A multi-objective approach for sustainable generative audio models

Constance Douwes, Philippe Esling, Jean-Pierre Briot

A MULTI-OBJECTIVE APPROACH FOR SUSTAINABLE GENERATIVE AUDIO MODELS

Constance Douwes
IRCAM, Sorbonne Université, CNRS
UMR 9912 F-75004 Paris, France
douwes@ircam.fr

Philippe Esling
IRCAM, Sorbonne Université, CNRS
UMR 9912 F-75004 Paris, France
esling@ircam.fr

Jean-Pierre Briot
Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
UNIRIO, Rio de Janeiro, RJ 22290-250, Brazil
jean-pierre.briot@lip6.fr

July 7, 2021

ABSTRACT

In recent years, the deep learning community has largely focused on the accuracy of deep generative models, resulting in impressive improvements in several research fields. However, this scientific race for quality comes at a tremendous computational cost, which incurs vast energy consumption and greenhouse gas emissions. If the current exponential growth of computational consumption persists, Artificial Intelligence (AI) will sadly become a considerable contributor to global warming.

At the heart of this problem are the measures that we use as a scientific community to evaluate our work. Currently, researchers in the field of AI judge scientific works mostly based on the improvement in accuracy, log-likelihood, reconstruction or opinion scores, all of which entirely obliterates the actual computational cost of generative models.

In this paper, we introduce the idea of relying on a *multi-objective measure* based on Pareto optimality, which simultaneously integrates the models accuracy, as well as the environmental impact of their training. By applying this measure on the current state-of-the-art in generative audio models, we show that this measure drastically changes the perceived significance of the results in the field, encouraging optimal training techniques and resource allocation. We hope that this type of measure will be widely adopted, in order to help the community to better evaluate the significance of their work, while bringing computational cost – and *in fine* carbon emissions – in the spotlight of AI research.

1 Introduction

The motivation of this work comes from the following observation: between 2012 and 2018, the amount of computation used in deep learning grew by a factor of *300,000* Dario and Danny [2018]. This exponential growth might have permitted to achieve impressive results across a wide variety of tasks, but it also strongly increased the demand for energy production, responsible for approximately 35% of total greenhouse gas emissions in 2010. If this trend continues, it is fairly logical to predict that deep learning will be a significant contributor to climate change.

Most of the recent advances produced by deep approaches rely on a significant increase in terms of both size and complexity Hernandez and Brown [2020], as well as an ever-growing number of training examples. Hence, such improvements are often only permitted by a concomitant increase in power consumption Thompson et al. [2020] and, thus, carbon emission Strubell et al. [2020a]. In the audio synthesis domain, deep generative models have reached an unprecedented quality for waveform synthesis.They are used routinely for speech synthesis in assistant such as Apple Siri or Amazon Alexia. However, researchers concentrate on high-quality real-time raw waveform synthesis which is by far the largest data we could use to perform synthesis. It requires handling complex temporal structures

at both local and global scales; therefore, models are complex and computationally expensive, with either enormous recurrent neural cells, or big kernel convolutions (or both). The disparity of proposed models in the literature and the training time needed for them to converge questions the real effectiveness with regards to the quality of the generated results, and what could be the best compromise in terms of energy and environment. Moreover, research institutes and individuals can lack sufficient resources, due to the demand of countless types of specialized hardware (GPUs, TPUs), often running continuously for several days and even up to weeks. Hence, obtaining a quality similar to that of state-of-the-art models is becoming an unattainable goal, both financially and ecologically Schwartz et al. [2019].

Generally speaking, the absence of energy-based criteria for generative models falls within the broader lack of suitable evaluation methods, notably for assessing the quality of the generated content Theis et al. [2016]. In Figure 1, we display our analysis of the distribution of different evaluation metrics used in twenty-five state-of-the-art neural audio synthesis research papers. We can clearly see that current researches are more focused on measures of generation quality, rather than measures of algorithmic performance when evaluating and comparing models. Energy consumption, in that field, is never taken into account, neither for training nor sample generation. Some studies do mention the training time per iteration (*DeepVoice* Arik et al. [2017]) and the number of generated samples per second (*WaveRNN* Kalchbrenner et al. [2018]). However, measuring the precise energy consumption of a given model is a complex endeavor García-Martín et al. [2019], which remains mostly neglected. The new application domain of *green computing* aims to address this kind of issues. This aspect is a novel field in deep learning research, already emerging in some communities as natural language processing (NLP) Strubell et al. [2020b].

Generative audio models are promising advances for speech synthesis and music production. As a reminder, We also question the possibility to embed such models and hope our approach will allow preserving the battery lifetime by ranking the number of parameters at the same level as gains in quality.

Figure 1: Distribution of commonly-used measures to compare and evaluate generative audio models. In purple (left) those that refer to the quality of the generated samples, and in green (right) those that refer to their algorithmic complexity and performances.

In this article, we propose a new method to evaluate both accuracy (or quality) and energy efficiency of generative models. First, we present estimations of training costs in terms of CO_2 emissions for all state-of-the-art models for which we had enough training details among the twenty-five used in Figure 1: *SampleRNN* Mehri et al. [2019], *SING* Défossez et al. [2018], *WaveGAN* Donahue et al. [2019], *GANSynth* Engel et al. [2019] and *FloWaveNet* Kim et al. [2019]. We then propose the use of a multi-objective Pareto optimality criterion to provide fair comparisons regarding both quality and energy efficiency when publishing new models. We compute a subjective score for quality, and present two Pareto fronts, one for the training based on our CO_2 estimation, and one for the inference based on the number of parameters.

2 State-of-the-art

2.1 Neural audio synthesis

Audio synthesis has been a field of interest for over a century now, opening interesting doors for both musicians and scientists Briot [2020]. It can be defined as the process of generating sound, using electronic hardware or software. The expression "Neural audio synthesis" refers to audio synthesis performed by neural networks. It holds the promise of speeding-up human-computer interactions, increasing performance and expressivity, enabling unprecedented accuracy in statistical modeling tasks based on statistical modeling, and offering new tools for computational creativity Esling and Devis [2020].

Audio data can be expressed in many different ways. We count three categories with different levels of abstraction: symbolic, time-frequency representation (spectrograms) and waveform. Although spectrograms have historically been the most commonly used representation, they still lack in audio quality when applied to real-time synthesis especially due to the phase reconstruction issue. Thus, working directly on waveform could both improve generation times (by removing any form of post-processing), but also improve the quality of generated results.

2.2 Deep generative audio models

Deep generative models are a flourishing class of machine learning approaches, which deal with learning to generate novel data based on the observation of existing examples. Given training data points $\mathbf{x}$ following an unknown probability distribution $p(\mathbf{x})$, generative models aims to learn a parametric distribution $p_\theta(\mathbf{x})$ from a model family that best approximates $p(\mathbf{x})$, by iteratively changing model parameters θ. Several methods exist to address this, that we can split in four categories : *auto-regressive models*, *Variational Auto-Encoders* (VAE) Kingma and Welling [2014], *Generative Adversarial Networks* (GAN) Goodfellow et al. [2014] and *Flow-based models*Rezende and Mohamed [2015].

Auto-regressive models try to model examples $\mathbf{x} = x_{1...T}$ by making the assumption that each dimension x_t is only dependent on the previous ones:

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

Following this formulation, *WaveNet* van den Oord et al. [2016] and *SampleRNN* Mehri et al. [2019] have tackled direct waveform learning and generation. Unfortunately, these methods are based on heavy architectures whose computational complexity require humongous energy, both for training and inference. Furthermore, they also provide almost no direct control on the generative process. Some approaches use *VAEs* Esling et al. [2018] that learn a latent space providing a low-dimensional representation of the data while remaining rather simple and fast to train. However, the generated samples tend to be slightly blurry compared to recent adversarial networks, such as *WaveGan* Donahue et al. [2019] or *GANSynth* Engel et al. [2019]. These show impressive reconstruction abilities but lack latent expressivity and are difficult to optimize due to unstable training dynamics. The recently proposed *Normalizing Flows* (NF) allow to model highly complex distributions in the latent space and already yield remarkable results such as the *FlowSynth* Esling et al. [2020] or *FloWaveNet* Kim et al. [2019] models.

2.3 Measures of energy efficiency

First, we present general notions surrounding energy and power measurement in order to clarify these concepts. For the sake of brevity, we avoid going into too much detail, but refer interested readers to García-Martín et al. [2019] for additional explanations. The energy E (in Joules) is defined as the effort to perform a task during a certain period of time T (in seconds). This can be expressed as the integral of the instantaneous power $P(t)$ during that period as

$$E = \int_0^T P(t)dt \tag{2}$$

The resulting average power (in Watts) is defined as

$$P_{avg} = \frac{E}{T} \tag{3}$$

Generally, the goal of *energy efficiency* is to reduce the amount of energy required to perform the same task. In machine learning contexts, we consider two types of energy efficiency measures: the amount of energy required to train a model (until convergence), and the amount of energy required by the model for inference steps (generating a sample in the case of audio synthesis). However, measuring the energy consumption of any kind of computer program is already a challenging task, since there are many variables involved (e.g. cache hits, cache misses, DRAM accesses).

To quantify the environmental cost of training deep neural networks models for NLP, Strubell et al. Strubell et al. [2020b] decided to sample GPU, CPU and DRAM power consumption, respectively named p_g, p_c, and p_r, using the NVIDIA System Management Interface and the Intel's Running Average Power Limit. The sum of these three components is then multiplied by the *Power Usage Effectiveness* (PUE) coefficient, which estimates additional energies required to sustain the computing infrastructure (mainly cooling). They relied on a PUE coefficient of 1.58 as it is the 2018 global average for data centers, and end up with the following formula for the total average power

$$p_t = \text{PUE} \cdot (p_c + p_r + gp_g) \tag{4}$$

with g the total number of the GPUs used for training. The energy consumed is obtained as the multiplication of this total average power by the training time in seconds according to Equation (3). A popular metric for energy measurement is the kiloWatt-hour (kWh). As its name suggests, it is the multiplication of the power in kilo-Watts by the time in hours, given that 1 kWh = 3600 kJ. Finally, to link kilowatt-hour and CO_2 equivalent, Strubell et al. use the carbon emission *intensity factor* (in $kgCO_2eq/kWh$). This factor is location-dependent, but can be captured in real-time thanks to the online electricity map[1].

Lacoste et al. Lacoste et al. [2019] led to a simpler online tool called the "Machine Learning Impact Calculator[2]" that provides an approximation of the carbon emission required for training a model, considering the location of the servers, the total training time, and the hardware on which the training takes place. Very recently, Wolff Anthony et al. Wolff Anthony et al. [2020] developed an open-source tool written in Python called "Carbontracker", which tracks and predicts carbon emissions produced for training deep leaning models. This provides a more accurate estimation while being user-friendly.

Another common measure of efficiency is the total number of model parameters Engel et al. [2019], Vasquez and Lewis [2019] as it is quite easy to determine and usually directly correlated with computational complexity. Unlike aforementioned measures, this one is hardware- and location- independent. A very recent and successful approach in audio generation precisely attempted to reduce this number of parameters by using the *lottery ticket hypothesis* (Esling et al. [2020]). Lighter models require less memory space (especially crucial for embedded devices) but also incur less energy consumption. Nonetheless, the number of parameters does not accurately reflect power consumption as some operations consume more than others. Hence, the best way to alleviate that issue is to consider the number of Floating Point Operations (FPO) Schwartz et al. [2019] of a model.

2.4 Pareto optimization

Pareto optimization is a branch of mathematical optimization problems involving several conflicting objectives to be optimized simultaneously. This notion is used when it is impossible to improve one objective without degrading another. Formally, considering a multi-objective optimization problem

$$\min_{x \in X}(f_1(x), f_2(x), \ldots, f_k(x)) \tag{5}$$

with $k \geq 2$ the number of objective functions and x the decision vector in the feasible set X. $f : X \to \mathbb{R}^k, f(x) = (f_1(x), \ldots, f_k(x))^T$ is the vector-valued objective function to be minimized.

A feasible solution $x_a \in X$ is said to *dominate* another feasible solution $x_b \in X$, notated $x_a \prec x_b$, if :

- $\forall i \in \{1, \ldots, k\}, f_i(x_a) \leq f_i(x_b)$
- $\exists j \in \{1, \ldots, k\}, f_j(x_a) < f_j(x_b)$

A solution $x^* \in X$ is a *Pareto optimal point* and $f(x^*)$ is a Pareto optimal objective vector if there does not exist $\hat{x}$ such that $\hat{x} \prec x^*$. The set of all these optimal objective vectors is called the *Pareto front*. An example for $k = 2$ is presented in Figure 2, note that the minimization problems are transposable to maximization problems.

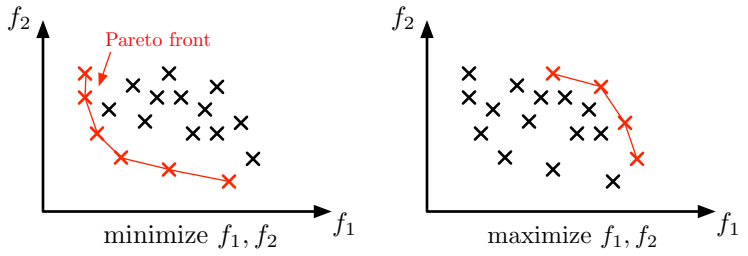

Figure 2: Example of two Pareto fronts (in red). Crosses represent feasible choices, red ones are Pareto optimal solutions while black ones are dominated by at least one Pareto optimum.

[1] `https://www.electricitymap.org/map`
[2] `https://mlco2.github.io/impact/`

3 Estimation of Carbon Emissions for Training Models

3.1 Models

After a review of all state-of-the-art neural audio synthesis models working directly on waveform, we selected those for which we had enough training details. These include the hardware used to train the model, such as the type of GPU and total training time in hours. Surprisingly, we found out that only five of the studies properly specified both criteria. Here, we present a short description of these models and the details of their training procedure according to the original papers.

SampleRNN introduced by Mehri et al. [2019] is an auto-regressive model producing one sample at a time, composed of auto-regressive multilayer perceptrons working at different temporalities. This model is trained for about one week on a GeForce TITAN X on three different datasets containing speech, vocal sounds and piano sonatas leading to a total of 168 hours of raw audio.

SING proposed by Défossez et al. [2018] is a convolutional neural audio synthesizer that generates waveform given desired categorical inputs. The training is composed of three parts on 4 NVIDIA P100 GPU on the NSynth dataset Engel et al. [2017] (333 hours): first, an auto-encoder is trained for 12 hours, then a sequence generator for 10 hours and finally an end-to-end fine-tuning for 30 hours.

WaveGAN (Donahue et al. [2019]) is a GAN that performs raw-waveform synthesis using transposed convolutions acting as upsampling modules. The network is trained on a single NVIDIA P100 GPU and converges within 4 days. Four different datasets are used: bird vocalization, speech, drum sound effects and piano (15.6 hours).

GANSynth Engel et al. [2019] uses GANs to generate log-magnitudes spectrograms and phases instead of modeling raw waveform directly. The training lasts 4.5 days on a NVIDIA V100 GPU on a subset of the NSynth dataset (78 hours).

FloWaveNet proposed by Kim et al. [2019] is a flow-based model for parallel waveform speech synthesis using the WaveNet architecture (Van Den Oord et al. [2018]) as an inverse transformation function. The training lasts 11.3 days on a NVIDIA Tesla V100 GPU and operates on the LJSpeech dataset (24 hours).

Note that although these five models were chosen for the availability of their training details and not for their specific architecture, we assume they form a representative set of generative models.

3.2 Experiments and results

Here, we want to estimate carbon emissions of each of these training procedures. Since we do not have all of the previously mentioned specific hardware, some hypotheses have to be taken into account. First of all, we make the assumption of the worst-case scenario, as does the Machine Learning Impact Calculator: we take the maximum power consumption p_{max} in Watts for each of the GPUs according to their technical specifications, and multiply it by n, the number of GPUs used for training and by t the training time in hours, to get the kilo-Watt hours consumption. We assume that the models are optimal and take most of the GPU resources. Note that the percentage of GPU utilization from `nvidia-smi` is not equivalent to the percentage of power consumption.

As carbon emissions are location-dependent, we took a carbon intensity factor of 0.437 kgCO_2eq/kWh as it is the global yearly average of 2018[3] to convert kilowatt-hours to carbon emissions. We ended up with the following formula to estimate the carbon emission (CO_2e) of a whole training as

$$\text{CO}_2\text{e} = 0,437 \times n \times p_{max} \times t \tag{6}$$

Results are shown in Table 1. We summarize the training details from column 1 to 3, and display the corresponding kilo-Watt hours and carbon footprint estimations for each of the five studied models. As we can see, the energy consumption ranges from 24 to 81.6kWh, and corresponding CO_2e estimation from 10,5 to 35,7 kgCO_2eq.

The estimations presented in Table 1 are linearly dependent on the training time, which is itself linearly dependent on the number of epochs before convergence and thus on the accuracy of the model. In other words, it is arguable that the more you train a model, the more energy it consumes but the more accurate it is. Therefore, we should consider the best "trade-off" between accuracy and carbon emission.

[3] https://www.carbonfootprint.com

Model	$n \times$ Hardware	p_{max}	t	kWh	CO_2e
SampleRNN	1 × GTX TITAN X	250	168	42.0	18.4
SING	4 × NVIDIA P100	1000	52	52.0	22.7
WaveGAN	1 × NVIDIA P100	250	96	24	10.5
GANSynth	1 × NVIDIA V100	300	108	32.4	15.45
FloWaveNet	1 × NVIDIA V100	300	272	81.6	35.7

Table 1: Approximated carbon emissions (named CO_2e) in kgCO_2eq of training several state-of-art neural audio synthesis models.

4 Multi-objective criteria

4.1 Our proposal

Increasing the size of a model and the number of training examples generally increases its accuracy, but also the energetic cost of its training. As these objectives are clearly conflicting, our idea is to rely on Pareto optimality, in order to evaluate a model according to both its accuracy and its environmental impact. Given two different models A and B with the same accuracy, but where A is more energy-efficient than B, A is said to *dominate* B (noted $A \succ B$). If there is no better solution than A, it is Pareto optimal. Hence, we aim to find the set of all Pareto optimal models to form a Pareto front and remove non-optimal models.

As discussed earlier, measuring the accuracy of generative models is a daunting task. The plurality of metrics used in the literature comes with the plurality of architectures. Indeed, no straightforward accuracy (or quality) objective score can be computed in creative tasks, conversely to classification or prediction tasks (apart from reconstruction rate in the case of VAE-based generative models). Hence, we took the most popular measure in audio synthesis evaluation (as seen in Figure 1), which best coincided with our 5 models. This may be a subjective evaluation, but it seems to be the most relevant across the audio generation literature. The MOS is a human-based measure of quality, ranging from 1 to 5, where participants are asked to rate as 1 the lowest perceived quality and 5 the highest when comparing a set of results. The final measure is computed as

$$\text{MOS} = \frac{1}{N}\sum_{n=1}^{N} R_n \tag{7}$$

where R_n is one rating and N the number of trials. As this score is highly dependent on each experimental setup, we compute

$$\%\text{MOS} = \frac{\text{MOS}_M}{\text{MOS}_{GT}} \tag{8}$$

to allow more accurate comparisons, where MOS_M and MOS_{GT} stands respectively for the MOS obtained by the model and the one obtained by the respective "ground truth" from each original paper. The higher the perceived quality of the sound produced by the model, the closer this ratio will be to 1, and conversely the lower the perceived quality, the closer it will be to 0. The goal is to maximize this ratio, and thus to minimize $1 - \%\text{MOS}$. We consider this last measure as our subjective accuracy score.

Regarding the energy-efficiency score, we separate training from inference. Regarding training, we take the previously introduced measure of carbon emission per training (see Table 1). Regarding inference, we rely on the number of parameters of the models. As discussed in Section 2, this count is highly correlated to the computational complexity and is independent of the device used to perform inference. We choose not to use the number of floating-point operations, as this computation is not straightforward : only manual counting (or coding an automatic implementation) can be done, which is rather constraining because it depends on each layer's characteristics (e.g., input size, kernel size, stride, padding, bias). A python package exists called "PyTorch-OpCounter"[4], but we found out that confusions were made between FPO and MACC (multiply-accumulate) operations. Moreover, researchers have to include their calculations when using different types of layers than the implemented ones (such as windowed convolution for SING).

[4] `https://github.com/Lyken17/pytorch-OpCounter`

4.2 Results

We summarize in Table 2 the MOS of the models and those of the ground truth reported in each original paper. We also compute our subjective score $1 - \%$MOS and count the number of parameters used by each model to infer new samples according to their original architectures. As SampleRNN and GANSynth use pairwise comparison instead of MOS, they are removed from this study.

Model	MOS_M	MOS_{GT}	1-%MOS	Param.
SampleRNN	-	-	-	52M
SING	$2,8 \pm 0,24$	$3,86 \pm 0,24$	0,26	64M
WaveGAN	$2,3 \pm 0,9$	$3,9 \pm 0,9$	0,41	89M
GANSynth	-	-	-	15M
FloWaveNet	$3,95 \pm 0,15$	$4,67 \pm 0,08$	0,15	186M

Table 2: Comparative Mean Opinion Scores ratios ($1 - \%$MOS) and number of parameters of several state-of-the-art neural audio synthesis models.

We display in Figure 3 the multi-objective space, where we plot the Pareto front for training (left) and for inference (right). The three models FloWaveNet, SING and WaveGAN are Pareto optimal in training, whereas WaveGAN is dominated by SING in inference and, therefore, is sub-optimal.

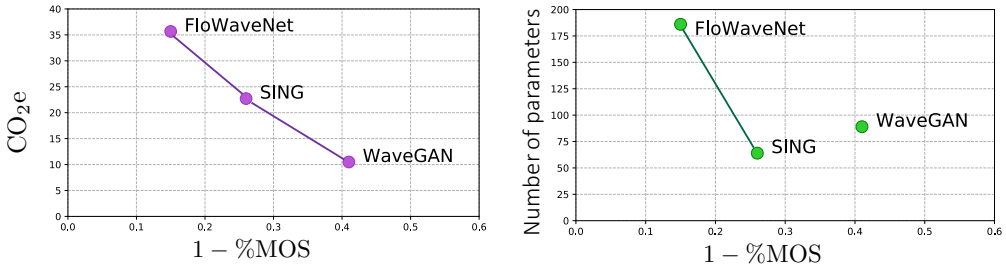

Figure 3: Representation of two Pareto Fronts. The objective is to minimize the subjective score ($1 - \%$MOS) along with the energy efficiency of either the training (left) with the measure of the carbon emission (CO_2e) per training, or the inference (right) with the number of parameters.

Since our goal is to propose a new tool for sustainable evaluation of models, we did not re-train the models to make our work more consistent and greener. Therefore, we would like to clarify to readers that we rely on approximations and hand-crafted measures; these figures support our overall approach, but it warrants more extensive and reliable analyses, with a larger array of models. However, it should be noted that our approach is generic, and could be applied to any type of model or input data.

5 CONCLUSIONS

In this paper, we first showed that the carbon footprint of a training procedure is far from marginal. We use indications (hardware and training time) from state-of-the-art neural audio synthesis models to approximate carbon emissions without having to re-train them. However, the lack of suitable training details affected our work, so we argue that authors must report the training time along with the device used for their training when publishing a new model. In general, a good habit would be to report actual carbon consumption of the training using tools such as the *Carbontracker* Python package or the online *ML CO2 impact* calculator.

While increasing awareness, we also showed that this calculation must be linked to the quality of the models. To that end, we proposed the use of a new metric based on Pareto optimality to give an equivalent importance to the model quality as their energy efficiency. Thus, this would place computational complexity at the heart of the research process. We rely on a subjective score for quality based on MOS and compute two Pareto fronts, one for training with our first estimation, and one for inference with the number of parameters of each models. In conclusion, we argue that our approach allows to find models that are non-optimal in both training and inference, facilitating the overall evaluation of research across all objectives simultaneously.

References

Amodei Dario and Hernandez Danny. Ai and compute, 2018. URL https://openai.com/blog/ai-and-compute/.

Danny Hernandez and Tom B. Brown. Measuring the Algorithmic Efficiency of Neural Networks. 2020. URL http://arxiv.org/abs/2005.04305.

Neil C. Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F. Manso. The Computational Limits of Deep Learning. 2020. URL http://arxiv.org/abs/2007.05558.

Emma Strubell, Ananya Ganesh, and Andrew Mccallum. Energy and Policy Considerations for Modern Deep Learning Research. *Aaai*, 2020a. URL www.aaai.org.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. Green AI. pages 1–12, 2019. URL http://arxiv.org/abs/1907.10597.

Lucas Theis, Aäron Van Den Oord, and Matthias Bethge. A note on the evaluation of generative models. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, pages 1–10, 2016.

Sercan Arik, Mike Chrzanowski, Adam Coates, Gregory Diamos, Andrew Gibiansky, Yongguo Kang, Xian Li, John Miller, Andrew Ng, Jonathan Raiman, Shubho Sengupta, and Mohammad Shoeybi. Deep voice: Real-time neural text-to-speech. *34th International Conference on Machine Learning, ICML 2017*, 1(Icml):264–273, 2017.

Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimber, Aäron Van Den Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis. *35th International Conference on Machine Learning, ICML 2018*, 6:3775–3784, 2018.

Eva García-Martín, Crefeda Faviola Rodrigues, Graham Riley, and Håkan Grahn. Estimation of energy consumption in machine learning. *Journal of Parallel and Distributed Computing*, 134:75–88, 2019. ISSN 07437315. doi:10.1016/j.jpdc.2019.07.007. URL https://doi.org/10.1016/j.jpdc.2019.07.007.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, (1):3645–3650, 2020b. doi:10.18653/v1/p19-1355.

Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. Samplernn: An unconditional end-to-end neural audio generation model. *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, pages 1–11, 2019.

Alexandre Défossez, Neil Zeghidour, Nicolas Usunier, Léon Bottou, and Francis Bach. Sing: Symbol-to-instrument neural generator. *Advances in Neural Information Processing Systems*, 2018-Decem(Nips):9041–9051, 2018. ISSN 10495258.

Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. *7th International Conference on Learning Representations, ICLR 2019*, pages 1–16, 2019.

Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. GANSynth: Adversarial neural audio synthesis. *7th International Conference on Learning Representations, ICLR 2019*, pages 1–17, 2019.

Taejun Kim, Jongpil Lee, and Juhan Nam. Comparison and Analysis of SampleCNN Architectures for Audio Classification. *IEEE Journal on Selected Topics in Signal Processing*, 13(2):285–297, 2019. ISSN 19410484. doi:10.1109/JSTSP.2019.2909479.

Jean Pierre Briot. From artificial neural networks to deep learning for music generation: history, concepts and trends. *Neural Computing and Applications*, 2020. ISSN 14333058. doi:10.1007/s00521-020-05399-0.

Philippe Esling and Ninon Devis. Creativity in the era of artificial intelligence. 2020. ISSN 23318422. URL http://arxiv.org/abs/2008.05959.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, (Ml):1–14, 2014.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 3, 2014. doi:10.3156/jsoft.29.5_177_2.

Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *32nd International Conference on Machine Learning, ICML 2015*, 2:1530–1538, 2015.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. pages 1–15, 2016. URL http://arxiv.org/abs/1609.03499.

Philippe Esling, Axel Chemla–Romeu-Santos, and Adrien Bitton. Generative timbre spaces: Regularizing variational auto-encoders with perceptual metrics. *DAFx 2018 - Proceedings: 21st International Conference on Digital Audio Effects*, pages 369–376, 2018.

Philippe Esling, Naotake Masuda, Adrien Bardet, Romeo Despres, and Axel Chemla-Romeu-Santos. Flow synthesizer: Universal audio synthesizer control with normalizing flows. *Applied Sciences (Switzerland)*, 10(1):1–11, 2020. ISSN 20763417. doi:10.3390/app10010302.

Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. Quantifying the Carbon Emissions of Machine Learning. 2019. URL http://arxiv.org/abs/1910.09700.

Lasse F. Wolff Anthony, Benjamin Kanding, and Raghavendra Selvan. Carbontracker: Tracking and Predicting the Carbon Footprint of Training Deep Learning Models. *arXiv*, 2020. ISSN 23318422.

Sean Vasquez and Mike Lewis. MelNet: A Generative Model for Audio in the Frequency Domain. (Figure 1), 2019. URL http://arxiv.org/abs/1906.01083.

Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan. Neural audio synthesis of musical notes with WaveNet autoencoders. *34th International Conference on Machine Learning, ICML 2017*, 3:1771–1780, 2017.

Aaron Van Den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George Van Den Driessche, Edward Lockhart, Luis C. Cobo, Florian Stimberg, Norman Casagrande, Dominik Grewe, Seb Noury, Sander Dieleman, Erich Elsen, Nal Kalchbrenner, Heiga Zen, Alex Graves, Helen King, Tom Walters, Dan Belov, and Demis Hassabis. Parallel WaveNet: Fast high-fidelity speech synthesis. *35th International Conference on Machine Learning, ICML 2018*, 9:6270–6278, 2018.