mathematics* 66.2 (2013), pp. 145–164 (cit. on p. 10).

Van den Oord, Aäron, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. “WaveNet: A Generative Model for Raw Audio”. In: (2016). arXiv: 1609.03499 [cs] (cit. on p. 11).

Van den Oord, Aäron, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. “Conditional Image Generation with PixelCNN Decoders”. In: (2016). arXiv: 1606.05328 [cs] (cit. on pp. 11, 12).