Gómez Mellado, Esteban; Vali, Mohammadhassan; Bäckström, Tom

Low-complexity Real-time Neural Network for Blind Bandwidth Extension of Wideband Speech

Published in:
31st European Signal Processing Conference, EUSIPCO 2023 - Proceedings

DOI:
10.23919/EUSIPCO58844.2023.10290072

Published: 04/09/2023

Document Version
Peer-reviewed accepted author manuscript, also known as Final accepted manuscript or Post-print

Please cite the original version:
Gómez Mellado, E., Vali, M., & Bäckström, T. (2023). Low-complexity Real-time Neural Network for Blind Bandwidth Extension of Wideband Speech. In *31st European Signal Processing Conference, EUSIPCO 2023 - Proceedings* (pp. 31-35). (European Signal Processing Conference). European Association For Signal and Image Processing. https://doi.org/10.23919/EUSIPCO58844.2023.10290072

Low-complexity Real-time Neural Network for Blind Bandwidth Extension of Wideband Speech

Esteban Gómez*†, Mohammad Hassan Vali*, Tom Bäckström*

*Department of Information and Communications Engineering, Aalto University, Espoo, Finland
†Voicemod S.L., Valencia, Spain
{esteban.gomezmellado, mohammad.vali, tom.backstrom}@aalto.fi

***Abstract*—Speech is streamed at 16 kHz or lower sample rates in many applications (e.g. VoIP, Bluetooth headsets). Extending its bandwidth can produce significant quality improvements. We introduce BBWEXNet, a lightweight neural network that performs blind bandwidth extension of speech from 16 kHz (wideband) to 48 kHz (fullband) in real-time in CPU. Our low latency approach allows running the model with a maximum algorithmic delay of 16 ms, enabling end-to-end communication in streaming services and scenarios where the GPU is busy or unavailable. We propose a series of optimizations that take advantage of the U-Net architecture and vector quantization methods commonly used in speech coding, to produce a model whose performance is comparable to previous real-time solutions, but approximately halving the memory footprint and computational cost. Moreover, we show that the model complexity can be further reduced with a marginal impact on the perceived output quality.**

***Index Terms*—Bandwidth extension, speech processing, real-time, deep learning**

I. Introduction

A sample rate of 16 kHz (also known as wideband) continues to be a common choice in many speech applications such as VoIP or videoconferencing systems. Moreover, the bandwidth of certain Bluetooth profiles found in commercial hardware such as Hands-Free Profile (HFP) or Headset Profile (HSP) may be limited to this sample rate or even lower (e.g. 8 kHz, also known as narrowband).

Wideband audio is considered sufficient to capture the most important features of speech, allowing end-to-end communication without significant intelligibility loss [1], [2]. However, contradicting evidence has been found, showing that consonant identification is harder when speech information above 8 kHz (i.e. wideband's Nyquist frequency) is removed [3]. Additionally, spectral peaks of fricatives may be found between 8 kHz and 9 kHz [4]. In terms of quality, perceptual scores associated with speech naturalness substantially drop when the cutoff of speech utterances is reduced from 10.9 kHz to 7 kHz [5]. For these reasons, wideband speech quality is often described as "muffled", "muddy" or "mushy" [6].

These degradations can be mitigated by increasing the sample rate to a higher one such as 32 kHz (superwideband) or 48 kHz (fullband). However, it is not always possible due to system constraints (e.g. limited maximum bandwidth). In these cases, it is possible to artificially reconstruct the missing bandwidth to maximize the output quality. Such a task is known as *bandwidth extension* and sometimes referred to as *audio super resolution*, due to its image counterpart [7]. If no side information is used other than the input speech itself, it is known as *blind bandwidth extension*.

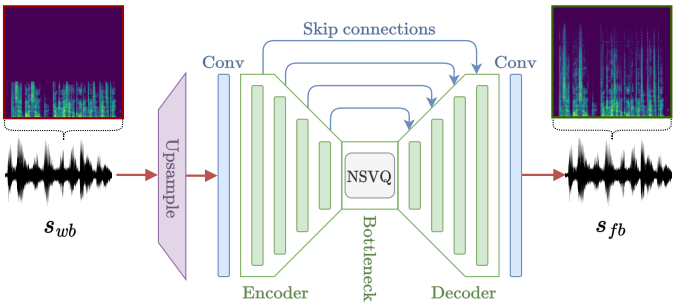

Fig. 1. BBWEXNet overview. First, the wideband input s_{wb} is upsampled to match the expected output dimensions. Then, it is fed to the network. The resulting output corresponds to s_{fb}, the fullband estimate of s_{wb}.

Formulating a rule-based solution to extend the bandwidth of an arbitrary audio containing speech is a challenging task because multiple candidates can be proposed. Neural networks have demonstrated to have the capacity to estimate sensible alternatives of the missing high frequency content in a data-driven manner [8]–[12]. In our work, we aimed to extend the bandwidth of an arbitrary speech stream from wideband to fullband in real-time. We restrict ourselves to low latency on-device processing using only CPU, because we target consumer devices. These may present a high variability in GPU specifications or it may not be available at all. In cases such as videoconferencing, online gaming or live streaming, a significant amount of GPU resources may already be busy, thus making any assumption about free resources difficult to anticipate. Our proposed **b**lind **b**and**w**idth **ex**tension **net**work (**BBWEXNet**) outlined in Fig. 1 and detailed in II-A, leverages the U-Net architecture [13] together with vector quantization techniques commonly used in speech coding [14] and learnable upsampling methods [15] to produce a low-complexity model that meets these requirements. Our main contributions can be summarized as follows:

- First, we propose a causal model capable of extending speech from wideband to fullband in real-time that approximately halves the amount of parameters and esti-

mated operations per inference compared to our chosen baseline while preserving its output quality.
- Secondly, we show that vector quantization can be successfully used to model the network's bottleneck distribution with a small codebook.
- Finally, we propose alternatives to reduce the model complexity even further with a marginal trade off in the perceived output quality.

An accompanying demo page is provided[1].

II. Method

The goal of blind bandwidth extension is to simultaneously map a source audio sampled at sample rate f_l to a target sample rate f_h ($f_h > f_l$) and to reconstruct the missing spectral content between $f_l/2$ and $f_h/2$ (i.e. their respective Nyquist frequencies) as if the source audio was originally recorded at f_h. In our case we want to map the source wideband audio s_{wb} (sampled at $f_l = 16\,\text{kHz}$), to a target fullband audio s_{fb} (sampled at $f_h = 48\,\text{kHz}$). The upsampling ratio between s_{fb} and s_{wb} is $f_h{:}f_l = 3{:}1$, meaning that for a given time interval in seconds, s_{fb} will have three times the amount of samples of s_{wb}, thus, the network must estimate $66,\overline{6}\%$ of the total data. As a first step, it is necessary to match the dimensions of s_{wb} and s_{fb}. To achieve this, we upsample s_{wb} as a preprocessing step. Since we need to upsample by an integer ratio, we implemented a causal version of sinc interpolation [16] to to use it in real-time. The upsampled input waveform is then fed to the model. The whole process can be expressed as

$$s_{fb} = \mathcal{T}\left(U\left(s_{wb}\right), \lambda\right), \tag{1}$$

where U is an upsampling function, λ is an arbitrary number of parameters, and $\mathcal{T}$ is the bandwidth extension function learned by the neural network. It is important to notice that for a given input, an infinite number of candidates can be proposed. By restricting the input to speech signals, we can narrow down our search and train the network to exploit the commonalities of signals in this domain.

A. Model architecture

Our model is based on the Streaming SEANet architecture which we will refer to as our baseline model [9]. It is a symmetrical encoder-decoder U-Net with skip connections. An additional plain convolutional layer is added before the encoder and after the decoder. The encoder has four convolutional blocks that follow a downsampling scheme of 2:2:8:8 controlled by the stride size. This pattern is replicated by the decoder in the reverse order using transposed convolutions. In each encoder downsampling step, the number of convolutional channels is increased by a factor of 2. Conversely, in each decoder upsampling step, this number is halved.

In our model, we replaced the transposed convolutions of the decoder blocks by *SampleShuffle* blocks that follow the same upsampling pattern. We did this to prevent checkerboard artifacts that are commonly associated with transposed convolutions [17], [18] and to reduce the model complexity by using a smaller kernel size in the corresponding convolutional layers. The structure of this block is explained in II-B.

[1]https://eagomez2.github.io/bbwexnet/

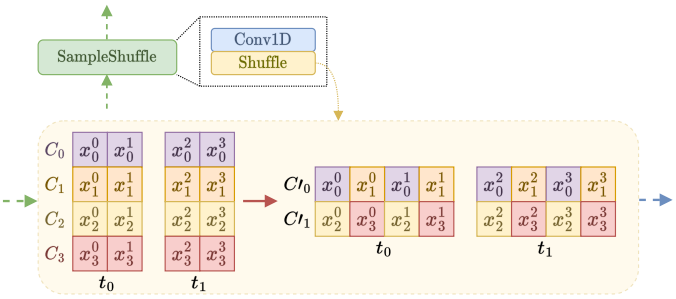

Fig. 2. *SampleShuffle* block. Input channels C_0 to C_3 are interleaved to generate output channels $C\prime_0$ and $C\prime_1$ for time steps t_0 and t_1. Each feature is represented as x_i^j where i is the channel index and j is the feature index. In this example $u_f = 2$.

The baseline bottleneck contains two convolutional layers with a kernel size of 7 to process the features learned by the encoder. We replaced it by a single noise substitution in vector quantization (NSVQ) block [14] to model the bottleneck output distribution. We hypothesize that this module can have a similar capacity using fewer computations. Additionally, NSVQ provides faster convergence than plain vector quantization (VQ) used with a straight-through estimator [19]. This block is explained in II-C.

B. Sample shuffling

Similarly to [15], we replaced all transposed convolutional layers in the decoder by a 1D version of the *PixelShuffle* upsampling layer. We called it *SampleShuffle* to differentiate it from its 2D image counterpart. Each *SampleShuffle* block is composed of a shuffle function expressed as a combination of reshaping, transposing and concatenation operations, followed by a 1D convolution of kernel size 1. The role of the shuffle function is to interleave the features of different convolutional channels in such a way that after applying it, the number of convolutional channels is decreased by an upsampling factor u_f, and the number of features per channel is multiplied by the same factor. Subsequently, the 1D convolution adapts the number of channels to the required dimensions of the following decoder block and its corresponding skip connection. In order to maintain causality, we group convolutional channels in u_f groups.

As an example, if we have 4 input convolutional channels $\{C_0, C_1, C_2, C_3\}$ and we need to upsample the output by a factor of $u_f = 2$, then, the *SampleShuffle* block will create 2 groups of 2 input channels each $\{(C_0, C_1), (C_2, C_3)\}$ to produce 2 output channels $\{C\prime_0, C\prime_1\}$. $C\prime_0$ will be formed by interleaving C_0 and C_1, and $C\prime_1$ will be formed by interleaving C_2 and C_3 as shown in Fig. 2. This guarantees that producing a frame by frame inference in real-time yields the same results as computing the inference over a whole audio file at once.

C. Vector quantization

Vector quantization is a data compression technique similar to k-means that is frequently used in, for example, speech coding. It models the probability density function of a distribution by a set of template vectors called *codebook* [20]. As one prevalent application of VQ, it is used in vector quantized variational autoencoders (VQ-VAE) to capture an abstract high-level representation of the latent space (bottleneck distribution) [21].

In our model, we use the NSVQ to optimize the VQ module. Getting intuition from reparameterization trick [22], NSVQ simulates the VQ error by adding noise to the input vector, such that the simulated noise gains the shape of the original VQ error distribution. In addition, since NSVQ does not add any additional loss term to the global optimization loss function, it does not incur any hyperparameter tuning (weight coefficient for VQ module) in the training phase.

One of the main challenges of using vector quantization in machine learning optimization is the so-called *codebook collapse* [23]. When this happens, a subset of the codebook is no longer used, yet the computational burden remains intact. To solve this problem, we adopted the *codebook replacement* technique proposed in [14], in which we replace inactive codebook elements by a perturbation of active elements.

III. Experiments setup

A. Dataset

We performed all our experiments using the VCTK dataset [24]. It contains data corresponding to 110 English speakers of various accents reading about 400 sentences each. All audios are provided as mono files recorded at 48 kHz with a resolution of 16-bits. To accommodate it to our needs, all the active speech regions in the dataset were extracted and concatenated to form samples of 5 seconds of speech, thus, a single sample could contain utterances of one or more speakers. This way, we could ensure that the model is focused on speech content rather than silent sections. By using a fixed file duration, we avoid the need of padding batches of data. We split our data into 80% for training, 10% for validation and 10% for testing, totalling 28 hours of fullband speech.

B. Training

We trained 6 different models including the baseline. Each model was trained for 500 epochs using the same generative adversarial network (GAN) configuration, optimizer and training objectives as in [9] to be able to compare our models against the baseline. The model to be trained acts as the generator G and it is jointly optimized with a convolutional multi-scale waveform discriminator composed of three discriminator instances D_0, D_1 and D_2 that share the same architecture but do not share parameters. The learning objective for the generator G is the weighted combination $\alpha\mathcal{L}_G^{\text{adv}}+\beta\mathcal{L}_G^{\text{dfm}}$, where $\alpha = 1$ and $\beta = 100$. $\mathcal{L}_G^{\text{adv}}$ is the adversarial loss and $\mathcal{L}_G^{\text{dfm}}$ is the deep feature matching loss. These terms are defined by (2) and (3), respectively as

$$\mathcal{L}_G^{\text{adv}} = \mathbb{E}_x \left[\frac{1}{K} \sum_{k,t} \frac{1}{T_k} \max\left(0, 1 - D_{k,t}\left(G\left(x\right)\right)\right) \right], \quad (2)$$

$$\mathcal{L}_G^{\text{dfm}} = \mathbb{E}_x \left[\frac{1}{KL} \sum_{k,l} \frac{1}{T_{k,l}} \sum_{t} \left| D_{k,t}^{(l)}\left(y\right) - D_{k,t}^{(l)}\left(G\left(x\right)\right) \right| \right], \quad (3)$$

where $\mathbb{E}$ is the expected value, k is the discriminator index that goes from 0 to $K-1=2$, t is the time step index from 0 to $T-1$, and l is the discriminator layer index that goes from 0 to $L-2$. The discriminator loss $\mathcal{L}_D$ is defined as

$$\begin{aligned} \mathcal{L}_D = \quad & \mathbb{E}_y \left[\frac{1}{K} \sum_{k} \frac{1}{T_k} \sum_{t} \max\left(0, 1 - D_{k,t}\left(y\right)\right) \right] + \\ & \mathbb{E}_x \left[\frac{1}{K} \sum_{k} \frac{1}{T_k} \sum_{t} \max\left(0, 1 + D_{k,t}\left(G\left(x\right)\right)\right) \right], \end{aligned} \quad (4)$$

where $\mathbb{E}$ is the expected value, k is the discriminator index that goes from 0 to $K-1=2$, and t is the time index that goes from 0 to $T-1$.

In each case, the model is fed with a batch of 8 samples from the dataset that are downsampled to 16 kHz and upsampled back to 48 kHz in place. This way, we match the expected output dimensions with an input that has no content above 8 kHz other than potential upsampling artifacts. We also trained 2 additional models where the input is upsampled to 32 kHz instead. This way, we can assess the perceptual importance of the spectral content above 16 kHz as we hypothesize that targeting 32 kHz instead of 48 kHz could offer a convenient trade off in terms of reducing the model complexity without a significant degradation in quality. This could be advantageous to alleviate the real-time throughput requirements.

IV. Results and discussion

First, we performed an objective evaluation of the trained models to measure the complexity reduction due to proposed baseline modifications, and to quantify the impact of these in the output quality. Subsequently, we carried out a subjective evaluation to investigate how these quality changes are perceived by human subjects, and to compare the perceptual importance of targeting 32 kHz compared to 48 kHz.

A. Objective evaluation

We performed an objective evaluation of all 6 trained models. The results are shown in Table I. Model variants ending in *Shuffle* use a decoder made of *SampleShuffle* blocks. The number appended to *NSVQ* (e.g. *NSVQ6*) indicates the codebook size in bits. For example, a *NSVQ6* model will have a codebook made up of $2^6 = 64$ template vectors. We used the log spectral distance (LSD) [25] as our objective quality metric. We also tried scale invariant signal-to-distortion ratio (SI-SDR) [26], but similarly to [9], we found that this metric is not adequate for bandwidth extension and results in

TABLE I
OBJECTIVE METRICS OF ALL TRAINED MODELS.

Model	LSD	Parameters	Operations per inference[a]	Inference speed[b]
Unprocessed audio 16k	5.05 ± 0.19	-	-	-
Baseline 16k48k	1.18 ± 0.09	692 k	10.6 M	1.21 ± 0.13ms
Baseline Shuffle 16k48k	1.14 ± 0.08	528 k	4.5 M	1.27 ± 0.13ms
BBWEXNet NSVQ6 16k48k	1.19 ± 0.09	471 k	10.4 M	$\mathbf{1.15 \pm 0.12}$ms
BBWEXNet NSVQ8 16k48k	1.35 ± 0.14	496 k	10.5 M	1.17 ± 0.12ms
BBWEXNet NSVQ6 Shuffle 16k48k	$\mathbf{1.04 \pm 0.08}$	**306 k**	**4.3 M**	1.19 ± 0.08ms
BBWEXNet NSVQ8 Shuffle 16k48k	1.12 ± 0.09	331 k	4.4 M	1.20 ± 0.09ms

[a] Calculated over a tensor of dimensions $(1, 1, 256)$ that is the minimum that the network can process due to the encoder downsampling sequence.
[b] Mean $\pm$ standard deviation of 1000 inferences where the first 15 are discarded in a MacBook Pro with 2 GHz Quad-Core Intel i5.

values that neither correlate with other objective or subjective quality evaluations nor show any significant differences between unprocessed audios and different models' predictions. We hypothesize that this occurs because the upsampled input is highly correlated with the ground truth, shadowing perceptually relevant changes in the resulting values.

Additionally, we computed the number of parameters for each model, the estimated number of operations per inference and the inference speed. This way, we can characterize the relationship between quality and complexity, that can be used to determine the most suitable model for a given real-time scenario with specific computational constraints.

It can be observed that all models successfully perform bandwidth extension, showing a significant difference when compared to the unprocessed audio. *BBWEXNet NSVQ6 Shuffle 16k48k* is the best performing model, although there are no substantially large differences in LSD scores among models, suggesting that they have comparable quality. However, the larger differences are in terms of memory footprint (represented by the number of parameters) and operations per inference. In this regard, *BBWEXNet NSVQ6 Shuffle 16k48k* has a 55.78% smaller memory footprint compared to the baseline. Furthermore, the same model can perform an inference using 59.45% less operations than the baseline. The most significant reduction in the number of operations can be attributed to the *SampleShuffle* decoder, removing 6.1 M operations per inference between *BBWEXNet NSVQ6 16k48k* and *BBWEXNet NSVQ6 Shuffle 16k48k* variants.

By replacing the baseline bottleneck by a single NSVQ layer, we reduced the number of parameters by a larger amount than the reduction obtained in the number of operations. This is because each input is compared with the whole codebook, thus, such reduction will scale depending on the codebook size. We found that larger codebook sizes do not always benefit performance even if there are no idle template vectors. Even though they may be deemed as more expressive layers, our results show that such additional capability is not needed.

B. Subjective evaluation

We did a subjective evaluation of a subset of our models to investigate the correlation between human raters ($n = 13$) and objective metrics. We used the webMUSHRA framework that follows the ITU-R recommendation BS.1534 [27]. We included model variants targeting both fullband and superwideband. These are suffixed with *16k48k* and *16k32k*, respectively. We did this to assess the quality degradation in a scenario where decreasing the sample rate can be used to reduce complexity in real-time operation. We restricted our evaluation to 6 models in total to avoid having an overly large stimuli set. The results are shown in Fig. 3. The x-axis represents the number of floating-point operations per second in GFLOPS and the y-axis represents the delta continuous quality score (ΔCQS), calculated as the difference between the CQS of the unprocessed audio and the CQS of each corresponding model as evaluated in a scale from 0 to 100.

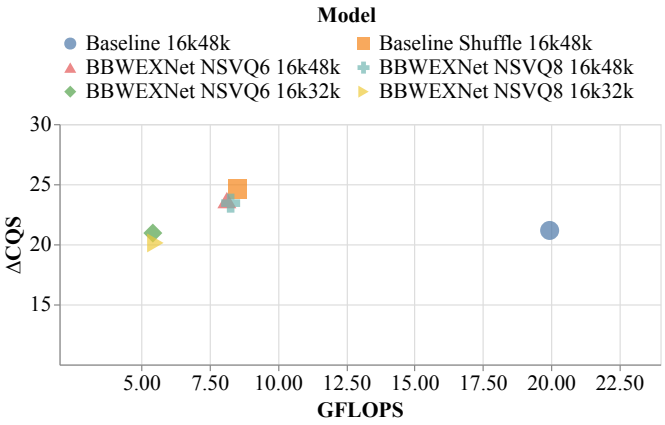

Fig. 3. Complexity versus perceptual quality improvement comparison of a selected number of models. The x-axis corresponds to the floating-point operations per second in GFLOPS and the y-axis is the continuous quality score difference (ΔCQS) between each model and the unprocessed audio.

Our results show that all selected fullband models consistently outperform the baseline. While the improvements in quality are not large (<5 ΔCQS), the computations needed per time unit are approximately halved. Additionally, superwideband models are capable of producing comparable bandwidth extension quality with the baseline, but requiring only around a quarter of its GFLOPS, considerably reducing its complexity.

Human subjects are capable of distinguishing between superwideband and fullband outputs, yet the difference in ΔCQS is marginal. This suggests that the contribution of artificially reconstructed frequencies between 8 kHz and 16 kHz is perceptually more relevant for speech than the reconstruction of frequencies above superwideband's Nyquist frequency.

V. Conclusion

We presented BBWEXNet, a neural network for blind bandwidth extension of speech from wideband to fullband in real-time on CPU. We showed that both noise substitution in vector quantization and sample shuffle techniques can be used to significantly reduce the overall complexity of the baseline model, while producing an output with comparable quality measured by objective and subjective metrics. Our best performing model has 55.78% less parameters than the baseline and needs 59.45% less operations to perform an inference, allowing it to run in a wider variety of consumer devices with possibly more strict computational constraints.

Additionally, we showed that decreasing the output sample rate from fullband to superwideband can be used to further reduce the computational complexity of the model in real-time scenarios with a marginal impact on the output quality as rated by human subjects, elucidating the relative perceptual importance of different frequency bands in the task of speech bandwidth extension.

Acknowledgment

The calculations presented in this publication were carried out using the computer resources of the Aalto University of Science "Science-IT" project.

References

[1] H. Fletcher and J. C. Steinberg. Articulation testing methods. *The Bell System Technical Journal*, 8(4):806–854, 1929.

[2] H. Fletcher and Rogers H. Galt. The perception of speech and its relation to telephony. *The Journal of the Acoustical Society of America*, 22(2):89–151, 1950.

[3] R.P. Lippmann. Accurate consonant perception without mid-frequency speech energy. *IEEE Transactions on Speech and Audio Processing*, 4(1):66–, 1996.

[4] G. W. Hughes and M. Halle. Spectral properties of fricative consonants. *Journal of the Acoustical Society of America*, 28:303–310, 1956.

[5] B. Moore and T. Chin-Tuan. Perceived naturalness of spectrally distorted speech and music. *The Journal of the Acoustical Society of America*, 114:408–19, 08 2003.

[6] H. F. Olson. Frequency range preference for speech and music. *The Journal of the Acoustical Society of America*, 19(4):549–555, 1947.

[7] M. Wang, Z. Wu, S. Kang, X. Wu, H. Jia, D. Su, D. Yu, and H. Meng. Speech super-resolution using parallel WaveNet. In *2018 11th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 260–264, 2018.

[8] J. Su, Y. Wang, A. Finkelstein, and Z. Jin. Bandwidth extension is all you need. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 696–700, 2021.

[9] Y. Li, M. Tagliasacchi, O. Rybakov, V. Ungureanu, and D. Roblek. Real-time speech frequency bandwidth extension. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[10] H. Wang and D. Wang. Towards robust speech super-resolution. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2058–2066, 2021.

[11] H. Liu, W. Choi, X. Liu, Q. Kong, Q. Tian, and D. Wang. Neural vocoder is all you need for speech super-resolution. In *Proceedings INTERSPEECH 2022*, 2022.

[12] C. Yu, S. Yeh, G. Fazekas, and H. Tang. Conditioning and sampling in variational diffusion models for speech super-resolution. *arXiv:2210.15793*, 10 2022.

[13] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, 2015.

[14] M. H. Vali and T. Bäckström. NSVQ: Noise substitution in vector quantization for machine learning. *IEEE Access*, 10:13598–13610, 2022.

[15] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[16] J. O. Smith. *Digital audio resampling home page*. http://www-ccrma.stanford.edu/~jos/resample/, January 28, 2002.

[17] A. Odena, V. Dumoulin, and C. Olah. Deconvolution and checkerboard artifacts. *Distill*, 1(10):e3, 2016.

[18] J. Pons, S. Pascual, G. Cengarle, and J. Serrà. Upsampling artifacts in neural audio synthesis. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3005–3009. IEEE, 2021.

[19] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432, 2013.

[20] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression*, volume 159. Springer Science & Business Media, 2012.

[21] A. van den Oord, O. Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, volume 30, pages 6309–6318, 2017.

[22] D. P. Kingma and M. Welling. Stochastic gradient VB and the variational auto-encoder. In *Second international conference on learning representations, ICLR*, volume 19, page 121, 2014.

[23] S. Dieleman, A. van den Oord, and K. Simonyan. The challenge of realistic music generation: Modelling raw audio at scale. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 8000–8010, 2018.

[24] J. Yamagishi, C. Veaux, and K. MacDonald. CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit (version 0.92). *Edinburgh DataShare*.

[25] A. Gray and J. Markel. Distance measures for speech processing. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(5):380–391, 1976.

[26] J. LeRoux, S. Wisdom, H. Erdogan, and J. R. Hershey. SDR - half-baked or well done? *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[27] M. Schoeffler, S. Bartoschek, F. Stöter, M. Roess, S. Westphal, B. Edler, and J. Herre. webMUSHRA — a comprehensive framework for web-based listening tests. *Journal of Open Research Software*, 6(1), 2018.