AUDIO TEXTURE SYNTHESIS WITH RANDOM NEURAL NETWORKS: IMPROVING DIVERSITY AND QUALITY

Joseph M. Antognini† * *Matt Hoffman*‡ *Ron J. Weiss*‡

† Whisper AI, Inc.
‡ Google, Inc.

ABSTRACT

Texture synthesis techniques based on matching the Gram matrix of feature activations in neural networks have achieved spectacular success in the image domain. In this paper we extend these techniques to the audio domain. We demonstrate that synthesizing diverse audio textures is challenging, and argue that this is because audio data is relatively low-dimensional. We therefore introduce two new terms to the original Grammian loss: an autocorrelation term that preserves rhythm, and a diversity term that encourages the optimization procedure to synthesize unique textures. We quantitatively study the impact of our design choices on the quality of the synthesized audio by introducing an audio analogue to the Inception loss which we term the VGGish loss. We show that there is a trade-off between the diversity and quality of the synthesized audio using this technique. Finally we perform a number of experiments to qualitatively study how these design choices impact the quality of the synthesized audio.

Index Terms— Neural networks, texture synthesis

1. INTRODUCTION

Texture synthesis has been studied for over fifty years [1]. The problem is to take a sample of some textured data (usually an image) and generate synthesized data which have the same texture, but are not identical to the original sample. This problem is interesting as a machine learning problem in its own right, but successful texture synthesis methods can also elucidate the way in which humans perceive texture [2].

Portilla and Simoncelli [3] pioneered a very successful approach to image texture synthesis that tries to find a complete set of statistics that describe the perceptually relevant aspects of a given texture. To synthesize a new texture, a random input is perturbed until its statistics match those of the targets. Portilla and Simoncelli [3] developed a set of four classes of statistics consisting of 710 parameters which produced extremely realistic images of natural and synthetic textures. McDermott and Simoncelli [2] used a similar approach to develop four classes of statistics in a cochlear model to synthesize textural audio spectrograms. This work produced convincing audio data for many natural audio textures (e.g., insects in a swamp, a stream, applause), but had difficulty with pitched and rhythmic textures (e.g., wind chimes, walking on gravel, church bells).

Gatys et al. [4] introduced an extremely successful technique that replaced hand-crafted statistics with Gram-matrix statistics derived from the hidden feature activations of a trained convolutional neural network (CNN). By perturbing a random input to match these Gram matrices, Gatys et al. [4] produced compelling textures that were far more complicated than those achieved by any earlier work.

*Work done as a Google AI resident.

Given the success of Gatys et al. [4] in the image domain relative to the hand-crafted approach of Portilla and Simoncelli [3], it is natural to ask whether a similar CNN-based strategy could be adapted to the audio domain to build on the hand-crafted approach of McDermott and Simoncelli [2]. Ulyanov and Lebedev [5] proposed just such an extension of the approach of [4] to audio. Their basic approach works fairly well on many of the 15 examples they consider, but (as we demonstrate in Sec. 3) it has some of the same failure modes as the approach of McDermott and Simoncelli [2].

In this work, we examine the causes of these problems, analyze why they are more serious in the audio domain than in the image domain, and propose techniques to fix them.

2. METHODS

2.1. Signal processing

We produce audio textures by transforming the target audio to a log spectrogram and synthesizing a new spectrogram. We then use the Griffin-Lim algorithm [6] to invert the spectrogram and generate the synthesized audio texture. If necessary, we resample the target audio to 16 kHz and normalize. We produce a spectrogram by taking the absolute value of the short-time Fourier transform with a Hann window of size 512 samples and a hop size of 64 samples. Although taking the absolute value removes any explicit phase information, if the hop size is less than or equal to half the window size phase information is implicitly retained (i.e., there exists a unique audio signal corresponding to such a spectrogram up to a global phase; [7]). We then add 1 to every magnitude in the spectrogram and take the natural logarithm. Adding 1 guarantees that the log-spectrogram is finite and positive.

2.2. Architecture of the neural networks

We obtained the best textures with a set of six single-hidden-layer random CNNs. Unlike the case of image texture synthesis, audio spectrograms are one-dimensional so we therefore use a one-dimensional convolution. Each CNN had a convolutional kernel with a different width, varying in powers of 2 from 2 to 64 frames. We applied a ReLU activation after the convolutional layers. Each layer had 512 filters randomly drawn using the Glorot initialization procedure [8]. Several authors have found that random convolutional layers perform as well as trained convolutional layers for image texture synthesis [9, 10]. Shu et al. [11] furthermore showed that a random CNN retains as much information to reconstruct an image as a trained convolutional network, if not more. Although we also tried synthesizing textures with an audio model that was trained on AudioSet [12], a dataset consisting of about one million 10 second audio clips with

527 labels, we did not find that this trained model produced textures that were any better than those produced by a random CNN.

Using an ensemble of CNNs with varying kernel sizes is crucial for obtaining high quality textures since each kernel size is most sensitive to audio features whose duration is comparable to the kernel size. The features of real-world audio can span many different timescales (e.g., just a few milliseconds for a clap and up to several seconds for a bell) so it is important to use an architecture which is sensitive to the range of timescales that is likely to be encountered. We consider the impact of our architecture design choices experimentally in Section 3.3.

2.3. Loss terms

The loss we minimize consists of three terms:

$$\mathcal{L} = \mathcal{L}_{\text{Gram}} + \alpha \mathcal{L}_{\text{autocorr}} + \beta \mathcal{L}_{\text{div}}. \tag{1}$$

The first term, $\mathcal{L}_{\text{Gram}}$, was introduced by Gatys et al. [4] and is intended to capture the average local correlations between features in the texture. The second term, $\mathcal{L}_{\text{autocorr}}$, we adapt from Sendik and Cohen-Or [13] and is intended to capture rhythm. The final term, $\mathcal{L}_{\text{div}}$, we introduce to prevent the optimization process from exactly copying the original texture. The hyperparameters α and β are used to set the relative importance of these three terms. We find that $\alpha = 10^3$ and $\beta = 10^{-4}$ work well for many of the textures we studied, although hyperparameter tuning is sometimes required. In particular, highly rhythmic textures generally require a larger choice of α and a lower choice of β.

2.3.1. Gram loss

Let us write the features of the kth convolutional network as $F^k_{t\mu}$, where t indicates the position of a patch in the feature map (i.e., the time in the spectrogram), and μ indicates the filter. The Gram matrix for the kth convolutional network is the time-averaged outer product between the kth feature map with itself:

$$G^k_{\mu\nu} = \frac{1}{T} \sum_t F^k_{t\mu} F^k_{t\nu}, \tag{2}$$

where T is the number of windows in the spectrogram

We match this statistic by minimizing the Frobenius norm of the difference between the Gram matrices of the synthesized texture and the target for all layers and normalizing to the Frobenius norm of the target texture Gram matrix:

$$\mathcal{L}_{\text{Gram}} = \frac{\sum_{k,\mu,\nu} \left(G^k_{\mu\nu} - \widetilde{G}^k_{\mu\nu}\right)^2}{\sum_{k,\mu,\nu} \left(\widetilde{G}^k_{\mu\nu}\right)^2}. \tag{3}$$

Throughout this paper tilde denotes the target texture.

2.3.2. Autocorrelation loss

While minimizing the Gram loss alone produces excellent audio for many kinds of audio textures, we show in Sec. 3.2.1 that the Gram loss fails to capture rhythm. To this end, we adapt a loss term introduced by Sendik and Cohen-Or [13] derived from the autocorrelation of the feature maps that was developed to capture periodic structure in image textures.[1] The autocorrelation of the kth feature map is

$$A^k_{\tau\mu} = \mathscr{F}^{-1}_f \left[\mathscr{F}_t[F^k_{t\mu}] \, \mathscr{F}_t[F^k_{t\mu}]^* \right], \tag{4}$$

where $\mathscr{F}_t$ represents the discrete Fourier transform with respect to time t, * represents complex conjugation, and τ represents the lag. The autocorrelation loss is the sum of the normalized Frobenius norms of the squared differences between the target and synthesized autocorrelation maps:

$$\mathcal{L}_{\text{autocorr}} = \frac{\sum_{k,\tau,\mu} \left(A^k_{\tau\mu} - \widetilde{A}^k_{\tau\mu}\right)^2}{\sum_{k,\tau,\mu} \left(\widetilde{A}^k_{\tau\mu}\right)^2}. \tag{5}$$

We generally do not expect to encounter rhythmic structure on timescales longer than a few seconds, and autocorrelations on extremely short timescales (under 200 ms) are captured within the receptive fields of individual networks. Including very short and long lags in the loss tends to encourage overfitting without adding any useful rhythmic activity to the texture (this is particularly true for lags near 0 since the autocorrelation will always be largest there and will therefore be the largest contributor to $\mathcal{L}_{\text{autocorr}}$). For this reason we only sum over lags of 200 ms to 2 s.

2.3.3. Diversity loss

As we show in Sec. 3.2.2, a downside of using the previous two loss terms alone is that they tend to reproduce the original texture exactly. Sendik and Cohen-Or [13] proposed a diversity term for image texture synthesis of the form

$$\mathcal{L}_{\text{Sendik}} = -\sum_{k,t,\mu} \left(F^k_{t\mu} - \widetilde{F}^k_{t\mu}\right)^2, \tag{6}$$

which is maximized when the two feature maps match exactly. We found that this diversity term has two shortcomings: first, because this term can become arbitrarily negative, it can dominate the total loss and destabilize the optimization; second, we find that this loss has a tendency to reproduce the original input, but slightly shifted in time (see Fig. 1). To address these two issues, we propose the following shift-invariant diversity term:

$$\mathcal{L}_{\text{div}} = \max_s \left(\frac{\sum_{k,t,\mu} \left(\widetilde{F}^k_{t\mu}\right)^2}{\sum_{k,t,\mu} \left(F^k_{t+s,\mu} - \widetilde{F}^k_{t\mu}\right)^2} \right), \tag{7}$$

where the shift s can take on values ranging from 0 to $T-1$. In other words, we compute the negative inverse of the diversity term of Eq. 6 for all possible relative shifts between the original and synthesized textures and then take the maximum. Since computing this loss for all possible shifts is computationally expensive, we compute this loss in steps of 50 frames, cycling through different sets of frames in each step of the optimization process, along with computing the loss for the shifts which yielded the largest loss in the last 10 optimization steps.

2.4. Optimization

We find that L-BFGS-B [14] works well to minimize the loss and obtain high quality audio textures. We optimized for 2000 iterations and used 500 iterations of the Griffin-Lim algorithm. We furthermore found it useful to include the diversity loss term for only the first 100 iterations; by this point the optimizer had found a nontrivial local optimum, and continuing to incorporate the diversity loss reduced texture quality. Audio and spectrograms for our synthesized textures, along with supplementary information and figures for our experiments can be found at `https://antognini-google.github.io/audio_textures/`.

[1] Note that Sendik and Cohen-Or [13] use a variant of the feature map autocorrelation called the structural matrix, but we find that the autocorrelation works well and is faster to compute.

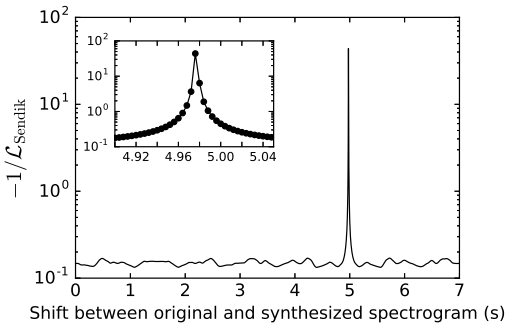

Fig. 1. The inverse negative diversity term used by Sendik and Cohen-Or [13] as a function of a synthesized texture shifted in time. The synthesized texture closely matches the original texture, but is shifted in time by about five seconds. $\mathcal{L}_{\text{Sendik}}$ fails to capture this effect as demonstrated by the sharp peak. Inset zooms in on the peak to show that the peak is resolved.

3. EXPERIMENTS

3.1. Quantitative evaluation of texture quality

Quantitatively evaluating the quality of generative models is difficult. Salimans [15] developed a useful quantitative metric for comparing generative adversarial networks based on the Kullback-Liebler divergence of the label predictions given by the Inception classifier [16] between the sampled images and the original dataset. We adapt this "Inception score" to assess the quality of our audio textures compared to other methods. Rather than Inception, we use the "VGGish" CNN[2] that was trained on AudioSet.[3] The motivation for our "VGGish score" is that the label predictions produced by the VGGish model should match between the original and synthesized textures. To this end, we define the score as

$$\mathcal{S}_{\text{VGGish}} \equiv \exp\left[\mathbb{E}_x\left[\text{KL}\left(p_{\text{VGGish}}(y|\widetilde{x}) \,||\, p_{\text{VGGish}}(y|x)\right)\right]\right], \quad (8)$$

where y represents the VGGish label predictions and x represents the texture audio. We compute $\mathcal{S}_{\text{VGGish}}$ over the 168 textures used by McDermott and Simoncelli [2]. These textures span a broad range of sound, including natural and artificial sounds, pitched and non-pitched sounds, and rhythmic and non-rhythmic sounds. We compare this VGGish score between our models optimized with different loss terms and the approaches used by Ulyanov and Lebedev [5] and McDermott and Simoncelli [2] in Table 1, separating out the scores for pitched and rhythmic textures. We also compare an autocorrelation score and a diversity score computed from the generated spectrograms discussed in Sections 3.2.1 and 3.2.2, respectively.

The best VGGish scores are obtained by using $\mathcal{L}_{\text{Gram}}$ alone. As expected, adding $\mathcal{L}_{\text{autocorr}}$ substantially reduces the autocorrelation score, though at the cost of increasing the diversity score, and adding a larger weight to $\mathcal{L}_{\text{div}}$ generally reduces the diversity score. The lowest diversity scores are obtained by McDermott and Simoncelli [2], though at the cost of substantially higher autocorrelation scores and relatively large VGGish scores for pitched textures.

It is unsurprising that adding $\mathcal{L}_{\text{div}}$ reduces the VGGish score because introducing any diversity will generally reduce the VGGish score (the model could achieve a perfect VGGish score simply by copying the original input). It is, however, surprising that adding $\mathcal{L}_{\text{autocorr}}$ alone also reduces the VGGish score. Although introducing $\mathcal{L}_{\text{autocorr}}$ qualitatively seems to increase overfitting, this overfitting occurs on very long timescales (i.e., the model will reproduce several seconds that sound very similar to the original audio). Introducing $\mathcal{L}_{\text{autocorr}}$ seems to make the optimization process more difficult for timescales much shorter than the minimum lag considered by $\mathcal{L}_{\text{autocorr}}$, which leads to lower quality on short timescales and thus higher VGGish scores.

3.2. Effect of the different loss terms

3.2.1. Autocorrelation loss

To demonstrate the necessity of $\mathcal{L}_{\text{autocorr}}$ we synthesize textures with a variety of values of α. For simplicity we set $\beta = 0$ in these experiments (i.e., we exclude $\mathcal{L}_{\text{div}}$ from the total loss). If the weight of $\mathcal{L}_{\text{autocorr}}$ is small, the synthesized textures reproduce tapping sounds which lack the precise rhythm of the original. Only when α is sufficiently large is the rhythm reproduced. We compute the squared loss between the autocorrelation of each synthesized texture and its target texture, normalized to the Frobenius norm of the autocorrelation of the target texture and present these scores in Table 1. Qualitatively we find that, as expected, $\mathcal{L}_{\text{autocorr}}$ is most important in textures with substantial rhythmic activity and so it is useful to use a relatively large value for α for these rhythms. For textures without substantial rhythmic activity we find that a smaller choice of α produces higher quality textures.

3.2.2. Diversity loss

To demonstrate the effect of $\mathcal{L}_{\text{div}}$ we synthesize textures with a variety of values of β, keeping α fixed to 10^3. Smaller values of β generally reproduce the original texture but shifted in time (about 2 s for the wind chimes and about 3.25 s for speech). Larger values of β produce spectrograms which are not simple translations of the original input, but the quality of the resulting audio is much lower. In the case of the wind chimes the chimes do not have the hard onset in the original, and in the case of speech the voice is echoey and superimposes different phonemes. This is an instance of a more general diversity-quality trade-off in texture synthesis. In Fig. 2 we show the VGGish score (a rough proxy for texture quality) vs. the weight on the diversity term. As the weight on the diversity term increases, the average quality decreases. We furthermore calculate the diversity loss on the spectrograms themselves to get a diversity score and present these scores in Table 1.

We find that it is crucial to tune the loss weights for different texture classes in order to obtain the highest quality textures. Large α and large β, for example, is especially important for reliably generating rhythmic textures. Pitched audio generally requries a smaller choice of α and β. For non-textured audio like speech and music, high quality audio is only obtained with a large α and small β, which will only reproduce the original with some shift; since these kinds of audio do not obey the assumptions required for texture synthesis (i.e., stationary statistics over long time periods), any set of weights that does not reproduce the original will produce low quality audio.

[2]Available from `https://github.com/tensorflow/models/tree/master/research/audioset`.

[3]VGGish produces 128 dimensional embeddings rather than label predictions. To obtain label predictions we trained a set of 527 logistic regression classifiers on top of the AudioSet embeddings (AudioSet's 527 classes are not mutually exclusive.) We trained for 100,000 steps with a learning rate of 0.1 and achieved a test accuracy of 99.42% and a test cross entropy loss of 0.0584.

Table 1. A comparison of scores between our model and other work.

	VGGish ($\times 10^{-4}$)			Autocorrelation			Diversity		
	Rthm.	Ptch.	Other	Rthm.	Ptch.	Other	Rthm.	Ptch.	Other
Spectrograms recovered via Griffin-Lim	9.7	12.6	7.1	7.4	0.54	2.9	21.4	29.7	22.7
McDermott and Simoncelli [2]	16.7	33.2	8.3	542.0	408.1	421.9	**1.6**	**1.6**	**2.0**
Ulyanov and Lebedev [5]	13.4	26.8	10.0	40.6	23.3	27.4	2.9	3.0	3.3
$\mathcal{L}_{\text{Gram}}$	**9.9**	**16.8**	**7.3**	29.0	9.7	6.5	2.4	3.0	3.5
$\mathcal{L}_{\text{Gram}} + \mathcal{L}_{\text{autocorr}}$	17.8	21.3	17.9	13.3	7.4	15.6	3.4	5.4	5.0
$\mathcal{L}_{\text{Gram}} + \mathcal{L}_{\text{autocorr}} + \mathcal{L}_{\text{div}}$ ($\beta = 10^{-5}$)	14.5	23.0	12.2	13.0	**2.3**	7.2	3.8	6.8	4.4
$\mathcal{L}_{\text{Gram}} + \mathcal{L}_{\text{autocorr}} + \mathcal{L}_{\text{div}}$ ($\beta = 10^{-3}$)	14.9	19.0	10.0	**4.7**	3.7	**7.1**	5.0	4.9	3.9

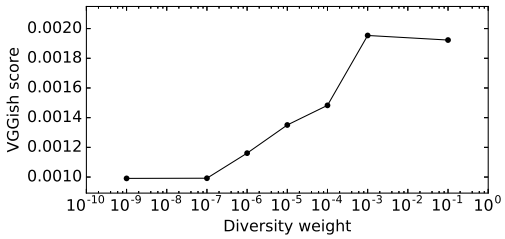

Fig. 2. The diversity-quality trade-off in texture synthesis. The VGGish score is a rough proxy for texture quality, with lower scores representing higher quality textures. As the diversity weight increases, the average quality of the textures decreases.

3.3. Neural network architecture

The receptive field size of the convolutional kernel has a strong effect on the quality and diversity of the synthesized textures. We experiment with the effect of changing the receptive field size for two textures by using the same set of single layer CNNs with exponentially increasing kernel sizes, but varying the maximum kernel size from 2 frames to 8. We find that CNNs with very small receptive fields produce novel, but poor-quality textures that fail to capture long-range structure. Networks with large receptive fields tend to reproduce the original. This is an example of the quality-diversity trade-off in texture synthesis.

Another design choice we consider is the number of filters in the each network. We experimented with using 32, 128, and 512 filters to synthesize two textures. Note that because there are six CNNs in all with varying kernel sizes, the total number of activations varies from 192 to 3072. We find that at least 128 filters are necessary to get reasonable textures, but the quality continues to improve with 512 filters.

We also considered stacking six convolutional layers on top of each other, each with a receptive field of 2 and separated by an average pooling layer with a pool size of 2 and a stride of 2. This network has the same distribution of receptive field sizes as the six separate networks, but the input to each layer here must pass through the (random) filters of all the earlier layers. We compared audio generated with this network to the six separate networks that we use elsewhere but find that the only effect of stacking the layers is a modest degradation in the quality of long-range sounds, best exemplified in the wind chimes texture.

4. CONCLUSIONS

We have demonstrated that the approach to texture synthesis described by Gatys et al. [4] of matching Gram matrices from convolutional networks can be extended to the problem of synthesizing audio textures. There are, however, certain differences in the audio domain vs. the image domain that require the addition of two more loss terms to produce diverse, robust audio textures: an autocorrelation term to preserve rhythm, and a diversity term to encourage the synthesized textures to not exactly reproduce the original texture. We test our technique across several classes of textures like rhythmic and pitched audio and find that tuning the weights on the autocorrelation and diversity terms is crucial to obtaining the highest quality textures for different classes. The choice of architecture is also important to obtain high quality textures; an ensemble random convolutional neural networks with a wide range of receptive field sizes allows the model to capture features that occur on timescales across many orders of magnitude. Finally, we show that this method has a trade-off between the diversity and the quality of the results.

Acknowledgments

The authors are grateful to Josh McDermott for providing the synthesized textures using the technique of McDermott and Simoncelli [2] for comparison with the technique in this paper. The authors thank Rif A. Saurous for helpful comments on the manuscript.

References

[1] Bela Julesz, "Visual pattern discrimination," *IRE Transactions on Information Theory*, vol. 8, no. 2, pp. 84–92, 1962.

[2] Josh H McDermott and Eero P Simoncelli, "Sound texture perception via statistics of the auditory periphery: evidence from sound synthesis," *Neuron*, vol. 71, no. 5, pp. 926–940, 2011.

[3] Javier Portilla and Eero P Simoncelli, "A parametric texture model based on joint statistics of complex wavelet coefficients," *International Journal of Computer Vision*, vol. 40, no. 1, pp. 49–70, 2000.

[4] Leon Gatys, Alexander S Ecker, and Matthias Bethge, "Texture synthesis using convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 262–270.

[5] Dmitry Ulyanov and Vadim Lebedev, "Audio texture synthesis and style transfer," 2016.

[6] Daniel Griffin and Jae Lim, "Signal estimation from modified short-time fourier transform," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 2, pp. 236–243, 1984.

[7] Nicolas Sturmel and Laurent Daudet, "Signal reconstruction from STFT magnitude: A state of the art," in *International Conference on Digital Audio Effects*, 2011.

[8] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

[9] Kun He, Yan Wang, and John Hopcroft, "A powerful generative model using random weights for the deep image representation," in *Advances in Neural Information Processing Systems*, 2016, pp. 631–639.

[10] Ivan Ustyuzhaninov, Wieland Brendel, Leon A Gatys, and Matthias Bethge, "Texture synthesis using shallow convolutional networks with random filters," *arXiv preprint arXiv:1606.00021*, 2016.

[11] Yao Shu, Man Zhu, Kun He, John Hopcroft, and Pan Zhou, "Understanding deep representations through random weights," *arXiv preprint arXiv:1704.00330*, 2017.

[12] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *IEEE ICASSP*, 2017.

[13] Omry Sendik and Daniel Cohen-Or, "Deep correlations for texture synthesis," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 5, pp. 161, 2017.

[14] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal, "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization," *ACM Transactions on Mathematical Software (TOMS)*, vol. 23, no. 4, pp. 550–560, 1997.

[15] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen, "Improved techniques for training GANs," in *Advances in Neural Information Processing Systems*, 2016, pp. 2234–2242.

[16] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al., "Going deeper with convolutions," in *CVPR*, 2015.