Proceedings of the Eighteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment
(AIIDE 2022)

Neural Synthesis of Sound Effects Using Flow-Based Deep Generative Models

Sergi Andreu,[1] **Monica Villanueva Aylagas**[2]
[1] KTH Royal Institute of Technology
[2] SEED - Electronic Arts (EA)
saandreu@kth.se, mvillanuevaaylagas@ea.com

Abstract

Creating variations of sound effects for video games is a time-consuming task that grows with the size and complexity of the games themselves. The process usually comprises recording source material and mixing different layers of sounds to create sound effects that are perceived as diverse during gameplay. In this work, we present a method to generate controllable variations of sound effects that can be used in the creative process of sound designers. We adopt WaveFlow, a generative flow model that works directly on raw audio and has proven to perform well for speech synthesis. Using a lower-dimensional mel spectrogram as the conditioner allows both user controllability and a way for the network to generate more diversity. Additionally, it gives the model style transfer capabilities. We evaluate several models in terms of the quality and variability of the generated sounds using both quantitative and subjective evaluations. The results suggest that there is a trade-off between quality and diversity. Nevertheless, our method achieves a quality level similar to that of the training set while generating perceivable variations according to a perceptual study that includes game audio experts.

1 Introduction

With the trend of video games becoming larger and more complex, maintaining a rich, high-quality sonic experience becomes a strain on content creators. Multiple studies have proven the relation between the game's sound and player immersion (Grimshaw 2007; Gormanley 2013). Creating multiple variations of each sound effect can foster immersion by decreasing listener fatigue, imitating reality's diversity (Barahona-Ríos and Collins 2021).

Traditionally, the creation workflow of sound designers includes recording sounds, or using libraries of pre-recorded sounds that are subsequently processed and layered to create a single, more complex sound effect (Collins 2008). Being able to produce variations of each layer in a faster, easier, or more cost-effective way than recording them can improve the development process but also enhance the creative experience of the sound designers.

A common way to create variations is to use procedural synthesizers, which generate sound effects in real-time based on input parameters that define them (Farnell 2007). Determining the proper input parameters is not straightforward for designers and the quality of the sounds is inferior to proposed data-driven approaches (Moffat and Reiss 2018; Barahona-Ríos and Collins 2021; Comunità, Phan, and Reiss 2021).

In recent years, the use of Deep Learning (DL) for audio generation has become more popular. While multiple methods focus on speech synthesis (Yamamoto, Song, and Kim 2020; Kong et al. 2020; Ping et al. 2020), fewer works are studying sound effects. In particular, generation of short, simple sound effects, such as knocking sounds (Barahona-Ríos and Pauletto 2020), footsteps (Comunità, Phan, and Reiss 2021) and gunshots or jumps (Barahona-Ríos and Collins 2021) using Generative Adversarial Networks (GANs) seem to be the primary target. To the best of our knowledge, longer, more complex sound effects have not been explored.

In this paper, we propose a method for user-guided generation of high-quality, complex sound effects variation, such as explosions, using WaveFlow (Ping et al. 2020), a generative flow model for raw audio. Our contributions include:

- High-quality generation in raw-form of sound effects using WaveFlow (Ping et al. 2020). Our conditional model allows users to generate variations similar to a given sound. To the best of our knowledge, this is the first time that a flow model has been used for neural synthesis of sound effects.
- A perceptual listening study that shows that our synthetic variations can be perceived as different, and achieve a quality similar to the training set, according to both game audio experts and non-experts working in the industry.
- Style-transfer capabilities when using example sounds out of the distribution of the training sound effect.

2 Related Work

Sound designers have traditionally used procedural audio as an alternative to pre-recorded sounds. Procedural audio is a term that can refer to a multiple areas of audio production including sound generation, remixing, music composition, etc. One of its benefits is real-time generation. In video games, parameters from the game engine, such as events or player actions, can be used to generate sounds or apply modifications to create variations (Farnell 2007).

There are multiple examples of procedural synthesizers in the literature (Bahadoran et al. 2018; Johansen, Pichlmair, and Risi 2020). While extremely useful for fully automatic generation of sound effects, they require ad-hoc mathematical models per effect. Moreover, previous works have shown that sounds synthesized with these methods have lower perceived quality compared to recordings and even the Deep Learning-based models presented in them (Moffat and Reiss 2018; Barahona-Ríos and Collins 2021; Comunità, Phan, and Reiss 2021). This makes procedural synthesizers poorly suited for our use case. We intend to enhance the creative experience of designers without sacrificing the final result excessively. For this reason, we will focus on data-driven approaches.

Zhao, Xia, and Togneri (2019) summarizes the early field of neural audio synthesis, commonly defined as generating audio with Deep Learning (Barahona-Ríos and Pauletto 2020). The survey presents an introduction to traditional methods for acoustic signal generation, focusing on speech and music, and how DL methods could be potentially replace them. A more recent survey (Tan et al. 2021) introduces a taxonomy and reviews state-of-the-art works in speech synthesis. Even though these methods are usually divided in three submodules, the last of them, known as vocoder, generates speech waveforms from acoustic features and can be trained on other type of sounds.

In the following sections, we review the main DL approaches that generate sound effects and present a brief overview of speech synthesis methods that could serve as inspiration to improve sound effects.

2.1 Sound Effect Synthesis

Barahona-Ríos and Pauletto (2020) use a conditional WaveGAN to generate knocking sound effects using as conditioner a label with emotional information. Similarly, Comunità, Phan, and Reiss (2021) train a conditional WaveGAN and a hybrid approach using Hifi-GAN (Kong, Kim, and Bae 2020) to synthesize footsteps on different surfaces.

Our approach differs from these works on several points. Conditional WaveGANs use label conditioning to generate different sounds but lack finer control. WaveGAN, even larger versions like the one described by Barahona-Ríos and Pauletto, can only synthesize a limited number of samples.

SpecSinGAN (Barahona-Ríos and Collins 2021) takes a different approach. A single-image GAN based on ConSinGAN (Hinz et al. 2021) learns the internal distribution of one training example, exploiting overlapping patches, to produce variations from it. The model can synthesize single or multi-layered sounds depending on the training spectrograms. The method is tested on four sound effects (footsteps on concrete and metal, gunshots, and jumps) but on short examples ($\approx 200 - 750$ ms).

This approach is particularly suitable for cases where a single example is available. When more data is accessible, choosing a single-image algorithm would compromise potential variability. Although the results reported achieve a high plausibility score, image-based algorithms for sound are problematic due to phase reconstruction being non-invertible (Donahue, McAuley, and Puckette 2018).

Even though GANs are widely used for sound effect synthesis, they are not the state of the art for other types of sound generation and are known to be unstable to train. In contrast, recent developments in speech synthesis are more promising.

2.2 Speech Synthesis

WaveGAN has been employed as the base for Barahona-Ríos and Pauletto (2020), Comunità, Phan, and Reiss (2021). However, it was originally tested on a wider and more complex set of domains, including speech and drums.

Parallel WaveGAN (Yamamoto, Song, and Kim 2020) differs from the original WaveGAN, among other things, in that it synthesizes using 80-band log-mel spectrograms as auxiliary input in a text-to-speech framework. Using spectrograms as conditioners is a feature that most speech synthesis generators have in common.

However, GANs are not the only models capable of generating high-quality waveforms. Other generative models have been successfully used for this task in recent years.

DiffWave (Kong et al. 2020) is a Denoising Diffusion Probabilistic Models (DDPM) model that converts noise into a waveform by optimizing a modified version of the variational lower bound (ELBO). It shows high-quality generation in different tasks, including speech synthesis conditioned on mel spectrograms. However, it is slower than WaveFlow (Ping et al. 2020), the state-of-the-art flow-based model, with similar performance in terms of quality as measured by Mean Opinion Score (MOS).

WaveFlow is a Normalizing Flow (NF) model that, as opposed to DDPMs, trains directly on maximum likelihood. It presents a unified framework for likelihood-based models that allows a trade-off between capacity and inference speed. Its small size makes it attractive for production settings.

We decided to use WaveFlow for its generation quality, the mathematical properties of Normalizing Flows, and the flexibility of the unified view that this architecture exposes, which could be helpful in particularly demanding production scenarios.

3 Method

The objective of this work is to explore a family of models that are able to generate high-quality variations in the style of a specific sound effect from an example sound given as condition. To do so, we use WaveFlow to learn the distribution of plausible sound effects given a lower-dimensional representation of the initial sound the user would like to diversify.

Our pipeline, described in Fig. 1 works as follows: During training, 1D waveforms $\boldsymbol{x} = x_{1:T}$ are taken from the dataset and reshaped into an h-row 2D matrix $X \in \mathbb{R}^{h \times w}$. Our conditioner is the ground-truth mel spectrogram aligned and reshaped $C \in \mathbb{R}^{c \times h \times w}$ to match X, with c being the number of mel bands. WaveFlow learns the invertible function f^{-1} that approximates the conditional distribution of the sound effect given the spectrogram and an isotropic Gaussian distribution (IGD). During inference, the user selects an example sound from which the conditioner C is computed. We

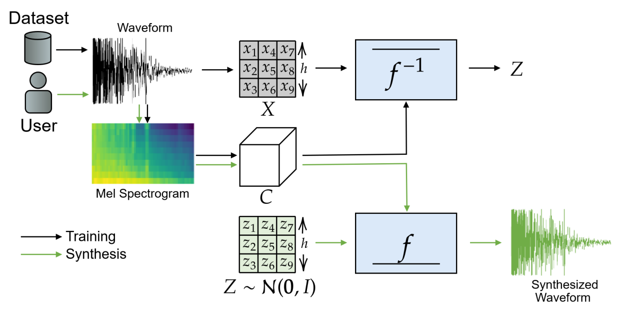


Figure 1: System overview. We learn an invertible function f implemented as a Normalizing Flow network (WaveFlow) that approximates the distribution of a type of sound effect by transforming an isotropic Gaussian distribution. The network is conditioned on a ground truth low resolution mel spectrogram during training that can be use in inference to guide the generation process. Z is the source of randomness that will introduce variations.

can use f together with the provided C to synthesize sounds. Z, sampled from the IGD, is the source of randomness that creates variability in the generation.

3.1 Normalizing Flow Framework and WaveFlow

As first presented by Dinh, Krueger, and Bengio (2014), flow models assume that the distribution of the data $p(\boldsymbol{x})$ can be expressed as a transformation of a simpler distribution $p(\boldsymbol{z})$, in our case an IGD, by a non-linear, invertible and differentiable function $\boldsymbol{x} = f(\boldsymbol{z})$.

The change of variable formula allows the exact optimization of the maximum likelihood of the data, following:

$$p(\boldsymbol{x}) = p(f^{-1}(\boldsymbol{x})) \left| \det \left(\frac{\partial f^{-1}(\boldsymbol{x})}{\partial \boldsymbol{x}} \right) \right| . \quad (1)$$

In practice, $f = f_1, ..., f_N$ is a chain of multiple simpler transformations implemented as a neural network (NN). Fundamental research in NFs centers in developing transformations that permit the efficient computation of the Jacobian determinant (Papamakarios et al. 2021). WaveFlow presents a unified view on autoregressive and bipartite transformations, being able to implement Wavenet and WaveGlow as special cases (Ping et al. 2020).

Autoregressive transformations can be trained in parallel, but inference is slow due to its dependency on all previous samples. On the other hand, bipartite transformations split the input. This allows both training and inference to be parallelizable but also reduces the expressivity of the flow since a part of the input is not transformed.

WaveFlow reshapes the waveform into a 2D matrix where contiguous samples fall under the same column. This reshaping allows trading inference time and model capacity. Long-range dependencies are processed with convolutions and short-range information with autoregressive functions over the rows h, being able to compute the columns w in parallel.

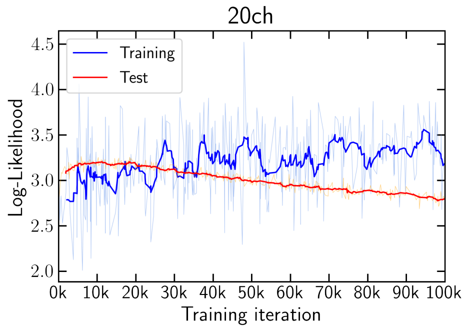


Figure 2: Log-likelihood over time of the training and test datasets for the model conditioned on 20 mel bands.

For speech synthesis, WaveFlow uses mel spectrograms computed from ground truth audio following the lead of WaveGlow (Prenger, Valle, and Catanzaro 2019), and FloWaveNet (Kim et al. 2019). The mel spectrograms are upsampled to the waveform length and reshaped to match the also reshaped 2D sound matrix, using the number of bands of the spectrogram as channels. The residual channels obtained from processing the aligned spectrogram via 1×1 convolutions are added at each layer.

3.2 Family of Models

We start from this setting in our experiments. However, the use of high-dimensional mel spectrograms (80 bands) for speech synthesis is designed to achieve the best generation quality, not to encourage variability in the generation. For that reason, one of the parameters we aim to study is how the dimensionality of the conditioner affects the quality and the diversity of the generated samples.

Previous works report a correlation between quality and the log-likelihood (LL) (Ping et al. 2020). We want to study if this is the case for our setting since we work with a smaller dataset. We also believe that sound effects are more difficult to evaluate qualitatively than human voices. To that end, we avoid early stopping to study how prolonged training affects both the quality and the diversity of the synthesized sounds. We report training and testing LL in Fig. 2

The base model, for which these parameters will be studied, complies with the following description: $h = 16$, 64 residual channels, and 8 flows of 8 layers. We set the dilation cycle of WaveNet to $d = [1, 2, 4, 8, 16, 32, 64, 128]$ following van den Oord et al., and with $k = 3$ we get a large receptive field that helps with long-range dependencies.

According to Ping et al., the best LL is achieved with $h = 64$ for these parameters. We decided to reduce it to $h = 16$ to speed up performance since we hypothesize that sound effects do not need as complex transformations as speech. Good results were originally achieved with these parameters, but it might be possible that a better architecture can be attained by systematically fine-tuning hyperparameters. However, the lack of good quantitative metrics makes this task challenging.

The perceptual study covers models trained with conditioners that use $c = [10, 15, 20, 30]$ mel bands at iteration

50K. We also study the training iteration for the model using $c = 20$, with checkpoints taken at $it = [10, 20, 50, 100, 200]$K, which includes decreasing LLs (Fig. 2).

3.3 Style Transfer

The only input required for generation is a vector sampled from an IGD and a low-dimensional mel spectrogram from the sound to emulate. The network has learned to map the conditioner to the particular type of sounds exposed to during training, hallucinating the details missing in the spectrogram. Thus, it is possible to generate the desired type of sound effect by conditioning on a different one, as long as the spectrogram distributions are not too dissimilar.

In our experiments, we explore the generation of explosion sound effects by conditioning on percussive sound spectrograms. Due to unreliable evaluation metrics and the secondary relevance of the application, we decided not to dilute the central results with more experiments. However, this byproduct opens an new and interesting use case worth reporting. Qualitative results can be found on our demo webpage [1].

3.4 Dataset and Data Processing

Our dataset consists of explosion sound effects. We have a small amount of in-house data that sums up to 177 files of stereo sound with a sample rate of 48KHz. Sounds are recorded under different conditions and have a variable number of variations, ranging from 3-16. The length of the sounds is also variable, fluctuating between 1.5-10s.

The dataset is split randomly into 90% training and 10% test subsets. In order to augment the dataset, we convert the effects to mono and use both channels as different samples. The audio is downsampled to 16KHz to allow fast iteration.

The waveform is normalized to $(-1, 1)$. The mel spectrogram is computed from the normalized waveform using the Short-time Fourier transform (STFT) and converted to a decibel scale. Then, it is normalized to $(0, 1)$.

Even though WaveFlow claims to avoid constant frequency noise, we detect cases in our synthesis. We designed a naive function that removes spiked and neighboring frequencies. This post-processing is performed on a complex STFT that allows a lossless reconstruction.

3.5 Performance Details

Training takes between 1.14-1.23 seconds/iteration using two NVIDIA GeForce GTX 1080, depending on the dimensionality of the conditioner. With just one GPU, inference can be performed $\approx 10.4\times$ faster than real-time at 16KHz (≈ 166667 samples/s). This means that our longest sound, 10s long, can be synthesized in 0.96s. In CPU, performance is $\approx 1.1\times$ faster than real-time, or ≈ 17544 samples/s.

Note that Ping et al. uses a V100 GPU while we report performance on hardware that can be reasonably found on a gaming computer.

[1]https://go.ea.com/exflowsions

4 Evaluation

Quantitative sound generation evaluation is still an unsolved research area. In this section, we review how similar works address it and motivate our choices for the listening study framework and quantitative metrics.

4.1 Qualitative Evaluation

We ran an internal user study with 15 participants (7 game audio experts and 8 non-experts working in the industry). Since our use case is more similar to SpecSinGAN (Barahona-Ríos and Collins 2021), we follow their listening study framework modifying the answers to use a four-point grading scale akin to Comunità, Phan, and Reiss due to the small range of variation and quality present in the samples.

The following are sources evaluated in the study. The alias of the models follows the number of mel bands and the number of iterations (e.g., *30ch_50k* is a model conditioned on a 30-band mel spectrogram, trained for 50K iterations).

- *Training*: Samples from the training set. They represent the upper limit of quality and diversity.
- Dimensionality of the mel spectrogram conditioner: We train models with $[10, 15, 20, 30]$ mel bands and compare them under the same number of iterations (50K).
- Training iterations: We compare models at $[10, 20, 50, 100, 200]$K iterations trained on the same conditioner (20 mel bands).
- Post-processing: Calibration sets with sounds coming from the model *20ch_50k*. *Unprocessed* samples come directly from the model and *Ultraprocessed* samples are more aggressively post-processed with our naive algorithm (it removes more neighboring frequencies).

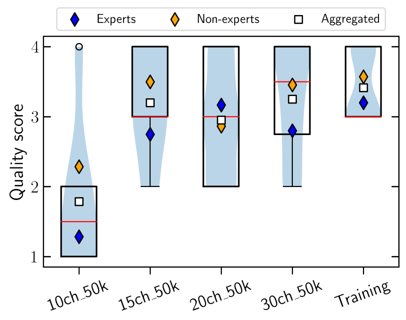
All sounds are mono tracks at 16KHz. Three sounds generated from the same conditioner (or recording condition) were concatenated to be able to assess diversity. The participants were presented 1 example from training, 1 from post-processing, and 10 from different models sampled randomly out of a pool of 55 possibilities. For each question, we asked: *How realistic/similar are the presented explosion sounds?*

We included a tutorial to create a reference. Firstly, we played *Training* samples, demonstrating the upper limit quality of our synthesis. Then, we evidence an extreme case of variability, where the sounds came from different recording conditions. Since our generated sounds need to sound close to the conditioned example, they would never achieve this level of diversity. Finally, we presented an example of low-quality sounds, which included artifacts.

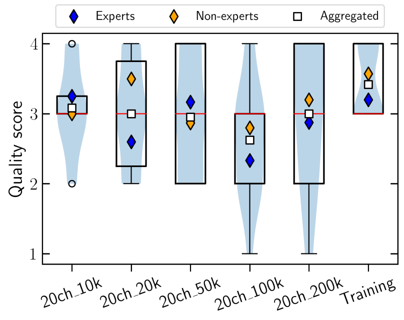
We designed an attention check to filter unreliable answers. Participants that graded *Training* samples as unrealistic [1-2] were removed from the analysis. We used the Mann–Whitney U tests to validate our hypotheses using the ratings of the study.

4.2 Quantitative Evaluation

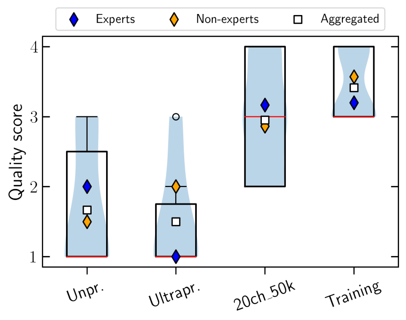
We use quality metrics previously presented in other works. Several metrics measure the distance between the real and



(a) Quality comparison of models with different dimensionality of the mel spectrogram conditioner.

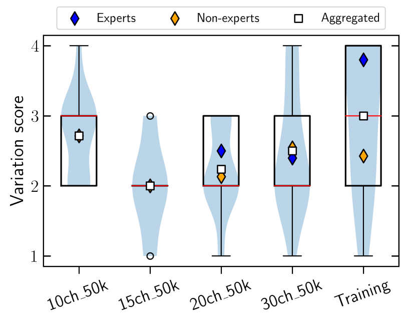


(b) Quality comparison of the model *20ch_50k* at different training iterations.

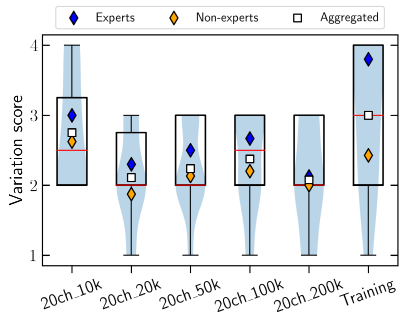


(c) Quality comparison of different post-processing for the model *20ch_50k*.

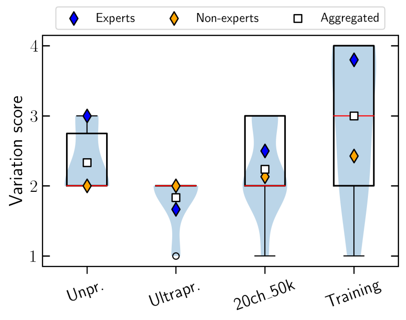
Figure 3: Quality ratings of the listening study.



(a) Variation comparison of models with different dimensionality of the mel spectrogram conditioner.



(b) Variation comparison of the model *20ch_50k* at different training iterations.



(c) Variation comparison of different post-processing for the model *20ch_50k*.

Figure 4: Variation ratings of the listening study.

synthetic distributions. We decided to explore Fréchet Audio Distance (FAD) (Kilgour et al. 2018) which correlates well with user preference according to Comunità, Phan, and Reiss. The LL of the model has also been reported to correlate with MOS (Ping et al. 2020). Additionally, we introduce a new metric, YAMNet Probability Score (YPS).

YAMNet (Plakal and Ellis 2020) is a NN pretrained on human-labeled sound from YouTube. The network predicts 521 classes that cover natural and human-made sounds.

Formally, we define the YAMNet Probability Score as:

$$YPS(x) = \frac{\sum_{i \in \mathcal{I}_p} c(y_i|x)}{1 - \sum_{i \in \mathcal{I}_n} c(y_i|x)}, \tag{2}$$

where $c(y_i|x)$ is the normalized YAMNet score for the class y_i and waveform input x. $\mathcal{I}_p$ and $\mathcal{I}_n$ represent sets of indices selected as positive and neutral classes, respectively. YAMNet performs classification over time windows, so we average the normalized probabilities over all the input.

Adopting YPS is particularly useful when the amount of data prevents training a good classifier, as in our case. It is also a flexible metric that can be designed for multiple types of sound effects by modifying the positive and neutral indices. This can be done by sound designers or semi-automatically by analyzing the classification of the dataset.

Variability is only measured in the literature with Inception Score (Salimans et al. 2016). However, it measures inter-class diversity. Instead, we would like to quantify the diversity of sounds generated from the same conditioner.

5 Results

In this section, we analyze the user study results and compare them to the quantitative metrics for the two aspects under examination: quality and diversity.

The results of the listening study were analyzed after filtering out the 2 participants (1 expert and 1 non-expert) that failed the attention check.

5.1 Quality

Attending to the number of mel bands used for conditioning, it is evident that $c = 10$ does not produce desirable results (Fig. 3a). The rest of the experiments, however, rank closer to *Training* (note that the distribution is artificially truncated due to the attention test). From those, the one with the best score is the model that uses $c = 30$. Interestingly, experts' opinions on quality are lower than non-experts except for *20ch_50k*, which suggests a measurement error. On

Model	FAD ↓	YPS% ↑
10ch_50k	14.74 ± 0.59	15.11 ± 0.72
15ch_50k	8.09 ± 0.38	57.07 ± 0.53
20ch_50k	8.01 ± 0.79	**61.16 ± 1.42**
30ch_50k	**7.50 ± 0.69**	54.17 ± 2.21
20ch_10k	10.69 ± 0.50	59.98 ± 3.58
20ch_20k	8.74 ± 0.85	48.70 ± 2.91
20ch_50k	**8.01 ± 0.79**	**61.16 ± 1.42**
20ch_100k	8.85 ± 0.85	47.81 ± 4.20
20ch_200k	9.84 ± 0.47	42.42 ± 3.04
Train	3.73	65.10
Unprocessed	5.70 ± 0.09	56.67 ± 1.80
Ultraprocessed	15.30 ± 0.25	4.33 ± 0.55

Table 1: Quantitative quality metrics. The models are separated in three groups according to the dimensionality of the mel spectrogram, the training iteration and the references.

the other hand, quality does not seem to be extremely influenced by training time. The mean and median scores fluctuate very little over the iterations under study. In fact, Fig. 3b suggests that early iterations (10-50K) produce consistently better results which, indeed, seems to correlate with the LL of the model in Fig. 2. However, the LL at iteration 0 is better than at the end, which would not be perceptually appropriate. The results of Fig. 3c show that our naive post-processing improves quality over *Unprocessed* sounds, and does not harm the results like *Ultraprocessed*.

None of the quantitative metrics reported in Table 1 seem to match human preference completely. FAD values correspond to user perception for *Training* and the best/worst mel band models. But it considers *Unprocessed* better than *20ch_50k*, unlike YPS and the listening study. That YPS matches the participants in this case suggest that the metric correlates better with sound quality than plausibility. A larger listening study might strengthen these relationships.

5.2 Diversity

The variation results in Fig. 4a clearly support our hypothesis that variation is better achieved with low dimensionality conditioners since the best performing model is *10ch_50k*. However, this is also the model that achieves worse quality. Among the best quality models, the one that attains better diversity is *30ch_50k* with a higher average and more consensus between experts and non-experts. Variability also seems to be better in early iterations. At $it = 20$K, the diversity decreases sharply and improves with training, but never to the same level as $it = 10$K. It is interesting to see that experts' variability ratings are consistently equal to or higher than non-experts (except for *30ch_50k*), which suggests that non-experts struggle to differentiate similar sounds, even in the training set. Our post-processing seems to slightly reduce the perceived variability compared with *Unprocessed* sounds but less than the *Ultraprocessed* version, according to Fig. 4a.

6 Discussion and Future Work

A Mann-Whitney test partially supports our hypothesis that using more mel bands in the conditioner produces better quality results (30ch vs 20ch, $\rho = 0.60$; 20ch vs 15ch, $\rho = 0.41$; 15ch vs 10ch, $\rho = 0.85$) but less diverse (30ch vs 20ch, $\rho = 0.44$; 20ch vs 15ch, $\rho = 0.60$; 15ch vs 10ch, $\rho = 0.76$). This suggests that there is a trade-off between quality and variability. From the iterations under study, $it = 10$K achieves the best quality (10k vs 20k, $\rho = 0.53$; 10k vs 50k, $\rho = 0.55$; 10k vs 100k, $\rho = 0.67$; 10k vs 200k, $\rho = 0.50$) and variability (10k vs 20k, $\rho = 0.69$; 10k vs 50k, $\rho = 0.65$; 10k vs 100k, $\rho = 0.60$; 10k vs 200k, $\rho = 0.69$). This indicates a detriment in long trainings and the necessity to finding an early stopping strategy. Finally, the Mann-Whitney test strongly suggests that our naive strategy is better than not post-processing the results ($\rho = 0.83$) and preferred over a more extreme approach ($\rho = 0.89$).

Overall, the expert participants liked the sounds generated by our best models in terms of realism and variability:

> - Participant 2 (expert): Noticeable transient differences. Sounds develop differently within first 0.5s.

However, there is a consistent dissatisfaction with the quality level from the initial downsampling, criticized both on the training examples and the synthetic sounds. Despite this, the overall sentiment towards the results is positive:

> - Participant 14 (expert): In general I still can't believe these sounds are "generated". It feels surreal to me. Really seems like this is the first steps into something that will blow everyone away (no pun intended).

Unfortunately, we did not find correlations between quantitative metrics and perceptual scores. More research is needed to find useful metrics to guide fine-tuning, but also a more extensive listening study to draw stronger statistical results. Metric correlation on human perception could also be dependent on the type of sound effect, thus worth studying.

The next steps to validate our approach would be to train the model on the original sounds at 48KHz to appease the need for increased quality from audio experts. Experiments are in progress, but we believe in their success due to our large receptive field and previous evidence that WaveFlow works at higher sampling rates. Testing on a dataset with different sound effects would also be a natural step forward. A recently published paper, SaShiMi (Goel et al. 2022), shows that replacing WaveNet with their method in DiffWave improves performance in multiple areas. Since WaveFlow also makes use of WaveNet, it would be interesting to investigate the same replacement in our approach.

Further research may include generating variation on the mel spectrogram. More exploration of the latent space could unlock better editability, similar to how image generation networks are able to change expressions (Kingma and Dhariwal 2018). Finally, training a single network conditioned on the type of sound effect is a compelling undertaking: the model would be able to draw information from all types of sounds while maintaining the style transfer capabilities through the label conditioner.

7 Conclusions

We have adopted WaveFlow, a speech synthesis model for raw waveforms, for the task of generating sound effect variations from a small dataset. We propose to apply our approach to enhance the creative process of sound designers by focusing on creating high-quality variations of a single sound effect layer to increase the number of available prerecorded sounds. The users can request as many variants as needed to process and remix with other layers until reaching the desired final sound.

Our evaluation shows that our models can achieve a quality near to the training set, both according to game audio experts and the average population while maintaining perceivable variations. Moreover, due to the nature of our approach, the model attains certain style transfer capabilities, augmenting the possibilities of creating the desired sound effect from other example sounds. We demonstrate this effect on our demo page by converting percussive sounds into explosions.

Acknowledgments

We would like to express our gratitude to Albin Jansson and the audio department at DICE for the helpful discussions and for supplying the training data, as well as to everyone who participated in the user study.

A Appendices

We include in this section some information that might be useful for reproducibility.

A.1 Model

We base our implementation on a publicly available PyTorch code (L0SG 2019). All of our models use the Adam optimizer (Kingma and Ba 2015) with the batch size set to 8 and a constant learning rate of 2e−4. The network is trained using the Apex mixed precision library (NVIDIA 2018), which increases training speed by making use of 16-bit floating-point operations.

A.2 Dataset

The sounds in our dataset are explosions recorded under different conditions and labeled using three high-level features: size, location and distance. These features can take three different values according to Table 2.

Initially, we trained our model with zero-padded sounds since we have samples of different lengths, but the training became unstable. To solve this, we decided to use only the first second of the samples, which always contain the transient, the body, and at least part of the tail of the explosion. During training, the dataloader randomly selects a segment of 14K samples per audio sample, per iteration.

A.3 Metrics

In our implementation of YAMNet Probability Score for explosions, we consider *explosion*, *fireworks*, *gunshots*, *firecracker*, *bursts*, *artillery fire*, *boom* and *eruption* as positive classes ($\mathcal{I}_p$) and *silence* and *percussion* as neutral classes ($\mathcal{I}_n$) in equation 2.

High-level features	**Values**
Size	Small Medium Large
Location	Indoor Urban Field
Distance	Close Distant Far

Table 2: Conditions under which the explosions in our dataset are recorded.

References

Bahadoran, P.; Benito, A.; Vassallo, T.; and Reiss, J. D. 2018. Fxive: A web platform for procedural sound synthesis. In *Audio Engineering Society Convention 144*. Audio Engineering Society.

Barahona-Ríos, A.; and Pauletto, S. 2020. Synthesising Knocking Sound Effects Using Conditional WaveGAN. In *SMC Sound and Music Computing Conference 2020, Torino, 24-26 June 2020.*

Barahona-Ríos, A.; and Collins, T. 2021. SpecSinGAN: Sound Effect Variation Synthesis Using Single-Image GANs. arXiv:2110.07311.

Collins, K. 2008. *Game sound: an introduction to the history, theory, and practice of video game music and sound design*. Mit Press.

Comunità, M.; Phan, H.; and Reiss, J. D. 2021. Neural Synthesis of Footsteps Sound Effects with Generative Adversarial Networks. arXiv:2110.09605.

Dinh, L.; Krueger, D.; and Bengio, Y. 2014. NICE: Non-linear Independent Components Estimation. arXiv:1410.8516.

Donahue, C.; McAuley, J.; and Puckette, M. 2018. Adversarial Audio Synthesis. In *International Conference on Learning Representations*.

Farnell, A. 2007. An introduction to procedural audio and its application in computer games.

Goel, K.; Gu, A.; Donahue, C.; and Re, C. 2022. It's Raw! Audio Generation with State-Space Models. In Chaudhuri, K.; Jegelka, S.; Song, L.; Szepesvari, C.; Niu, G.; and Sabato, S., eds., *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, 7616–7633. PMLR.

Gormanley, S. 2013. Audio immersion in games—a case study using an online game with background music and sound effects. *The Computer Games Journal*, 2(2): 103–124.

Grimshaw, M. N. 2007. Sound and immersion in the first-person shooter. In *The 11th International Computer Games Conference*. University of Wolverhampton, School of Computing and Information Technology.

Hinz, T.; Fisher, M.; Wang, O.; and Wermter, S. 2021. Improved techniques for training single-image gans. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 1300–1309.

Johansen, M.; Pichlmair, M.; and Risi, S. 2020. Squeezer-A Tool for Designing Juicy Effects. In *Extended Abstracts of the 2020 Annual Symposium on Computer-Human Interaction in Play*, 282–286.

Kilgour, K.; Zuluaga, M.; Roblek, D.; and Sharifi, M. 2018. Fréchet Audio Distance: A Metric for Evaluating Music Enhancement Algorithms. arXiv:1812.08466.

Kim, S.; Lee, S.-G.; Song, J.; Kim, J.; and Yoon, S. 2019. FloWaveNet: A Generative Flow for Raw Audio. In *International Conference on Machine Learning*, 3370–3378. PMLR.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In Bengio, Y.; and LeCun, Y., eds., *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Kingma, D. P.; and Dhariwal, P. 2018. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31.

Kong, J.; Kim, J.; and Bae, J. 2020. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis. *Advances in Neural Information Processing Systems*, 33: 17022–17033.

Kong, Z.; Ping, W.; Huang, J.; Zhao, K.; and Catanzaro, B. 2020. DiffWave: A Versatile Diffusion Model for Audio Synthesis. In *International Conference on Learning Representations*.

L0SG. 2019. A PyTorch implementation of "WaveFlow: A Compact Flow-based Model for Raw Audio" (ICML 2020).

Moffat, D.; and Reiss, J. D. 2018. Perceptual evaluation of synthesized sound effects. *ACM Transactions on Applied Perception (TAP)*, 15(2): 1–19.

NVIDIA. 2018. A PyTorch Extension: Tools for easy mixed precision and distributed training in Pytorch.

Papamakarios, G.; Nalisnick, E.; Rezende, D. J.; Mohamed, S.; and Lakshminarayanan, B. 2021. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57): 1–64.

Ping, W.; Peng, K.; Zhao, K.; and Song, Z. 2020. WaveFlow: A compact flow-based model for raw audio. In *International Conference on Machine Learning*, 7706–7716. PMLR.

Plakal, M.; and Ellis, D. 2020. YAMNet.

Prenger, R.; Valle, R.; and Catanzaro, B. 2019. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3617–3621. IEEE.

Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; and Chen, X. 2016. Improved techniques for training gans. *Advances in neural information processing systems*, 29.

Tan, X.; Qin, T.; Soong, F.; and Liu, T.-Y. 2021. A Survey on Neural Speech Synthesis. arXiv:2106.15561.

van den Oord, A.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A. W.; and Kavukcuoglu, K. 2016. WaveNet: A Generative Model for Raw Audio. In *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13-15 September 2016*, 125. ISCA.

Yamamoto, R.; Song, E.; and Kim, J.-M. 2020. Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6199–6203. IEEE.

Zhao, Y.; Xia, X.; and Togneri, R. 2019. Applications of deep learning to audio generation. *IEEE Circuits and Systems Magazine*, 19(4): 19–38.