ROOM IMPULSE RESPONSE ESTIMATIOM FROM REVERBERANT SPEECH USING GAN

Tom-Avi Shapira and Israel Cohen

Array Signal Processing and Analysis - 048828, Winter 2022
Technion - Israel Institute of Technology
{tom-avi@campus,icohen@ee}.technion.ac.il

ABSTRACT

Room impulse response (RIR) estimation is a fascinating problem that has a lot of applications. Augmented and virtual reality, automatic speech recognition (ASR) and de-reverberation audio processing are some of them. In this paper we present a method for RIR estimation from a reverberant speech audio file, using a neural network. The network architecture is based on FASR-RIR [1] architecture, which generates RIR from an embedded vector that represents room parameters as an input. To estimate RIR of a real room, in this paper we suggest a modification to the FAST-RIR architecture which allows to use audio file as an input. Namely the network was designed to generate new RIR that estimates the ground-truth RIR of the room where the input audio file was recorded in. We have shown that our model succeeds to extract a target room's features from a reverberant speech and then estimate RIRs well enough for a proof of concept.

1. INTRODUCTION

Room impulse response (RIR) represents the acoustic parameters of a room, giving a source and a sensor location, i.e., speaker and microphone locations. We can think about it as the impulse response of a system when the room is the system. So, if we record a sound of pulse in a room, the result will be the RIR of that room. One important parameter that can be calculated from the RIR is the time that takes to the amplitude of a sound to decay in the room by 60 decibels, this parameter is called the reverberation time and marked as T_{60}.

Today RIRs are used in a lot of applications. Augmented and virtual reality, automatic speech recognition (ASR) and de-reverberation audio processing are some of them. ASR for example requires a huge dataset of RIRs to train on. In fact, it is not scalable to record such dataset of real RIRs, so the solution is to generate them given rooms' parameters like dimensions and T_{60}.

RIRs generators are already exist, and as mentioned, are used to simulate a large dataset of RIRs successfully. We can divide them into three categories. The first is the wave-based generators. These generators give the most accurate results because they are based on solving wave equations [2] [3]. However due to the calculations complexity they can generate only RIRs of simple rooms. The second is the ray-based generators which relies on the wave reflections from a given surface. These generators are less accurate than the formers but more feasible to generate RIRs of complex rooms. One known ray-based method is the image method [4]. The last category I will mention is the neural network-based generators, like FAST-RIR [1]. The main improvement these generators archive comparing to the ray-based ones, is the calculation time of RIRs which improved significantly.

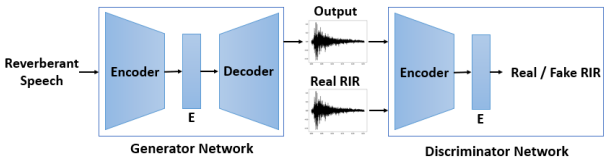

Figure 1: The architecture of our model. Our generator network takes a reverberant speech as input and generates corresponding RIR as output. Our discriminator network discriminates between the generated fake RIR and the ground truth RIR for the given reverberant speech during training.

While RIR generators are designed to generate new RIRs given specific parameters, RIR estimation is another problem which aims to estimate the real RIR of existing room. Estimated RIRs can be used for example in augmented and virtual reality to simulate a sound as if it was recorded in a specific room. There are some methods to calculate a RIR of a room as described in [5] [6] [7]. Going forward, we will refer to FiNS [8] architecture which estimates the time domain RIR from reverberant speech using a neural network.

In this paper we propose a method to estimate RIRs of real rooms from a reverberant speech audio file, using a modification of FAST-RIR architecture. In fact, we suggest estimating a RIR by generating a new RIR that will be close enough to the ground-truth RIR of a specific room, while the run-time complexity remains efficient (as in the original FAST-RIR). To allow an audio file as an input for our generator, the network architecture is a

combination of FAST-RIR architecture and the encoder from FiNS architecture.

2. APPROACH

2.1. The Model

2.1.1. Conditional GAN

Generally, our model is based on conditional GAN [9] (CGAN) architecture. The generator learns to generate new data with the same statistics as the training set and generates RIRs based on reverberant speech. In parallel, the discriminator classifies whether these RIRs are real or fake. The generator and the discriminator are trained to optimize a zero-sum game. Meaning that the loss of one decreases while the other's increases. Simplicity, it can be said that the generator tries to fool the discriminator with fake RIRs that it generates, and the discriminator tries to classify correctly between real and fake RIRs.

2.1.2. Network Architecture

Our network architecture is represented in Figure 1. Our generator is based on a combination between the encoder of FiNS [8] and stage-I netG of FAST-RIR [1]. Generally, we modified the architecture of FASR-RIR so it can accept as input a 128-dimensional embedded vector (E). This vector represents a target room features which extracts from a reverberant speech audio file instead of the originally embedded input vector at FAST-RIR which directly represents room dimensions and reverberation time. To allow this modification, we used the encoder from FiNS architecture which aim to learn a target room features from a reverberant speech audio file. FiNS works at 48 kHz sample rate, and it takes inputs of approximately 2.73 seconds (131072 samples). We chose to stick to FAST-RIR sample rate which is 16 kHz, and our inputs are 3 seconds length (48000 samples). Same as FiNS, every encoder block contains a 1-D convolution layer with a kernel size of 15 and stride 2. Due to the stride value and because of our inputs' samples length is about half of FiNS's inputs, our encoder consists of 12 blocks instead of 13 blocks as at FiNS. 12 blocks with kernel size of 15 and stride 2 (as mentioned before) achieves a receptive field of over 57,000 samples, which equates to 3.58 seconds at 16 kHz. Finally, our generator uses a decoder to decode the target room features and generates RIRs with a length of 4096 samples, as the originally FAST-RIR generated RIRs length. Our discriminator is based on stage-I netD of FAST-RIR. Generally, the discriminator gets as inputs real RIRs and fake RIRs which generated by the generator. Then, using an encoder, extracts from each of them the corresponding target room features vector (E) and classifies whether it is real or fake RIR. Notice the difference between the discriminator's encoder and the generator's encoder. The first one extracts target room features from RIR recording and the second one extracts target room features from reverberant speech recording.

2.1.3. Loss Functions

Generally, we use the loss functions as they were represented in FAST-RIR [1]. The generator network (G_N) is trained with a loss function that consists of three parts. The first one is a modified conditional GAN [9] (CGAN) error (Equation 1). This equation represents the Binary Cross Entropy (BCEloss) between the discriminator classification of fake (generated) RIRs and a vector of ones. Meaning that this loss will decrease when the generator succeeds to fool the discriminator. The second part is a mean square error calculated by comparing each sample (s) of the generated RIRs (R_G) with the corresponding ground-truth ones (R_{GT}) given an embedded target room features vector E (Equation 2). The third part is the reverberation time (T_{60}) error between R_G and R_{GT} measured in seconds (Equation 3). The total G_N loss function is calculated by weighting these three parts as written below (Equation 4). The discriminator network (D_N) is trained with a loss function that is same as in FAST-RIR (Equation 5). This equation represents the Binary Cross Entropy (BCEloss) between the discriminator classification and the ground-truth classification. Meaning that the BCEloss of real RIRs is calculated between the discriminator classification of real RIRs and a vector of ones, and the same for fake (generated) RIRs with a vector of zeros.

$$L_{CGAN} = \mathbb{E}_{E\sim P_{dataset}}\left[-\log\left(1 - D_N\left(G_N(E)\right)\right)\right] \tag{1}$$

$$L_{MSE} = \mathbb{E}_{E\sim P_{dataset}}\left[\mathbb{E}\left[\left(R_G(E,s) - R_{GT}(E,s)\right)^2\right]\right] \tag{2}$$

$$L_{T_{60}} = \mathbb{E}_{E\sim P_{dataset}}[|T_{60}(R_G) - T_{60}(R_{GT})|] \tag{3}$$

$$L_{G_N} = L_{CGAN} + \lambda_{\mathrm{MSE}} \mathrm{L}_{\mathrm{MSE}} + \lambda_{T_{60}} L_{T_{60}} \tag{4}$$

$$\begin{aligned} L_{D_N} = {} & \mathbb{E}_{(R_{GT},E)\sim P_{dataset}}\left[-\log\left(D_N\left(R_{GT}(E)\right)\right)\right] \\ & + \mathbb{E}_{E\sim P_{dataset}}\left[-\log\left(1 - D_N\left(G_N(E)\right)\right)\right] \end{aligned} \tag{5}$$

To sum it up, the generator is trained to fool the discriminator (Equation 1) and minimize both the MSE error (Equation 2) and reverberation time error (Equation 3) between generated and corresponding ground-truth RIRs. In parallel, the discriminator is trained to classifies correctly between real and fake RIRs (Equation 5).

Room ID	Room Name	T_{60} Range [second]	MSE	ρ	T_{60} Error [second]
0	Booth	0.17 – 0.19	1.19	0.9	0.47
1	Stairway	0.31 – 0.34	2.57	0.44	0.45
2	Meeting	0.31 – 0.34	2.5	0.92	0.13
3	Lecture	0.35 – 0.39	1.87	0.94	0.05
4	Office	0.43 – 0.46	4.18	0.9	0.04

Table 1: Test performance on objective metrics.

Epoch Range	Learning Rate
0 - 26	2E-05
27 - 34	1.5E-05
35 - 38	1E-05
39 - 45	9.6E-06 – 2E-06

Table 2: The learning rates values during the trainings.

3. EXPERIMENT AND RESULTS

3.1. Baselines

We have found our full loss function, as described in Equation 4 too complicated in relation to our computing resources. Given the CPU[1] we were using, we could not decrease the loss as required within a reasonable time. As a result, we decided to train our model with λ_{MSE} value equals to 5 and λ_{T60} equals to 0. Meaning with a simplified loss function that does not refer to T_{60} error. We will refer to the effect of neglecting T_{60} error later in this chapter.

3.2. Dataset

We generated training dataset of reverberant speech $x_r[t]$ convolving clean speeches $x_c[t]$ from the LibriSpeech test-clean dataset [10] and RIRs r[t] from the Aachen Impulse Response (AIR) database (Equation 6). More information about this database, including the rooms' pictures, available at the AIR page online[2]. Both the clean speeches and the RIRs were recorded at rate of 16 kHz.

$$x_r[t] = x_c[t] * r[t] \tag{6}$$

We chose from the LibriSpeech dataset recordings of 8 different speakers reading sentences of approximately 2 seconds length, and from the AIR database 73 different RIRs of 1-3 seconds length. The RIRs were recorded in 5 different rooms with T_{60} values range of 0.17 to 0.46. Meaning that our training dataset consists of 584 different reverberant speeches. Before the convolving operation, we adapted the RIRs dataset so each RIR size will be 4096, as our generated RIR size. The adaptation was done using sub-sampling and then cropping the last samples that had very small values. Finally, we adapted the convolution result so the reverberant speech $x_r[t]$ will be 3 seconds length, i.e., 48,000 samples at rate of 16 kHz. At this point we recall that in section 2.1.2. it was said that our generator's encoder achieves a receptive field of over 57,000 samples, which equates to 3.58 seconds at 16 kHz. While maintaining FiNS [8] encoder block architecture, we preferred a larger receptive field than the input size (which contains all the input samples), rather than use 11 encoder blocks with a small receptive field of 1.79 seconds. Note that due to resources limitation (as described in section 3.1.) our dataset is relatively small and insufficient for high quality training. But, as we will present later, it is sufficient for a proof of concept.

3.3. Training

We trained the generator network and the discriminator network alternatingly to minimize both the generator loss function L_{GN} (Equation 4) and the discriminator loss function L_{DN} (Equation 5). The model was trained until the discriminator loss increased to a value of approximately 0.5, for both real and fake RIRs classification. Meaning that the generator succeeded to fool him. Satisfying the condition above, the model was trained for 46 epochs with a batch size of 8 using RMSprop optimizer. We found that a good learning rate is in the range [2E-05, 2E-06] and we trained the model as described in Table 2. The learning rates in epochs 39 to 45 decay as cosine function.

3.4. Objective Metrics

In Table 1 we report an objective metrics evaluating our model performance. We divided our training dataset into groups of RIRs by rooms while detailing their corresponding T_{60} values. We evaluate performance of each group by averaging the results on it. We calculated

[1] Intel(R) Core(TM) i5-6200U CPU @ 2.30GHz.
[2] https://www.iks.rwth-aachen.de/en/research/tools-downloads/databases/aachen-impulse-response-database/

	Hardware	Average Time [seconds]
Our Model	CPU[1]	0.37
FAST-RIR	CPU[1]	0.3

Table 3: The runtime results of generating RIRs using our Model and FAST-RIR.

the MSE according to Equation 2, the Pearson correlation coefficient ρ according to Equation 7 (μ and σ are the expectation and the variance respectively), and the reverberation time (T_{60}) error according to Equation 3. It can be noticed that for every group of T_{60} ranges the MSE values are relatively small and the correlation ρ values in 4 out of 5 rooms are relatively close to 1. When it comes to the T_{60} Error, it can be noticed that our model succeeded more in generating RIRs with a larger reverberation time. FAST-RIR [1] T_{60} average error equals to 0.02 seconds, same order of magnitude as our average error when generating RIRs with reverberation time which is greater than 0.35 seconds. In addition, as much as our model estimates RIR with a larger reverberation time, thus the reverberation time error decreases. We estimate that it occurs due to our decision to neglect from the T_{60} error part in our loss function. Referring to the difference between stairways and meeting room RIRs (which has same T_{60} values range). It is noticeable that our model failed to learn well the stairway room which yielded results with a larger T_{60} error and generally low correlation value. We will address this in section 3.6.

$$\rho_{R_G,R_{GT}} = \frac{\mathbb{E}\left[\left(R_G - \mu_{R_G}\right)\left(R_{GT} - \mu_{R_{GT}}\right)\right]}{\sigma_{R_G}\sigma_{R_{GT}}} \tag{7}$$

3.5. Runtime

In Table 3 we represent the runtime results of generating RIRs using our model comparing to FAST-RIR [1]. It can be noticed that due to the adding architecture to our model, the runtime results are slightly higher than in FAST-RIR. The increase is about 23 percent. Despite the increase, our Model and FAST-RIR runtimes have the same order of magnitude. Meaning we can generate RIRs with efficient time complexity as FAST-RIR.

3.6. Embedding Space

In this section we check whether our generator's encoder succeeded in learning rooms' features. We will present an embedding space analysis of our generator's encoder output, meaning the 128-dimensional target room features vector E. In Figure 2, using UMAP [11], we present 2-D projection of 40 vectors that our generator extracts from reverberant speeches inputs. These 40

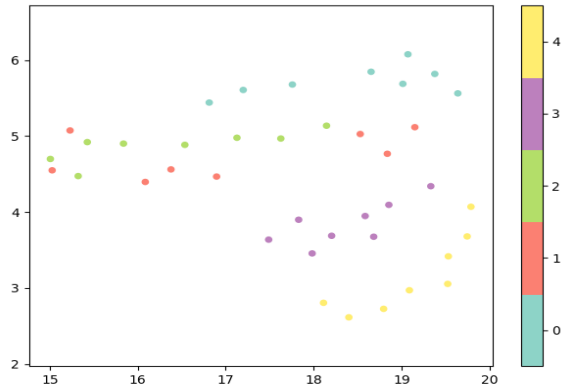

Figure 2: 2-D projection of our target room features vectors (E).

vectors contain features of 5 RIRs (one for each room in our dataset) convolving with 8 different speakers (as described in section 3.2.). The color bar represents the room ID as described in Table 1. Each room in the plot has 8 points, one for each speaker. The plot shows that generally our generator's encoder succeeds separating the rooms by the features it extracts from each reverberant speech recording, regardless of the speaker or the content of the sentence stated in the recording. Referring to the meeting and the stairway rooms (rooms ID 1 and 2) the separation is not as good. We assume that the reason is their same T_{60} values range. Furthermore, regarding the latter, because of the variance in the samples, it is clearly that the features extraction is the least successful. We consider it the main reason for the corresponding bad results as shown in section 3.4.

4. CONCLUSION

We propose a model to estimate room impulse responses (RIRs) of real rooms from a reverberant speech audio file as an input. Our proposed model architecture is based on a combination between FAST-RIR [1] and FiNS [8] architectures. Due to resources limitation, we had trained the model with an insufficient dataset for a small number of epochs, in order to achieve a proof of concept. We have shown that our model succeeds to extract a target room features from a reverberant speech and then estimate RIRs well enough for our purpose. In particular, our model succeeds estimating RIRs with an average reverberation time error of 0.049 seconds when estimating RIRs with a reverberation time greater than 0.35 seconds. Generally, our model succeeds to estimate RIRs with efficient time complexity of 0.37 seconds on average. We believe that training the model with a sufficient dataset for a suitable number of epochs would improve the results significantly.

5. REFERENCES

[1] Ratnarajah, A., Zhang, S.X., Yu, M., Tang, Z., Manocha, D. and Yu, D., 2021. FAST-RIR: Fast neural diffuse room impulse response generator. arXiv preprint arXiv:2110.04057.

[2] Nikunj Raghuvanshi, Rahul Narain, and Ming C. Lin, "Efficient and accurate sound propagation using adaptive rectangular decomposition," IEEE Trans. Vis. Comput. Graph., vol. 15, no. 5, pp. 789–801, 2009.

[3] Shinichi Sakamoto, Ayumi Ushiyama, and Hiroshi Nagatomo, "Numerical analysis of sound propagation in rooms using the finite difference time domain method," The Journal of the Acoustical Society of America, vol. 120, no. 5, pp. 3008–3008, 2006.

[4] Jont B. Allen and David A. Berkley, "Image method for efficiently simulating small-room acoustics," Acoustical Society of America Journal, vol. 65, no. 4, pp. 943–950, Apr. 1979.

[5] J. Y. Wen, E. A. Habets, and P. A. Naylor, "Blind estimation of reverberation time based on the distribution of signal decay rates," in ICASSP, 2008.

[6] C. S. Doire, M. Brookes, P. A. Naylor, D. Betts, C. M. Hicks, M. A. Dmour, and S. H. Jensen, "Single-channel blind estimation of reverberation parameters," in ICASSP, 2015.

[7] T. H. Falk and W.-Y. Chan, "Temporal dynamics for blind measurement of room acoustical parameters," IEEE Tran actions on Instrumentation and Measurement, vol. 59, no. 4, 2010.

[8] Steinmetz, C.J., Ithapu, V.K. and Calamia, P., 2021, October. Filtered Noise Shaping for Time Domain Room Impulse Response Estimation From Reverberant Speech. In 2021 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA) (pp. 221-225). IEEE.

[9] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio, "Generative adversarial nets," in NIPS, 2014, pp. 2672–2680.

[10] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in ICASSP. 2015, pp. 5206–5210, IEEE.

[11] L. McInnes, J. Healy, and J. Melville, "UMAP: Uniform manifold approximation and projection for dimension reduction," arXiv:1802.03426, 2018.