NEURAL WAVEFORM CODING: TOWARDS EFFICIENT AND SCALABLE COMPUTATIONAL PARADIGM FOR ACOUSTIC SIGNAL COMPRESSION

Kai Zhen

Dissertation Proposal

for the degree

Doctor of Philosophy

in the Department of Computer Science

and the Cognitive Science Program,

Indiana University

February 19, 2020

Kai Zhen

NEURAL WAVEFORM CODING: TOWARDS EFFICIENT AND SCALABLE COMPUTATIONAL PARADIGM FOR ACOUSTIC SIGNAL COMPRESSION

Acoustic signal compression techniques, converting floating-point waveforms into a bitstream representation, serve a corner stone in the current data storage and telecommunication infrastructure. Conventional digital signal processing (DSP) methodologies stem from human heuristics, which are with limited performance and highly domain specific. For the past decade, deep neural networks (DNNs) have shown the potential to tackle this problem in a pure end-to-end manner, without relying on human priors or feature-engineering but the data itself. Besides, due to this general-purpose computational paradigm, learning a compact representation of acoustic signals can be integrated to various downstream applications such as speech encryption, recognition and natural language understanding towards future multi-modal intelligent systems. However, the rise of DNNs brings in not only potentials but also concerns, among which model complexity is a major challenge especially for acoustic coding systems. Most codecs are deployed on low power devices, such as mobile phones and hearing aides which do not afford a gigantic neural network in spite of the impressive performance.

We propose a research methodology to not simply get rid of conventional DSP methods with the fancy design of advanced neural networks, but revitalize those lightweight yet effective techniques in the modern computational platform. By bridging these two approaches, we merge merits from both sides in terms of performance and complexity. For instance, the performance of end-to-end neural networks mainly depend on millions of parameters and optimization algorithms. This is far from necessary in the domain of speech/audio coding, as the encoding and decoding procedure can be conducted in multiple stages. We could implement this multistage quantization scheme with deep neural network techniques to simplify the model topology and optimization. In addition, speech production process is known to include several components, glottal pulses, noise excitation and the

response of the vocal tract, etc. There is no need to model all components with neural networks, when the response of the vocal tract, for example, can be simply well represented by an all-pole linear filter. By outsourcing sub-tasks to effective conventional DSP methods, the workload of the neural network can also be reduced, accordingly. We are also aware of the discrepancy between human auditory perception and objective loss functions used during model training. In many cases, the model is unessentially complicated to minimize the error not even salient to listeners. By leveraging psychoacoustics, the model for mono-channel audio coding can be with lower complexity yet higher coding efficiency, as its optimization better aligns human cognition.

Towards the remains of the journey, we plan to investigate schemes to incorporate μ-law companding algorithm to our neural codec which currently does not exert non-linear regularization during downsampling and upsampling. Extending this algorithm in the objective function may also be an interesting direction for speech coding where psychoacoustics do not apply. In addition, our current neural quantizer is trainable but scalar based in the lower dimensional space. Inspired from vector quantization-variational autoencoder (VQ-VAE), we are also interested in an index-based quantization that could significantly increase the coding efficiency by modeling the temporal pattern in waveform segments. The neural quantization method can also be employed to Wave-U-Net to make it an alternative modularized neural codec.

Contents

Abstract **ii**

1 INTRODUCTION **1**

1.1 Dissertation Proposal Overview 1

1.2 Related Acoustic Signal Compression Work 4

1.2.1 Conventional coding systems 4

1.2.2 Evaluation metrics 5

1.3 Related Deep Neural Network Techniques 7

1.3.1 Advanced convolutional operations 7

1.3.2 Neural discrete representation learning 11

1.4 Motivation From a Cognitive Science Perspective 15

1.4.1 Discrepancy in objective measurements and subjective evaluations 15

1.4.2 Psychoacoustics in MPEG-1 audio layer I 16

1.5 Summary and Dissertation Outline 20

2 CROSS-MODULE RESIDUAL LEARNING **23**

2.1 Motivation: from Multistaged Quantization to Cascaded Residual Coding 23

2.2 A Simplified Autoencoder For End-to-End Speech Coding 24

2.3 Proposed Cascaded Inter-Model Residual Learning Pipeline 26

2.3.1 The module carrier: CMRL 26

2.3.2 The two-round training scheme 27

2.3.3 Bitrate and entropy coding 28

2.4 Experimental Results 28

2.4.1 Objective test 30

2.4.2 MUSHRA test . . . 31
2.4.3 Model complexity analysis . . . 32
2.5 Summary . . . 32

3 COLLABORATIVE QUANTIZATION: BRIDGING LPC WITH DNN 33

3.1 Preprocessing With Linear Predictive Coding (LPC) . . . 33
3.2 Proposed Quantization Scheme For A Better Pivot . . . 34
3.2.1 Trainable LPC analyzer . . . 35
3.2.2 Residual coding . . . 37
3.3 Experimental Results . . . 37
3.4 Summary . . . 40

4 TOWARDS A PERCEPTUAL LOSS: PSYCHOACOUSTICAL CALIBRATION 41

4.1 Psychoacoustics in Audio Coding . . . 41
4.2 Psychoacoustical Calibration in Neural Audio Codec . . . 42
4.2.1 Priority weighting . . . 44
4.2.2 Noise modulation . . . 44
4.3 Experimental Results . . . 45
4.4 Summary . . . 48

5 FUTURE RESEARCH PLAN 49

5.1 From Wave-U-Net To a Modularized Neural Codec . . . 49
5.2 A μ-Law Conversed Loss Function For Speech Coding and Enhancement . . . 51

6 CONCLUSION 54

Chapter 1

INTRODUCTION

1.1 Dissertation Proposal Overview

Speech coding, where the encoder converts the speech signal into bitstreams and the decoder synthesizes reconstructed signal from received bitstreams, serves an important role for various purposes: to secure a voice communication [1][2], to facilitate data transmission [3], etc. For a speech signal with a sample rate of 16 kHz (16,000 samples per second), if each sample is represented by a 16-bit floating number, the bitrate is 256 kilo bits per second (kbps). Note that the sample rate for audio signals is even higher at 44.1 kHz with more than one channel. Such bitrate levels pose a burden even on modern Internet architecture. With a well-designed speech coding algorithm, the bitrate can be only 10% of the original bitrate or even lower, yet with a decent speech intelligibility. Conventionally, problems as such are addressed by intensive study on human auditory system with quite a handful well-tuned hyperparameters protected by patents, including linear predictive coding (LPC) [4], adaptive encoding [5], and perceptual weighting [6]. Stemmed from those coding techniques, Speex, AMR-WB and Opus are some of the industrialized speech codecs. AMR-WB consists of multiple steps including linear predictive coding (LPC) to estimate spectral envelopes, pre-emphasis and de-emphasis filterings to dim the blocking artifacts, perceptual weighting, the high frequency extension, adaptive and algebraic coding of residuals, etc.

The design of speech codecs is to address the trade-off among low bitrate, high perceptual quality, low complexity and delay, etc [7, 8]. Most conventional codecs are relatively computationally efficient, yet with less satisfying performance especially in low bitrate modes. Most of these speech codecs can be classified into two categorizes, *vocoders* and *waveform* coders [9]. Vocoders use few parameters to model the human speech production process, such as vocal tract, pitch frequency, etc

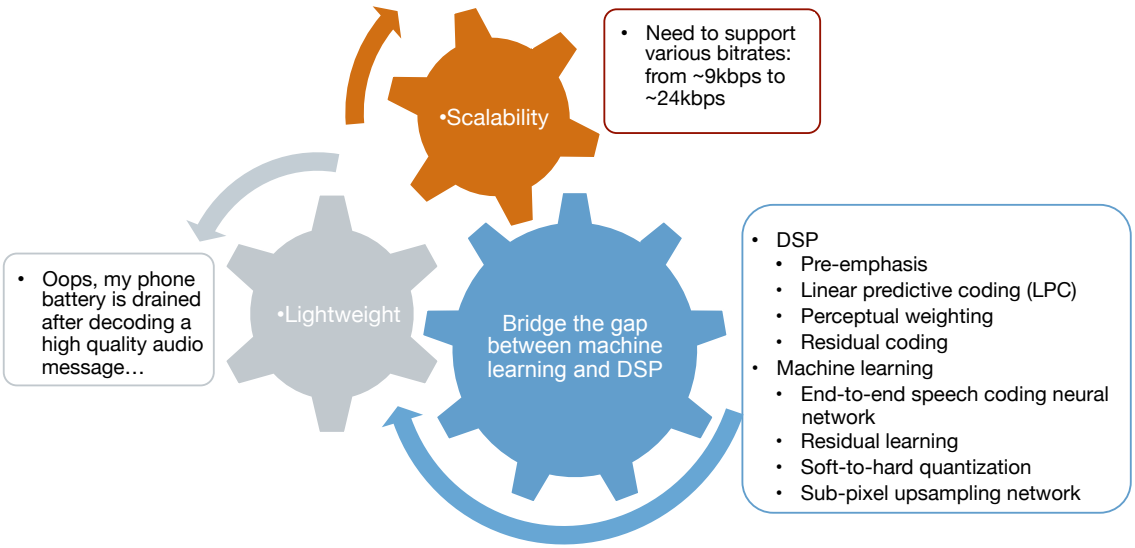


Figure 1.1: Tradeoffs in the design of speech/audio codecs

[10], such that the decoded speech is not exactly "recovered" but rather "synthesized". In comparison, waveform coders compress and reconstruct the waveform to make the decoded speech similar to the input as "perceptually" as possible. Whether to choose a vocoder or a waveform codec is mainly contingent on specific application scenarios. Vocoders are computationally efficient and can encode speech at very low bitrates with limited performance, while waveform coders support a much wider bitrate range with scalable performance and are more robust to noise.

The field has showed great enthusiasm towards deep neural network based methods as they feature the potential of cracking those conventional, domain-specific and less-open algorithms with this modern computational paradigm relying mostly on data, rather than less accessible heuristics. A speech coding system can be formulated by DNN as an autoencoder (AE) with a code layer discretized by vector quantization (VQ) [11] or bitwise network techniques [12], etc. Many DNN methods [13][14] take inputs in time-frequency (T-F) domain from short time Fourier transform (STFT) or modified discrete cosine transform (MDCT), etc. Recent DNN-based codecs [15][16][17][18] model speech signals in time domain directly without T-F transformation. They are referred to as end-to-end methods, yielding competitive performance comparing with current speech coding standards, such as AMR-WB [19].

However, people are less credulous than they used to be when there is a newly proposed neural network which is claimed to outperform the previous state-of-the-art, especially for speech/audio coding. The reason is that these acoustic codecs are usually deployed on low power devices with limited storage and energy supply, while many of these DNNs achieve competitive performance at the cost of model complexity. For example, a WaveNet based variational autoencoder (VAE) [18] outperforms other low bitrate codecs in the subjective listening test, however, with 20 millions parameters, which is a too big model for real-time processing in a resource-constrained device.

As an effort to bring deep neural networks closer to low power devices, we study ways to incorporate conventional digital signal processing (DSP) methods to DNNs. The reason behind this is simple: DSP methods are efficient and task specific, while DNNs are more general, powerful and expensive to operate. Using DNN as a platform where DSP methods are well placed to unload a certain amount of computational overhead can effectively reduce the model complexity with satisfying performance. The meaning of the study on efficient and scalable neural waveform codec is two fold: first, it helps to find a better pivot in the performance-complexity tradeoff, such that future neural codecs may have the potential to be employed to industrial products. Second, it could be seamlessly incorporated with other artificial intelligence tasks that have already heavily relied on deep neural networks. It is well known that future intelligent systems require not only on a single modality algorithm even with the surpassing human-level performance, either in objective recognition or speech recognition, but a multi-modal interactively learning system. Should the speech/audio codec be implemented purely based on DSP, it is not possible to be collaboratively trained along with other neural network components. Towards this big picture, making neural waveform codecs friendly to low-power device is the step one.

With this statement of purpose, we proposed a cascaded cross-module residual learning (CMRL) pipeline by enabling the conventional multi-staged residual coding concept in deep neural network platform for speech and audio coding. In addition, we combined linear predictive coding (LPC)

with CMRL and designed a collaborative quantization (CQ) scheme where LPC codebook is jointly learned with the corresponding residual quantization to achieve transparent speech coding with much lower model complexity. Note that psychoacoustics used in conventional audio coding can still be leveraged to inform a prioritized neural network optimization procedure. To that end, we plan to extend our previous preliminary study on psychoacoustically weighted cost function to audio coding, where the network is navigated to cut things fine by only remove the audible artifacts which have negative impact on subjective listening test.

1.2 Related Acoustic Signal Compression Work

1.2.1 Conventional coding systems

Data compression has been well investigated for decades with a rich literature. Even for audio coding specifically, it is hard to summarize various techniques comprising a wide range of audio coding formats, perceptual quality levels, bitrates, bandwidths, etc.

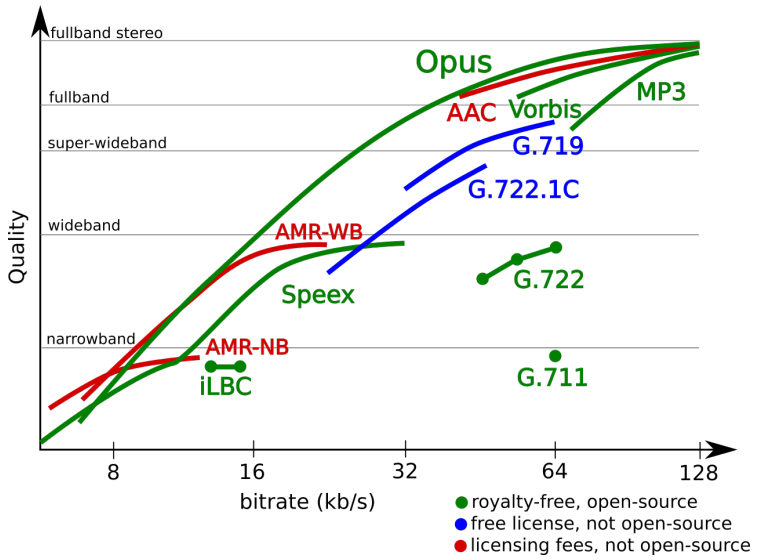


Figure 1.2: The codec comparison on performance, bitrate, and bandwidth [20]

We introduce three speech/audio codecs which have been compared to our proposed systems, AMR-WB [19], Opus [21], and MPEG Audio Layer III (MP3) [22]. Adaptive Multi-Rate Wide-

band, abbreviated as AMR-WB, is a patented speech coding standard developed by Nokia and VoiceAge. It improves the speech quality from its predecessor AMR-NB, due to a wide speech coding bandwidth of 50-7000 Hz. AMR-WB provides a scalable performance in multiple bitrate configurations from 6.6 kbps to 23.85 kbps. MP3 is a set of lossy audio coding standards which have been deployed in multiple industrial products. It can compress the CD quality stereo music 10-12X (from 1.4 Mbps to 128 kbps) without perceptual loss of quality. Psychoacoustic models serve a critical role in the design of MP3. Opus is more recently developed and covers a much wider bitrate range from 6 kbps to 510 kbps for stereo audio sources. Opus can be used to compress both speech and audio signals. As shown in Fig. 1.2, the speech quality from Opus at very low bitrate modes is less competitive.

1.2.2 Evaluation metrics

Objective measures

In acoustic signal processing, objective measures are highly task specific. For source separation, BSS_Eval toolbox decomposes an overall source-to-distortion ratio (SDR) to components corresponding to different error types: source-to-interference ratio (SIR), source image-to-spatial distortion ratio (ISR), and source-to- artifacts ratio (SAR). STOI measures objective intelligibility for enhanced signal, which is positively correlated with the performance of automatic speech recognition (ASR) systems. Speech coding differs from source separation, or speech enhancement, in that there is no additive interference, and the degradation is caused by quantization error and artifacts from the codec. To assess the speech quality from a codec, we typically rely on PESQ (Perceptual Evaluation of Speech Quality). PESQ, standardized as ITU-T recommendation P.862 (02/01), models mean-opinion-score (MOS) with the range of 1 (bad) to 5 (excellent). P.862.2 extends PESQ to support the evaluation of wideband telephone networks and speech codecs up to a sample rate of 16 kHz.

Mean Opinion Score (MOS) From Subjective Listening Tests

Note that the evaluation which is only supported by objective measures may not be sufficiently convincing as the gap between these objective scores and mean opinion score (a straightforward reflection of human auditory perception) is a known fact. Therefore, it is highly recommended to report MOS acquired from subjective listening test. There are two major caveats when conducting a test as such: first, it is relatively time consuming to collect MOS; second, the effectiveness of results is less statistically significant if the amount of subjects is not large enough. With that perspective, industrial research institutes, such as Google, have their own on-site subjective listening test platform to invite employees to input their opinions; other online crowdsourcing platforms, such as Amazon Mechanical Turk, can also be used to conduct subjective listening test by enrolling volunteers. Even with this effort, it may still be a questionable process as a subjective listening test: there is no control on the capability and level of commitment of human listeners. Comparing and evaluating different coding systems with very subtle differences can be tedious. As a consequence, subjective listening tests are expected to be conducted by audio experts.

In our research, we collect MOS based on MUltiple Stimuli with Hidden Reference and Anchor (MUSHRA) [23] standards. Each MUSHRA test may have several sessions, one for each bitrate mode. Each session usually includes multiple trials. As indicated by its name, each trial includes a reference signal, one or two low-pass anchor signals, several enhanced signals from different coding systems, and a hidden reference, and subjects are then asked to score each of them based on their perceptual preference. Fig.1.3 shows the interface of a MUSHRA trial. Among eight competing systems, there are

- **two anchors** that are processed by low-pass filters at 3.5 kHz and 7 kHz, respectively. The listener is expected to give relatively low scores for anchors;
- **one hidden reference** which is equivalent to the explicit reference signal on the left, and the listener should detect it and grade it 100;

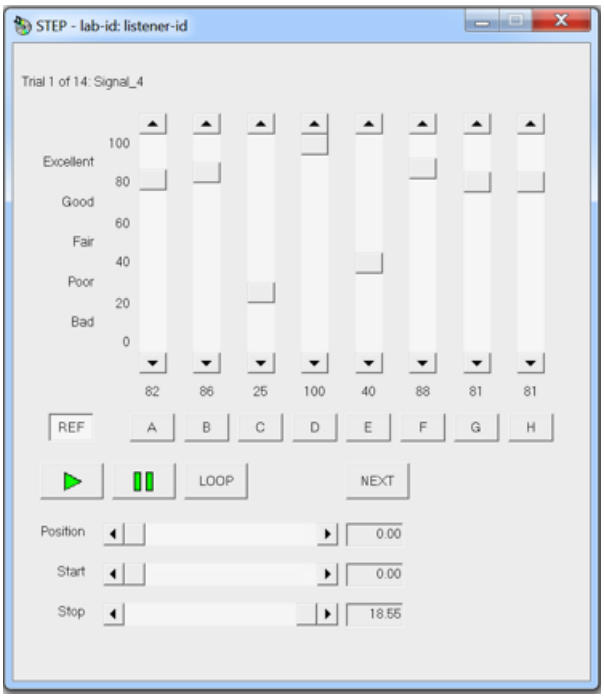


Figure 1.3: The interface for a MUSHRA trial

- other **decoded signals** with various degradation levels.

The scores for all trials will be accumulated to calculate the MOS. Typically, MOS can be descriptively represented in a box plot.

Aside from MUSHRA, A/B test is used in ablation analysis where only two competing systems are involved along with the reference. In A/B test, the listener is asked to select what is more similar to the reference out of the two decoded signals.

1.3 Related Deep Neural Network Techniques

This section introduces several advanced neural network techniques from non-linear transformation to trainable quantization, which are building blocks to our proposed systems.

1.3.1 Advanced convolutional operations

Unlike how CNN is mostly used in computer vision, for end-to-end acoustic signal processing where the input is waveform segment in time domain, it suffices to use 1-dimension (1-D) convolution.

Given the convolution operator denoted as $*$, the 1-D convolution evaluated at p on the signal $\mathcal{I}$ with the kernel $\mathcal{K}$ is formally defined in Eq.1.1, where γ denotes the dilation rate with the default value of 1. CNNs employ dilated layers to enlarge the receptive field, aggregating contextual information without increasing the kernel size [24].

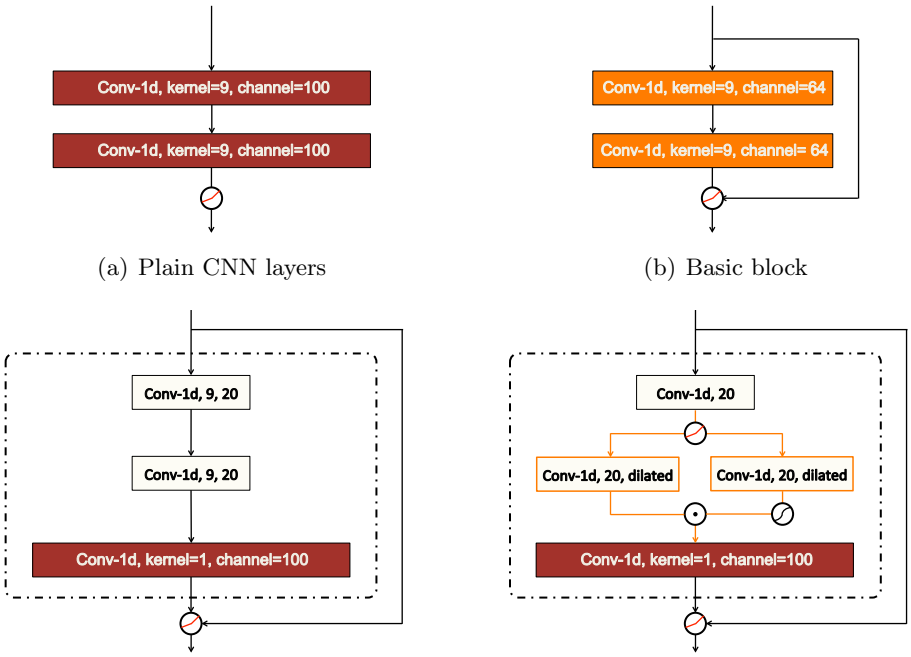
$$(\mathcal{I} *_\gamma \mathcal{K})(p) = \sum_{s+\gamma t=p} \mathcal{I}(s)\mathcal{K}(t)) \tag{1.1}$$

Residual learning blocks

Residual learning is arguably one of the most well known techniques to reduce model complexity and facilitate the optimization of very deep DNN models. As more layers are added, the model capacity increases. However, it poses a challenge to training a gigantic network, especially due to the gradient vanishing issue [25]. The core idea of residual learning is to add identical shortcuts (Fig.3.5 (b)), such that the gradient will be back-propagated through an identity function, or better preserved.

It is critical to make residual learning building blocks efficient, as they are repeatedly used in various advanced CNN architectures. Bottleneck residual learning blocks [26] usually replace $\left[\begin{smallmatrix} 9,\ 64 \\ 9,\ 64 \end{smallmatrix}\right]$ type of kernel setting to $\begin{bmatrix} 1,\ 20 \\ 9,\ 20 \\ 1,\ 64 \end{bmatrix}$. Note that not only the amount of parameters for each block is reduced, the dimension of the feature map is increased which further benefits the overall performance.

Originated from bottleneck residual blocks, gated linear unit implements a novel gating mechanism in convolutional residual learning blocks, since many tasks in acoustic signal processing requires long temporal dependency. In fact, a LSTM-style gating scheme was firstly proposed in [27], which is claimed to benefit more complex interactions by controlling the information flow in CNNs. The scheme is defined in Eq.1.2, where $\mathbf{v_1} = x * \mathbf{W_1} + \mathbf{b_1}$ and $\mathbf{v_2} = x * \mathbf{W_2} + \mathbf{b_2}$, with $\mathbf{W}$ and $\mathbf{b}$ being the kernel and bias. However, as shown in its gradient (Eq.1.3), this LSTM-style gating is subject to the gradient vanishing issue due to downscaling factors in its gradient. Gated



(a) Plain CNN layers (b) Basic block

(c) Bottleneck block (d) Gated linear unit

Figure 1.4: CNN building blocks

linear unit [28] defines the gating scheme in Eq.1.4, to address the gradient vanishing problem in conventional gated convolutional layers (Figure.1.4(d) (a)), as the downscaling factor from the first term is removed from the gradient (Eq.1.5). In addition, the gated bottleneck layers are usually dilated to better aggregate contextual information in the time domain, as proposed in [24].

$$y = tanh(\mathbf{v_1}) \odot \sigma(\mathbf{v_2}) \tag{1.2}$$

$$\nabla[tanh(\mathbf{v_1}) \odot \sigma(\mathbf{v_2})] = tanh'(\mathbf{v_1})\nabla\mathbf{v_1} \odot \sigma(\mathbf{v_2}) + \sigma'(\mathbf{v_2})\nabla\mathbf{v_2} \odot tanh(\mathbf{v_1}) \tag{1.3}$$

$$y = \mathbf{v_1} \odot \sigma(\mathbf{v_2}) \tag{1.4}$$

$$\nabla[\mathbf{v_1} \odot \sigma(\mathbf{v_2})] = \nabla\mathbf{v_1} \odot \sigma(\mathbf{v_2}) + \sigma'(\mathbf{v_2})\nabla\mathbf{v_2} \odot \mathbf{v_1} \tag{1.5}$$

Sub-pixel upsampling convolution

Upsamling convolution is essential for our neural codecs as it recovers the feature map, after down-sampling with strided convolutions, to the original shape. Sub-pixel upsampling is proposed by [29]

for super resolution images. Unlike the plain deconvolutional layer, Sub-pixel upsampling involves a novel permutation operation which aggregates feature maps from previous convolutional layers to build a super resolution (SR) image. Suppose the waveform segment contains 12 samples (shaped as $< 1, 12, 1 >$). After downsamling by 4X through convolutional layers, the feature map is with the shape of $< 1, 3, 8 >$ with 8 being the number of output channels and 3 being the length of each 1-D feature. Sub-pixel upsamling is conducted by firstly reshape the feature map to the one with the shape of $< 1, 3, 2, 4 >$ where 4 is the desired upsampling factor; then the last two dimensions are permuted to yield a feature map shaped as $< 1, 3, 4, 2 >$ which is then transformed to the shape of $< 1, 12, 2 >$. With another channel-change convolutional layer, we can acquire a feature map with the original shape of $< 1, 12, 1 >$. This process is instantiated in Fig. 1.5. Note that the feature map is learnable during backpropagation, although these feature map transformations are deterministic.

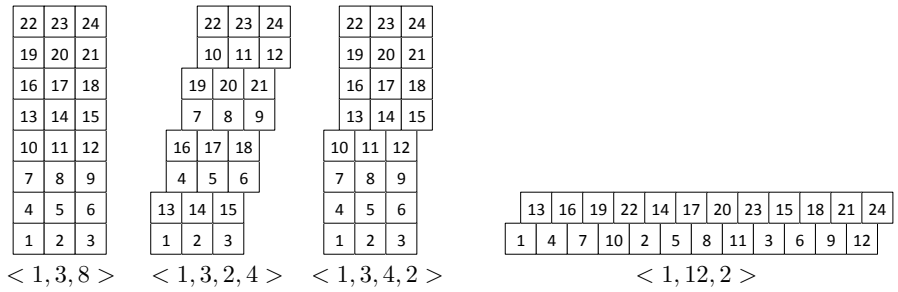


Figure 1.5: An example of 4X sub-pixel upsampling

Depthwise separable convolution

The runtime efficiency is highly related to the amount of feature transformations which can be calculated by the amount of parameters in convolutional kernels. Consider the transformation for the last feature map in Fig. 1.5 without changing its shape. If the kernel size is 3, we need $3 \times 2 \times 2 = 12$ parameters from a normal convolution. However, if we decompose this convolution into two steps, by conducting time domain convolution and depthwise convolution separately, we only need $3 \times 1 \times 2 + 1 \times 1 \times 2 = 8$ parameters. This technique is referred to as "depthwise separable

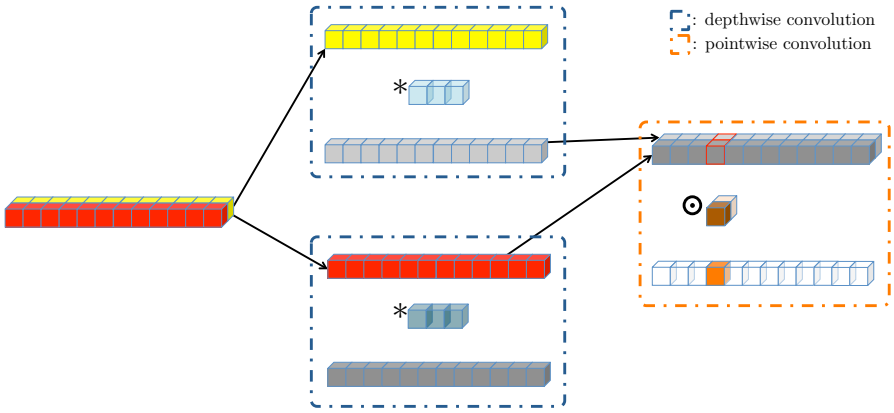


Figure 1.6: An example of 4X upsampling

convolution". The toy example is described in Fig. 1.6. Basically, a normal 1-D convolution that transforms a feature with the shape of $< l, d_1 >$ to $< l, d_2 >$ requires a kernel with $k_s \times d_1 \times d2$ parameters; for depthwise separable convolution, the number can be lowered to $k_s \times 1 \times d2 + 1 \times 1 \times d2$. Note that in a more realistic case, this technique brings in noticeable efficiency gain.

1.3.2 Neural discrete representation learning

Bitwise neural network

Bitwise neural network, or BNN, quantizes the input, parameters and activation functions in a fully-connected network, such that the binarized model can be implemented by the XNOR gate. BNN benefits the memory usage as well: for a floating-point network that takes 32 MB space, the corresponding bitwise version only ask for 0.5 MB of memory space. Empirical results show that BNN performs well for speech denoising and handwritten digit recognition at minimal computational cost.

While BNN is proposed as a network compression technique, its binarized activation function can be utilized to generate bitstrings similar to a vector quantization procedure. For instance, the output of the code layer can be bounded between -1 to $+1$, with a *tanh* function as in $\mathbf{y}_{\text{code}} = tanh(\mathbf{W}\mathbf{x}_{\text{code}} + b)$. During feedforward, the *tanh* function is replaced by *sign* function, such that the output can only be either -1 or $+1$ as in $\mathbf{y}_{\text{code}} = sign(\mathbf{W}\mathbf{x}_{\text{code}} + b)$. Note that *sign*

function is non-differentiable. To make it compatible with backpropagation during model training, the derivative of *sign* function is approximated by that of a *tanh* function (Eq.1.7).

$$\nabla[tanh(\mathbf{W}\mathbf{x}+b)] = 1 - tanh^2(\mathbf{W}\mathbf{x}+b) \tag{1.6}$$

$$\approx \nabla[sign(\mathbf{W}\mathbf{x}+b)] \tag{1.7}$$

The downside of BNN as a discrete representation learning technique lies in the difficulty in coupling with entropy coding. In other words, even though a BNN converts waveform segments to binary vectors, it is hard to exert a certain form of regularization to tune the frequency distribution of these vectors. The amount of unique binary vectors increases exponentially, as the binary vector with the length of l has 2^l permutations: if these bitstring symbols are almost uniformly distributed, entropy coding may not benefit the coding efficiency.

Vector quantized-variational autoencoder

As indicated by its name, VQ-VAE has two major components: first, it's a variational autoencoder (VAE), whose encoder outputs a posterior distribution, not a discriminate code, and decoder takes the sample from the distribution. Unlike the conventional autoencoder which targets input reconstruction, VAE is capable of synthesizing the input, or generating similar but new samples that have never existed before; the other component is the built-in vector quantization. The encoder outputs a binary string which is the index of the codebook. Once the binary string is transmitted to the receiver side, the decoder fetches the corresponding vector from the codebook with the index to proceed to the feedforward.

VQ-VAE has become a popular neural network model in speech coding as it can downsample the feature map by 64X and deliver decent performance at very low bitrates below 2 kbps. As with most vector quantization schemes, the downsampling capacity of VQ-VAE lies in the fact that the encoder does not compress the input to the code but the index that maps to the code, while the

decoder takes the index to query the code and conduct the feedforward. By maintaining a Hash table (the codebook), not only can the code efficiency be improved, but the temporal structure of the input waveform can also be leveraged. Due to its "variational" nature, the "synthesized" speech does not preserve the prosody of the speaker. Stacked by a WaveNet decoder which considers pitch (f_0) information, [18] achieves prosody-transparency at 1.6 kbps.

Unlike speech synthesis, for acoustic waveform compression, we aim at reconstructing the input with high fidelity. Particularly, for audio coding, without exerting constraints on the posterior distribution of the encoder, the synthesized output may not be preferred in subjective similarity test. However, the proposed trainable vector quantization module which shows impressive coding efficiency can be leveraged.

Another drawback of VQ-VAE is the difficulty in updating the codebook. By definition, the vector quantization block is non-differentiable, due to the *argmin* operator. The author simply copies the gradient from the decoder to the encoder over the block, such that the error can be passed through during model training. We argue that the codebook can be better optimized if Soft-to-hard technique can be integrated to enable a full trainable pipeline.

Soft-to-hard quantization

To compress speech signals, a core component of this AE is the trainable quantizer which learns a discrete representation of the code layer in the AE. Out of the recent neural network-compatible quantization schemes, such as VQ-VAE [30] and soft-to-hard quantization [31], we focus on soft-to-hard quantization, namely *softmax* quantization as in the other end-to-end speech coding AEs [32, 33]. Given an input frame $\boldsymbol{x} \in \mathbb{R}^S$ of S samples, the output from the encoder is $\boldsymbol{h} = \mathcal{F}_{\text{Enc}}(\boldsymbol{x})$, each is a 16-bit floating-point value. Given $J = 32$ centroids represented as a vector $\boldsymbol{b} \in \mathbb{R}^J$, softmax quantization maps each sample in $\boldsymbol{h}$ to one of J centroids, such that each quantized sample can be represented by $\log_2 J$ bits (5 bits when $J = 32$).

This quantization process uses a hard assignment matrix $\boldsymbol{A}_{\text{hard}} \in \mathbb{R}^{I \times J}$, where I and J are the dimension of the code and the vector of centroids, respectively. It can be calculated based on the element-wise Euclidean distance matrix $\boldsymbol{D} \in \mathbb{R}^{I \times J}$.

$$\boldsymbol{A}_{\text{hard}}(i,j) = \begin{cases} 1 & \text{if } \boldsymbol{D}(i,j) = \min_{j'} \boldsymbol{D}(i,j') \\ 0 & \text{otherwise} \end{cases} . \tag{1.8}$$

Then, the quantization can be done by assigning the closest centroid to each of $\boldsymbol{h}$'s elements: $\bar{\boldsymbol{h}} = \boldsymbol{A}_{\text{hard}}\boldsymbol{b}$. However, this process is not differentiable and blocks the backpropagation error flow during training. Instead, soft assignment is used during training as follows:

(a) Calculate the distance matrix $\boldsymbol{D} \in \mathbb{R}^{I \times J}$ between the elements of $\boldsymbol{h}$ and $\boldsymbol{b}$.

(b) Calculate the soft-assignment matrix from the dissimilarity matrix using the softmax function $\boldsymbol{A}_{\text{soft}} = \text{softmax}(-\alpha \boldsymbol{D})$, where the softmax function applies to each row of $\boldsymbol{A}_{\text{soft}}$ to turn it into a probability vector, e.g., $\boldsymbol{A}_{\text{soft}}(i,j)$ holds the highest probability iff $\boldsymbol{h}_i$ is most similar to $\boldsymbol{b}_j$. Therefore, during the training phase $\boldsymbol{A}_{\text{soft}}\boldsymbol{b}$ approximates hard assignments and is fed to the decoder as the input code, while still differentiable. The additional variable α controls the softness of the softmax function, i.e., $\lim_{\alpha \to \infty} \boldsymbol{A}_{\text{soft}} = \boldsymbol{A}_{\text{hard}}$. We use $\alpha = 300$ to minimize the gap between $\boldsymbol{A}_{\text{soft}}$ and $\boldsymbol{A}_{\text{hard}}$.

(c) At testing time, $\boldsymbol{A}_{\text{hard}}$ replaces $\boldsymbol{A}_{\text{soft}}$ by turning the largest probability in a row into one and zeroing the others. $\boldsymbol{A}_{\text{hard}}\boldsymbol{b}$ creates the quantized code $\bar{\boldsymbol{h}}$.

Fig. 1.7 summarizes the softmax quantization process.

Note that there are two major constraints that limit the systematic performance.

(a) The quantization is conducted per sample (scalar quantization). Basically, the quantizer maps each sample from neural encoder to the closest kernel. This procedure is not optimal as it does not utilize the inherit temporal structure in waveforms, which could be modeled in

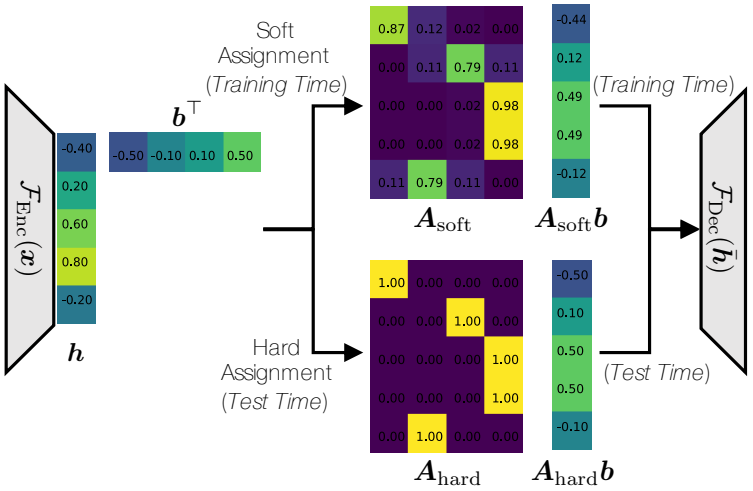


Figure 1.7: An example of the softmax quantization process.

vector quantization. However, empirically, we could not implement a vector based soft-to-hard quantization.

(b) Unlike the encoder in VQ-VAE with 64X dimensionality reduction, the downsampling factor can not be greater than 4X, beyond which the system will collapse.

1.4 Motivation From a Cognitive Science Perspective

Many recently developed deep learning techniques for audio signal processing are found conceptually related to human auditory perceptual systems. For instance, the term receptive field in convolutional neural networks echoes the concept reminiscent of that in actual biological auditory systems. Our research focuses on achieving model efficiency by leveraging the domain knowledge of human auditory perception. Particularly, we are interested in better modeling our perception of sound in the objective function of a neural network, such that the model is trained to meet up the expectation of human ears, not less relevant benchmarks.

1.4.1 Discrepancy in objective measurements and subjective evaluations

Whether it is supervised learning or unsupervised learning, neural networks are trained to lower the loss defined in an objective function. In computer vision, neural networks have been optimized to

achieve the surpassing human level accuracy for object recognition; in automatic speech recognition, the model is trained to lower the word error rate. For these problems, the lower the loss is, the better the performance or model is. Therefore, efforts have been made to propose advanced neural network models for scaled datasets to further lower the loss, or improve the performance.

Neural speech/audio coding differs from many of the tasks mentioned above, in that the model output, the decoded signal, is eventually evaluated not by an objective measure, but human ears. Hence, a very natural question is raised that if the objective function is ill-defined, will the output with a satisfying objective score be correspondingly appreciated by human subjects. The answer to this question is case by case: if the objective score is very high, say over 60 dB in terms of signal-to-noise ratio (SNR), then subjects will also be likely to think the decoded speech is of transparent quality; if the objective score is extremely low, human will not like it neither. The real uncertainty trick likes in somewhere in the middle: if the SNR is 15 dB, it is hard to predict the perceptual quality level of the model output.

Since our goal is to not trade model complexity for good performance, a gold objective function that truly reflects human auditory perception is highly desired. The hope is that the model size can be reduced with the remained capacity fully being committed to lowering the loss that impacts subjective listening experience. The perception of sound has been studied in the field of psychoacoustics, a branch of psychophysics. In fact, the usage of psychophysics is a key factor of the commercial success of many conventional audio codecs.

1.4.2 Psychoacoustics in MPEG-1 audio layer I

This section illustrates how psychoacoustic principles mentioned above are applied to the MPEG-I Audio Layer I, commonly referred to as MP1, for audio compression. Although MP1 is a simplified version of MP2 and more popular MP3, all these three encoding standards utilize psychoacoustics to determine the maximum allowable quantization error in each critical band so that the noise

remains inaudible. The following steps explain how MP1 estimates individual masking thresholds per frame and then accumulates them into global masking threshold.

A general procedure

Spectral analysis and SPL normalization is to first acquire digital representations in the frequency domain, with spectral components expressed in sound pressure level (SPL). Given a clean signal, we first conduct spectral analysis with STFT. For a standard CD quality pop music sampled at 44.1 kHz/16 bits per sample, and the power spectral density (PSD) estimation is conducted via a 512-point FFT with a $1/16th$-overlapped Hann window. The PSD estimation from FFT is then normalized to SPL in Eq. 1.9, where the power normalization term PN is fixed at 90.302 dB.

$$P(k) = PN + 10\log_{10}|STFT(k)|^2 \tag{1.9}$$

Tonal and noise maskers identification: once the the spectral features are normalized, tones, the peaky components, are identified by finding local maximum values. Concretely, tones are defined as local maxima exceeding their neighbours by at least 7 dB, see Eq. 1.10.

$$S_T = \left\{ P(k) \middle| \begin{array}{l} P(k) > P(k \pm 1), \\ \\ P(k) > P(k \pm \Delta_k) + 7dB, \end{array} \right\} \tag{1.10}$$

where

$$\Delta_k \in \left\{ \begin{array}{llll} 2 & 2 < k < 352 & (0.03 - 5.5kHz) \\ \\ [2,3] & 352 < k < 512 & (5.5 - 8kHz). \end{array} \right\} \tag{1.11}$$

The tonal masker is then calculated by combining energy from three adjacent spectral components at the peak as in Eq. 1.12.

$$P_{TM}(k) = 10\log_{10}\sum_{j=-1}^{1} 10^{0.1P(k+j)}(dB) \tag{1.12}$$

By default, the residual spectral energy within a critical bandwidth that is not associated with a tonal masker must be associated with a noise masker. Hence, a noise masker, in each critical band, combines energy of all spectral components that do not contribute to the tonal masker. Specifically, the noise masker is calculated in Eq. 1.13, where $\bar{k}$ is the geometric mean of the critical band, $(\prod_{j=l}^{u} j)^{1/(l-u+1)}$.

$$P_{NM}(\bar{k}) = 10\log_{10}\sum_{j} 10^{0.1P(j)}(dB), \forall P(j) \notin P_{TM}(k, k\pm 1, k\pm\Delta_k) \tag{1.13}$$

Masker decimation reduces the amount of maskers using two criteria. First, we only preserve maskers that are above the absolute threshold of hearing, $P_{TM,NM} \geq T_q(k)$, where $T_q(k)$ is the absolute threshold at spectral line k. All other maskers will be dropped. Next, due to the distinct frequency selectivity in various critical bands, a sliding window with 0.5 Bark bandwidth is used to replace a pair of maskers within the window by the stronger one. That is, if more than two stimulus are present in frequency by less than 0.5 Bark, only the strongest one will be selected.

Individual masking thresholds are calculated based on a set of tonal and noise maskers in Step. 1.4.2. The individual masking threshold of a tonal masker is specified in Eq. 1.14

$$T_{TM}(i,j) = P_{TM}(j) - 0.275z(j) + SF(i,j) - 6.025(dB - SPL), \tag{1.14}$$

where P_{TM} denotes SPL of the tonal masker at frequency bin j, $z(j)$ is the Bark frequency of j-th bin, and $SF(i,j)$ models the spread of masking from i-th bin to j-th bin, Eq. 1.15.

$$SF(i,j) = \begin{cases} 17\Delta_z - 0.4P_{TM}(j) + 11, -3 \leq \Delta_z < -1 \\ (0.4P_{TM}(j) + 6)\Delta_z, -1 \leq \Delta_z < 0 \\ -17\Delta_z, 0 \leq \Delta_z < 1 \\ (0.15P_{TM}(j) - 17)\Delta_z - 0.15P_{TM}(j), 1 \leq \Delta_z < 8. \end{cases} \tag{1.15}$$

The piece-wise function (1.15) echoes the fact that the frequency selectivity of human ears decreases towards higher frequencies. For simplicity, the spread of masking is constrained to 10-Bark neighborhood.

Likewise, the individual masking threshold for a noise masker is given in Eq. 1.16

$$T_{NM}(i,j) = P_{NM}(j) - 0.175z(j) + SF(i,j) - 2.025(dB - SPL), \tag{1.16}$$

where P_{TM} in (1.14) and (1.15) is replaced by P_{NM}.

Global masking threshold combines individual masking curves from the tonal and noise maskers, along with the absolute threshold of hearing in Eq. 1.17

$$T_g(i) = 10\log_{10}(10^{0.1T_q(i)} + \sum_{l=1}^{L} 10^{0.1T_{TM}(i,l)} + \sum_{m=1}^{M} 10^{0.1T_{NM}(i,m)}), \tag{1.17}$$

where $T_q(i)$ is the absolute hearing threshold at i-th frequency bin, $T_{TM}(i,l)$ and $T_{NM}(i,m)$ are individual masking thresholds of l-th tonal masker and m-th noise masker at i-th frequency bin, respectively.

Bit allocation algorithm

The bit allocation algorithm leverages the global masking threshold calculated from the psychoacoustic model, to determine how many bits to be allocated for each sub-band. This iterative algorithm contains following steps:

(a) calculate MNR (mask-to-noise ratio) for each sub-band;

(b) rank sub-bands in the order of lowest to highest MNR;

(c) allocate bits to sub-bands: the one with the lowest MNR receives the least amount of bits;

(d) iterative (a)-(c) steps, until no more bits can be allocated.

1.5 Summary and Dissertation Outline

In this chapter, I briefly outlined the motivation of the research in terms of the potential performance gain over conventional methods, and its application to other artificial intelligence related field from an even bigger perspective. The challenge of simply employing deep neural networks to this problem in an end-to-end manner was discussed: to make the future neural speech/audio codecs computationally affordable on intelligent device, we must reduce the model complexity especially during the feedforward. There could be multiple directions towards efficient neural codecs, such as proposing novice neural networks, and enlarge the size of the training data, etc. We believe the capacity and efficiency of neural codecs not only lies in the power of computation itself, but an elegant design to integrate conventional yet effective digital signal processing methods to this new paradigm. This is by no means a given solution, and many questions remained to be clarified: what specific sub-tasks in speech coding can be more effectively solved by DSP methods? How to implement those methods with deep learning techniques such that the whole system is still differentiable?

The rest part of the proposal explains several primitive work aligned with our research goal.

- **Chapter 2** talks about cascaded cross-module residual learning, a neural network based coding pipeline that implements a conventional concept of multistage vector quantization (MSVQ). The idea is to not attempt to quantize the speech signal with one-shot. Instead, encode the signal through multiple phases. This allows us to not rely on a powerful and gigantic neural codec, but a set of serialized lightweight neural codecs. We illustrate, in details, a compact design of this lightweight neural codec, and how many of these codecs can be hosted and optimized to deliver transparent decoded speech signals.

- In **chapter 3**, we proposes an algorithm similar to LPC-Net but for waveform quantization. Instead of modeling the whole speech generation process in neural network, we outsource

the vocal tract response part to linear predictive coding, a very effective DSP technique of low computational overhead. In addition, we manage to integrate the DSP workflow to the TensorFlow graph, such that the quantization for LPC coefficients and the residuals can be collaboratively optimized. Empirical results show that the proposed collaborative quantization (CQ) scheme find a better pivot between LPC module and neural network residual quantization module, such that the performance is more robust even at lower bitrates.

- The effort of proposing efficient neural codecs should be made not only on new model architectures, but more perceptually salient objective functions as well. In **chapter 4**, we introduce an optimization scheme which is better correlated to human auditory perception, by leveraging psychoacoustics. By calculating the global masking threshold for each input audio waveform segment, the model is trained to only remove audible artifacts generated during compression while being tolerant to other errors. Therefore, there is no need to send as many bits per second or use as large neural network to deliver comparable performance. To that end, we have proposed two loss terms to prioritize the training and modulate the artifact, respectively. This is, to our best knowledge, the first work to bring psychoacoustics to neural audio coding.

We are aware of the fact that many of these primitive results can be further investigated from multiple aspects. To make most of the rest journey, I will tentatively outline two concrete research plans in **chapter 5** as natural extensions to our previous work.

- Inspired from our work on psychoacoustical calibration for audio coding, we wonder if there exists a better loss function for speech coding as well. Note that it is not feasible to directly shoehorn the loss function for audio coding to speech coding, due to the bandwidth difference between these two types of acoustic signals and the fact that there is no evidence that the masking curve from a speech signal is as useful. Empirically, we find that the error generated from end-to-end neural networks should not be weighted uniformly: those from samples

with low amplitudes should be weighed more. This can be explained by μ-law companding algorithm, a well-known digitization technique.

- Aside from the proposed CMRL pipeline, we find that Wave-U-Net can also serve a modulized model for acoustic waveform compression. However, the identity shortcuts bridging the encoder to the decoder disable it from being a codec. We plan to replace these shortcuts with neural quantizers. Instead of copying and pasting the residual, we approximate it from a trainable quantization scheme. To better leverage the temporal pattern in waveform segments, we plan to extend the current sample based neural quantizer to the vector based counterpart.

Chapter 2

CROSS-MODULE RESIDUAL LEARNING

2.1 Motivation: from Multistaged Quantization to Cascaded Residual Coding

Since the last decade, data-driven approaches have vitalized the use of deep neural networks (DNN) for speech coding. A speech coding system can be formulated by DNN as an autoencoder (AE) with a code layer discretized by vector quantization (VQ) [11] or bitwise network techniques [12], etc. Many DNN methods [13][14] take inputs in time-frequency (T-F) domain from short time Fourier transform (STFT) or modified discrete cosine transform (MDCT), etc. Recent DNN-based codecs [15][16][17][18] model speech signals in time domain directly without T-F transformation. They are referred to as end-to-end methods, yielding competitive performance comparing with current speech coding standards, such as AMR-WB [19].

While DNN serves a powerful parameter estimation paradigm, they are computationally expensive to run on smart devices. Many DNN-based codecs achieve both low bitrates and high perceptual quality, two main targets for speech codecs [34][35][36], but with a high model complexity. A WaveNet based variational autoencoder (VAE) [18] outperforms other low bitrate codecs in the listening test, however, with 20 millions parameters, a too big model for real-time processing in a resource-constrained device. Similarly, codecs built on SampleRNN [37][38] can also be energy-intensive.

Conventional codecs achieve model efficiency by decomposing the whole task of speech signal quantization and reconstruction into multiple stages [39, 40]. For instance, [41] adopts multi-stage vector quantization (MSVQ) to discretize LPC coefficients in 4kbps. In comparison, end-to-end DNN systems tackle the problem in one gigantic model. It seems that the solution is simplified but tuning a model as such requires millions of parameters, a significant amount of training data and

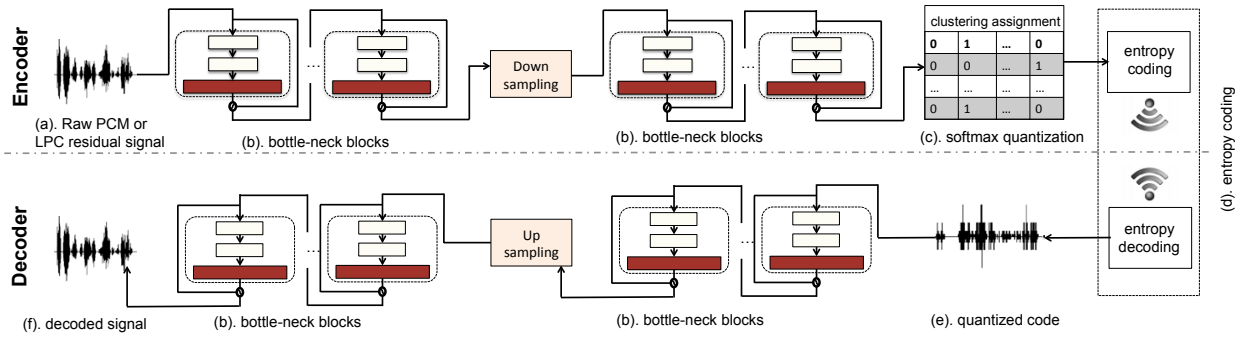


Figure 2.1: A schematic diagram for the end-to-end speech coding component module: some channel change steps are omitted.

advanced optimization techniques.

We focus on reducing the computational overhead of neural speech codecs by disassembling the "big" DNN into a list of "small" DNNs, and compressing the speech signal in a cascaded, or multi-staged, manner. That said, we need to re-define the conventional MSVQ technique with DNN techniques. The neural network based pipeline that hosts a list of compact codecs is dubbed as Cross-Model Residual Learning (CMRL). Note that each "small" DNN needs to be simplified such that the accumulative model complexity of this cascaded residual coding system does not surpass its baseline counterpart. A two-phase training scheme is also proposed to optimize CMRL.

2.2 A Simplified Autoencoder For End-to-End Speech Coding

Before introducing CMRL as a module carrier, we describe the component module to be hosted by CMRL.

Recently, an end-to-end DNN speech codec (referred to as Kankanahalli-Net) has shown competitive performance comparable to one of the standards (AMR-WB) [16]. We describe our component model derived from Kankanahalli-Net that consists of bottleneck residual learning [42], soft-to-hard quantization [43], and sub-pixel convolutional neural networks for upsampling [29]. Figure 2.1 depicts the component module.

In the end-to-end speech codec, we take $S = 512$ time domain samples per frame, 32 of which

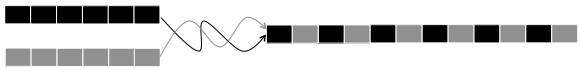


Figure 2.2: The interlacing-based upsampling process.

are windowed by the either left or right half of a Hann window and then overlapped with the adjacent ones. This forms the input to the first 1-D convolutional layer of C kernels, whose output is a tensor of size $S \times C$.

There are four types of non-linear transformations involved in this fully convolutional network: downsampling, upsampling, channel changing, and residual learning. The downsampling operation reduces S down to $S/2$ by setting the stride d of the convolutional layer to be 2, which turns an input example $S \times C$ into $S/2 \times C$. The original dimension S is recovered in the decoder with recently proposed sub-pixel convolution [43], which forms the upsampling operation. The super-pixel convolution is done by interlacing multiple feature maps to expand the size of the window (Figure 3.5). In our case, we interlace a pair of feature maps, and that is why in Table 2.1 the upsampling layer reduces the channels from 100 to 50 while recovers the original 512 dimensions from 256.

In this work, to simplify the model architecture we have identical shortcuts only for cross-layer residual learning, while Kankanahalli-Net employs them more frequently. Furthermore, inspired by recent work in source separation with dilated convolutional neural network [44], we use a "bottleneck" residual learning block to further reduce the number of parameters. This can lower the amount of parameters, because the reduced number of channels within the bottleneck residual learning block decreases the depth of the kernels. See Table 2.1 for the size of our kernels. Likewise, the input $S \times 1$ tensor is firstly converted to a $S \times C$ feature map, and then downsampled to $S/2 \times C$. Eventually, the code vector shrinks down to $S/2 \times 1$. The decoding process recovers it back to a signal of size $S \times 1$, reversely.

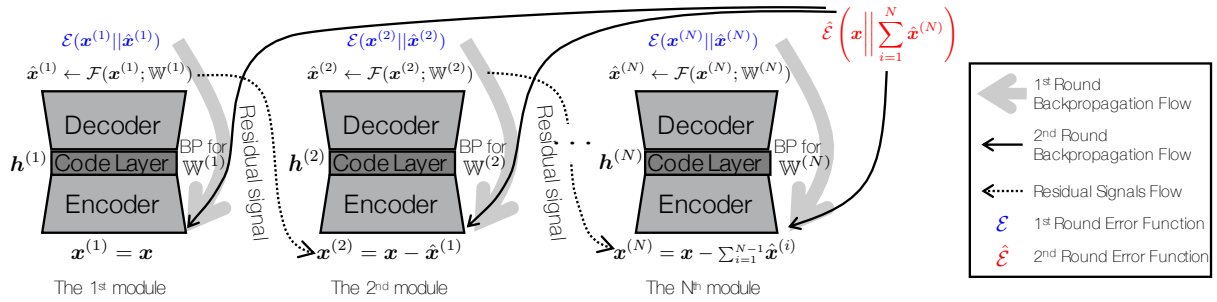


Figure 2.3: Cross-module residual learning pipeline

2.3 Proposed Cascaded Inter-Model Residual Learning Pipeline

2.3.1 The module carrier: CMRL

Figure 2.3 shows the proposed cascaded cross-module residual learning (CMRL) process. In CMRL, each module does its best to reconstruct its input. The procedure in the i-th module is denoted as $\mathcal{F}(\boldsymbol{x}^{(i)}; \mathbb{W}^{(i)})$, which estimates the input as $\hat{\boldsymbol{x}}^{(i)}$. The input for the i-th module is defined as

$$\boldsymbol{x}^{(i)} = \boldsymbol{x} - \sum_{j=1}^{i-1} \hat{\boldsymbol{x}}^{(j)}, \tag{2.1}$$

where the first module takes the input speech signal, i.e., $\boldsymbol{x}^{(1)} = \boldsymbol{x}$. The meaning is that each module learns to reconstruct the residual which is not recovered by its preceding modules. Note that module homogeneity is not required for CMRL: for example, the first module can be very shallow to just estimate the envelope of MDCT spectral structure while the following modules may need more parameters to estimate the residuals.

Each AE decomposes into the encoder and decoder parts:

$$\boldsymbol{h}^{(i)} = \mathcal{F}_{\text{enc}}(\boldsymbol{x}^{(i)}; \mathbb{W}_{\text{enc}}^{(i)}), \quad \hat{\boldsymbol{x}}^{(i)} = \mathcal{F}_{\text{dec}}(\boldsymbol{h}^{(i)}; \mathbb{W}_{\text{dec}}^{(i)}), \tag{2.2}$$

where $\boldsymbol{h}^{(i)}$ denotes the part of code generated by the i-th encoder, and $\mathbb{W}_{\text{enc}}^{(i)} \cup \mathbb{W}_{\text{dec}}^{(i)} = \mathbb{W}^{(i)}$.

The encoding process: For a given input signal $\boldsymbol{x}$, the encoding process runs all N AE

modules in a sequential order. Then, the bistring is generated by taking the encoder outputs and concatenating them: $\boldsymbol{h} = \left[\boldsymbol{h}^{(1)^\top}, \boldsymbol{h}^{(2)^\top}, \cdots, \boldsymbol{h}^{(N)^\top}\right]^\top$.

The decoding process: Once the bitstring is available on the receiver side, all the decoder parts of the modules, $\mathcal{F}_{\text{dec}}(\boldsymbol{x}^{(i)}; \mathbb{W}^{(i)}_{\text{dec}})\ \forall N$, run to produce the reconstructions which are added up to approximate the initial input signal with the global error defined as

$$\hat{\mathcal{E}}\left(\boldsymbol{x} \middle|\middle| \sum_{i=1}^{N} \hat{\boldsymbol{x}}^{(i)}\right). \tag{2.3}$$

2.3.2 The two-round training scheme

Intra-module greedy training: We provide a two-round training scheme to make CMRL optimization tractable. The first round adopts a *greedy* training scheme, where each AE tries its best to minimize the error: $\arg\min_{\mathbb{W}^{(i)}} \mathcal{E}(\boldsymbol{x}^{(i)} || \mathcal{F}(\boldsymbol{x}^{(i)}; \mathbb{W}^{(i)}))$. The greedy training scheme echoes a divide-and-conquer manner, leading to an easier optimization for each module. The thick gray arrows in Figure 2.3 show the flow of the backpropagation error to minimize the individual module error with respect to the module-specific parameter set $\mathbb{W}^{(i)}$.

Cross-module finetuning: The greedy training scheme accumulates module-specific error, which the earlier modules do not have a chance to reduce, thus leading to a suboptimal result. Hence, the second-round cross-module finetuning follows to further improve the performance by reducing the total error:

$$\underset{\mathbb{W}^{(1)} \ldots \mathbb{W}^{(N)}}{\arg\min}\ \hat{\mathcal{E}}\left(\boldsymbol{x} \middle|\middle| \sum_{i=1}^{N} \mathcal{F}\big(\boldsymbol{x}^{(i)}; \mathbb{W}^{(i)}\big)\right). \tag{2.4}$$

During the finetuing step, we first (a) initialize the parameters of each module with those estimated from the greedy training step (b) perform cascaded feedforward on all the modules sequentially to calculate the total estimation error in equation 2.3 (c) backpropagate the error to update parameters in all modules altogether (thin black arrows in Figure 2.3). Aside from the total reconstruction error equation 2.3, we inherit Kankanahalli-Net's other regularization terms, i.e., perceptual loss,

quantization penalty, and entropy regularizer.

2.3.3 Bitrate and entropy coding

The bitrate is calculated from the concatenated bitstrings from all modules in CMRL. Each encoder module produces S/d quantized symbols from the softmax quantization process (Figure 2.1 (e)), where the stride size d divides the input dimensionality. Let $c^{(i)}$ be the average bit length per symbol after Huffman coding in the i-th module. Then, $c^{(i)}S/d$ stands for the bits per frame. By dividing the frame rate, $(S-o)/f$, where o and f denote the overlap size in samples and the sampling rate, respectively, the bitrates per module add up to the total bitrate: $\xi_{\text{LPC}} + \sum_{i=1}^{N} \frac{fcS}{(S-o)d}$, where the overhead to transmit LPC coefficients is ξ_{lpc}=2.4kbps, which is 0 for the case with raw PCM signals as the input.

By having the entropy control scheme proposed in Kankanahalli-Net as the baseline to keep a specific bitrate, we further enhance the coding efficiency by employing the Huffman coding scheme on the vectors. Aside from encoding each symbol (i.e., the softmax result) separately, encoding short sequences can further leverage the temporal correlation in the series of quantized symbols, especially when the entropy is already low [45] [46]. We found that encoding a short symbol sequence of adjacent symbols, i.e., two symbols, can lower down the average bit length further in the low bitrates.

2.4 Experimental Results

We first show that for the raw PCM input CMRL outperforms AMR-WB and Kankanahalli-Net in terms of objective metrics in the experimental setup proposed in [16], where the use of LPC was not tested. Therefore, for the subjective quality, we perform MUSHRA tests [47] to show that CMRL with an LPC residual input works better than AMR-WB and OPUS at high bitrates.

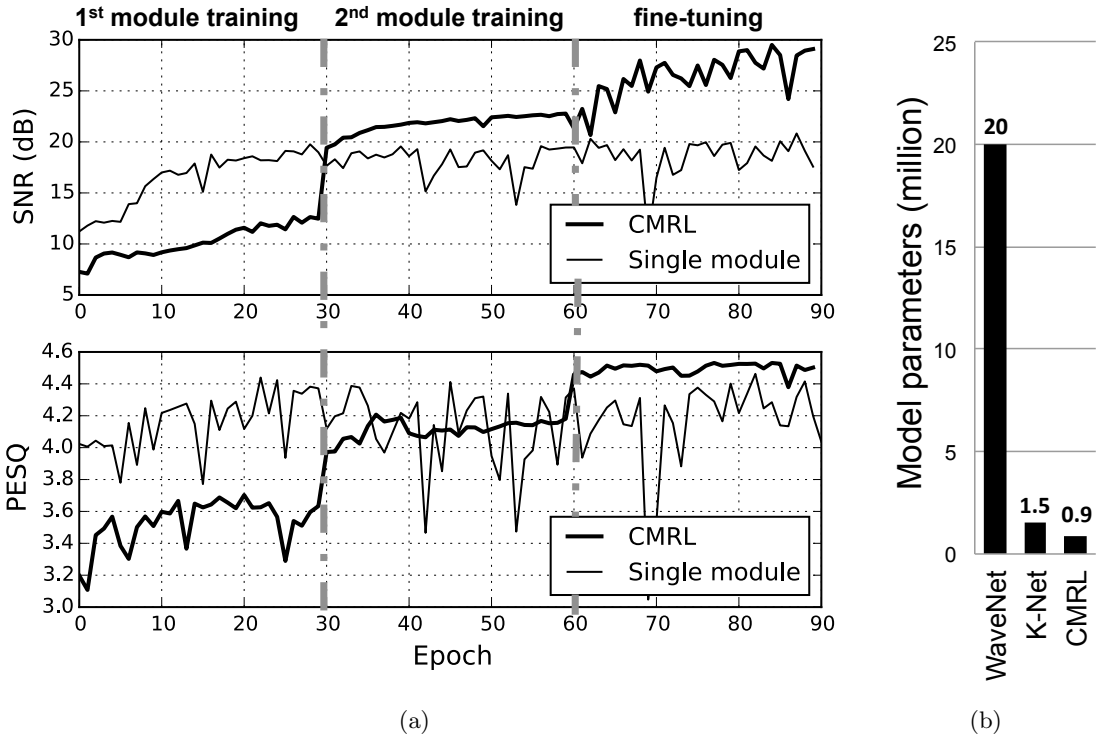
300 and 50 speakers are randomly selected from TIMIT [48] training and test datasets, respec-

Table 2.1: Architecture of the component module as in Figure 2.1. Input and output tensors sizes are represented by (width, channel), while the kernel shape is (width, in channel, out channel).

Layer	Input shape	Kernel shape	Output shape
Change channel	(512, 1)	(9, 1, 100)	(512, 100)
1st bottleneck	(512, 100)	$\begin{bmatrix} (9, 100, 20) \\ (9, 20, 20) \\ (9, 20, 100) \end{bmatrix} \times 2$	(512, 100)
Downsampling	(512, 100)	(9, 100, 100)	(256, 100)
2nd bottleneck	(256, 100)	$\begin{bmatrix} (9, 100, 20) \\ (9, 20, 20) \\ (9, 20, 100) \end{bmatrix} \times 2$	(256, 100)
Change channel	(256, 100)	(9, 100, 1)	(256, 1)
Change channel	(256, 1)	(9, 1, 100)	(256, 100)
1st bottleneck	(256, 100)	$\begin{bmatrix} (9, 100, 20) \\ (9, 20, 20) \\ (9, 20, 100) \end{bmatrix} \times 2$	(256, 100)
Upsampling	(256, 100)	(9, 100, 100)	(512, 50)
2nd bottleneck	(512, 50)	$\begin{bmatrix} (9, 50, 20) \\ (9, 20, 20) \\ (9, 20, 50) \end{bmatrix} \times 2$	(512, 50)
Change channel	(512, 50)	(9, 50, 1)	(512, 1)

tively. We consider two types of inputs in time-domain: raw PCM and LPC residuals. For the raw PCM input, the data is normalized to have a unit variance, and then directly fed to the model. For the LPC residual input, we conduct a spectral envelope estimation on the raw signals to get LPC residuals and corresponding coefficients. The LPC residuals are modeled by the proposed end-to-end CMRL pipeline, while the LPC coefficients are quantized and sent directly to the receiver side at 2.4 kbps. The decoding process recovers the speech signal based on the LPC synthesis procedure using the LPC coefficients and the decoded residual signals.

We consider four bitrate cases: 8.85 kbps, 15.85 kbps, 19.85 kbps and 23.85 kbps. All convolutional layers in CMRL use 1-D kernel with the size of 9 and the Leaky Relu activation. CMRL hosts two modules: each module is with the topology as in Table 2.1. Each residual learning block contains two bottleneck structures with the dilation rate of 1 and 2. Note that for the lowest bitrate case, the second encoder downsamples each window to 128 symbols. The learning rate is 0.0001 to train the first module, and 0.00002 for the second module. Finetuning uses 0.00002 as the learning rate, too. Each window contains 512 samples with the overlap size of 32. We use Adam optimizer



(a) (b)

Figure 2.4: (a) SNR and PESQ per epoch (b) model complexity

[49] with the batch size of 128 frames. Each module is trained for 30 epochs followed by finetuning until the entropy is within the target range.

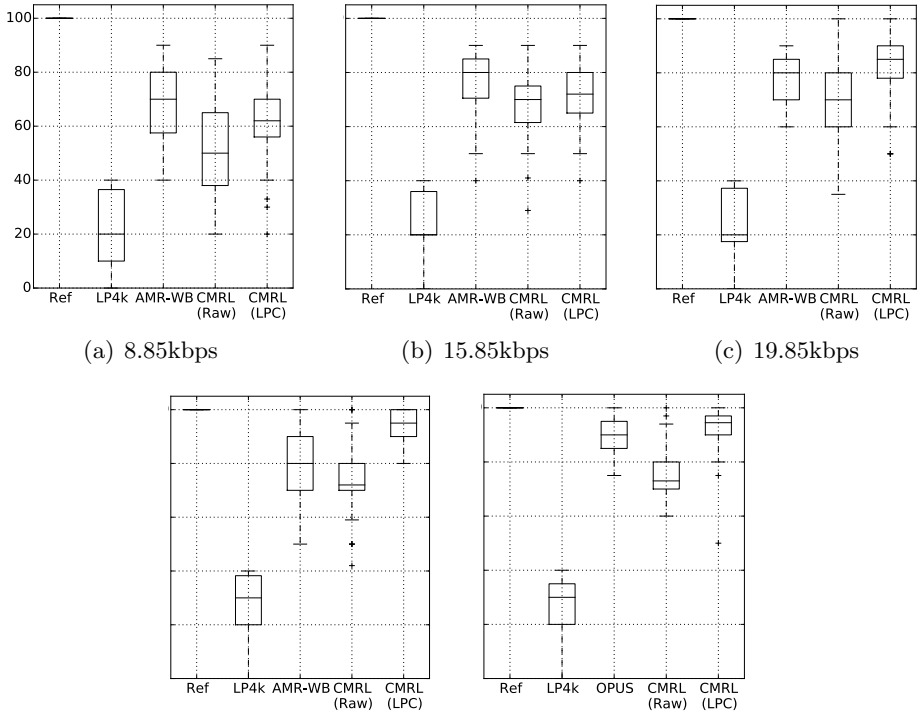
2.4.1 Objective test

We evaluate 500 decoded utterances in terms of SNR and PESQ with wide band extension (P862.2) [50]. Figure 2.4 (a) shows the effectiveness of CMRL against a system with a single module in terms of SNR and PESQ values per epoch. The single module is with three more bottleneck blocks and twice more codes for a fair comparison. It is trained for 90 epochs with other hyperparameters are unaltered. For both SNR and PESQ, the plot shows a noticeable performance jump as the second module is included, followed by another jump by finetuning.

Table 2.2 compares CMRL with AMR-WB and Kankanahalli-Net at four bitrates for the raw PCM input case. CMRL achieves both higher SNR and PESQ at all four bitrate cases. Note that the SNR for CMRL at 8.85 kbps is greater than AMR-WB at 23.85 kbps. CMRL also gives a

Table 2.2: SNR and PESQ scores on raw PCM test signals.

Metrics	SNR (dB)				PESQ			
Bitrate (kbps)	8.85	15.85	19.85	23.85	8.85	15.85	19.85	23.85
AMR-WB	9.82	11.93	12.46	12.73	3.41	3.99	4.09	4.13
K-Net	-	-	-	-	3.63	4.13	4.22	4.30
CMRL	**13.45**	**16.35**	**17.18**	**17.33**	**3.69**	**4.21**	**4.34**	**4.42**



(a) 8.85kbps (b) 15.85kbps (c) 19.85kbps

(d) 23.85kbps (e) 23.85kbps

Figure 2.5: MUSHRA test results. From (a) to (d): the performance of CMRL on raw and LPC residual input signals compared against AMR-WB at different bitrates. (e) An additional test shows that the performance of CMRL with the LPC input competes with OPUS, which is known to outperform AMR-WB in 23.85kbps.

better PESQ score at 15.85 kbps than AMR-WB at 23.85 kbps.

2.4.2 MUSHRA test

Figure 2.5 shows MUSHRA test results done by six audio experts on 10 decoded test samples randomly selected with gender equity. At 19.85 kbps and 23.85 kbps, CMRL with LPC residual inputs outperforms AMR-WB. At lower bitrates though, AMR-WB starts to work better. CMRL on raw PCM is found less favored by listeners. We also compare CMRL with OPUS in the high

bitrate where OPUS is known to perform well, and find that CMRL slightly outperforms OPUS[1].

2.4.3 Model complexity analysis

The cross-module residual learning simplifies the topology of each component module. Hence, CMRL has less than 5% of the model parameters compared to the WaveNet based codec [18], and outperforms Kankanahalli-Net with 40% less model parameters. Figure 2.4 (b) summarizes the comparison.

2.5 Summary

In this work, we demonstrated that CMRL as a lightweight model carrier for DNN based speech codecs can compete with the industrial standards. By cascading two end-to-end modules, CMRL achieved a higher PESQ score at 15.85 kbps than AMR-WB at 23.85 kbps. We also showed that CMRL can consistently outperform a state-of-the-art DNN codec in terms of PESQ. CMRL is compatible with LPC, by having it as the first pre-processing module and by using its residual signals as the input. CMRL, coupled with LPC, outperformed AMR-WB in 19.85 kbps and 23.85 kbps, and worked better than OPUS at 23.85 kbps in the MUSHRA test. More work is required to examine other module structures to further improve the performance at low bitrates.

[1]Samples are available at `http://saige.sice.indiana.edu/research-projects/neural-audio-coding`

Chapter 3

COLLABORATIVE QUANTIZATION: BRIDGING LPC WITH DNN

3.1 Preprocessing With Linear Predictive Coding (LPC)

Speech coding quantizes speech signals into a compact bitstream for efficient transmission and storage in telecommunication systems [51, 52]. The design of speech codecs is to address the trade-off among low bitrate, high perceptual quality, low complexity and delay, etc [7, 8]. Most speech codecs are classified into two categorizes, *vocoders* and *waveform* coders [9]. Vocoders use few parameters to model the human speech production process, such as vocal tract, pitch frequency, etc [10]. In comparison, waveform coders compress and reconstruct the waveform to make the decoded speech similar to the input as "perceptually" as possible. Conventional vocoders are computationally efficient and can encode speech at very low bitrates, while waveform coders support a much wider bitrate range with scalable performance and are more robust to noise.

In both conventional vocoders and waveform coders, linear predictive coding (LPC) [53], an all-pole linear filter, serves a critical component, as it can efficiently model power spectrum with only a few coefficients through Levinson-Durbin algorithm [10]. For vocoders, the LPC residual is then modeled as a synthetic excitation signal with a pitch pulse train or white noise component [54]. On the other hand, for waveform coders, such as Opus [55], Speex [56] and AMR-WB [57], the residual is directly compressed to the desired bitrate before being synthesized to the decoded signal.

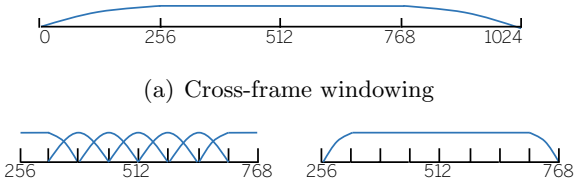
LPC is useful in modern neural speech codecs, too. While generative autoregressive models, such as WaveNet, have greatly improved the synthesized speech quality [58], it comes at the cost of model complexity during the decoding process [59]. For example, vector quantized variational autoencoders (VQ-VAE) with WaveNet decoder achieves impressive speech quality at a very low

bitrate of 1.6 kbps, yet with approximately 20 million trainable parameters [60]. To make such a system more efficient, LPC can still unload computational overheads from neural networks. LPCNet combines WaveRNN [61] and LPC to shrink down the complexity to 3 GFLOPS which enables real-time coding [62, 63]. Nevertheless, LPCNet, as a vocoder, provides a decent performance at 1.6 kbps, but does not scale up to transparent quality. In terms of the neural waveform coder, CMRL [33] uses LPC as a pre-processor and a variation of [32] to model the LPC residual to match the state-of-the-art speech quality with only 0.9 million parameters. However, both LPCNet and CMRL take LPC another blackbox shoehorned into advanced neural networks. Using LPC as a deterministic pre-processor can be sub-optimal, as its bit allocation is pre-defined and not integrated to model training.

To better incorporate LPC with neural networks towards scalable waveform coding with low model complexity, we propose a collaborative quantization (CQ) scheme where LPC quantization process is trainable. Coupled with the other neural network autoencoding modules for the LPC residual coding, the proposed quantization scheme learns the optimal bit allocation between the LPC coefficients and the other neural network code layers. With the proposed collaborative training scheme, CQ outperforms its predecessor at 9 kbps, and can scale up to match the performance of the state-of-the art codec at 24 kbps with a much lower complexity than many generative models.

3.2 Proposed Quantization Scheme For A Better Pivot

In the CMRL pipeline, LPC module serves a pre-processor with a fixed bitrate of 2.4 kbps. While it can effectively model the spectral envelope, it may not fully benefit the consequent residual quantization. For example, for a frame that LPC cannot effectively model, CQ can weigh more on the following AEs to use more bits, and vice versa. In this section, we break down the LPC process to make its quantization module trainable, along with the other AE modules in CMRL which are to recover the LPC residual as best as possible.



(a) Cross-frame windowing

(b) Sub-frame windowing (c) Synthesis windowing

Figure 3.1: LPC windowing schemes

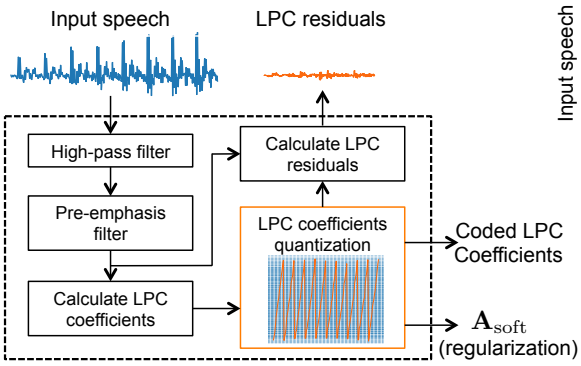


Figure 3.2: The trainable LPC analyzer

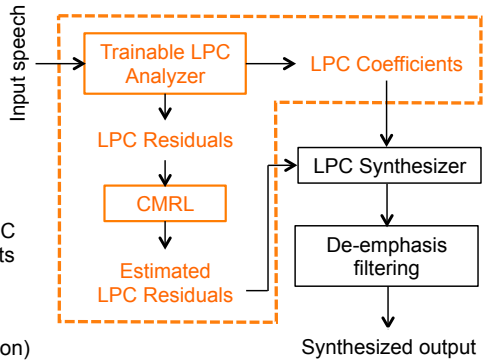


Figure 3.3: Overview of the CQ system.

3.2.1 Trainable LPC analyzer

Our goal is to incorporate LPC analysis into the CMRL pipeline so that it outsources the LPC coefficient quantization to the neural network training algorithm. The trainable LPC analyzer is derived from AMR-WB [64] with several necessary adjustments to be compatible with neural network computational paradigm.

High-pass filtering and pre-emphasizing: Given the input speech, we first adopt high-pass filtering and pre-emphasizing as in [64]. A high-pass filter is employed with a cut off frequency of 50 Hz. The pre-emphasis filter is $H_{emp}(z) = 1 - 0.68z^{-1}$, and the de-emphasis filter is employed to remove artifacts in the high frequencies.

Data windowing for LPC coefficients calculation: The pre-emphasis filtered utterances are segmented to frames of 1024 samples. Each frame is windowed before LPC coefficients are calculated. As shown in Fig. 3.1 (a), the symmetric window has its weight emphasized on the middle 50% samples: first 25% part is the left half of a Hann window with 512 points; the middle

50% is a series of ones; and the rest 25% part is the right half of the Hann window. Then, the linear prediction is conducted on the windowed frame in time domain s. For the prediction of the t-th sample, $\hat{s}(t) = \sum_i a_i s(t-i)$, where a_i is the i-th LPC coefficient. The frames are with 50% overlap. The LPC order is set to be 16. We use Levinson Durbin algorithm [10] to calculate LPC coefficients. They are are represented as line spectral pairs (LSP) [65] which are more robust to quantization.

Trainable LPC quantization: We then employ the trainable softmax quantization scheme to LPC coefficients in LSP domain, to represent each coefficient with its closest centroid. For each windowed frame $\boldsymbol{x}$, $\boldsymbol{h}_{\text{LPC}} = \mathcal{F}_{LPC}(\mathbf{x})$ gives corresponding LPC coefficients in the LSP representation. The rest of the process is the same with the softmax quantization process, although this time the LPC-specific centroids $\boldsymbol{b}_{\text{LPC}}$ should be learned and be used to construct the soft assignment matrix. In practice, we set LPC order to be 16, and the number of centroids to be 256 (i.e., 8 bits). Hence, the size of the soft and hard assignment matrices is 16×256, each of whose rows is a probability vector and a one-hot vector, respectively.

Data windowing for LPC residual calculation: We use a sub-frame windowing technique to calculate residuals (Fig. 3.1 (b)). For a given speech frame and its quantized LPC coefficients, we calculate residuals for each sub-frame, individually. The middle 50% of the 1024 samples, for example, [256:768] for the first analysis frame that covers [0:1024] and [768:1280] for the second frame of [512:1536], is decomposed into seven sub-frames, each with the size 128 and 50% overlap. Out of the seven sub-frames, the middle five are windowed by a Hann function with 128 points; the first and last frames are asymmetrically windowed, as shown in Fig. 3.1 (b). The residual is calculated with the seven sub-frames on the middle 512 samples, which amount to 50% of the frame. Hence, given the 50% analysis frame overlap, there is no overlap between residual segments.

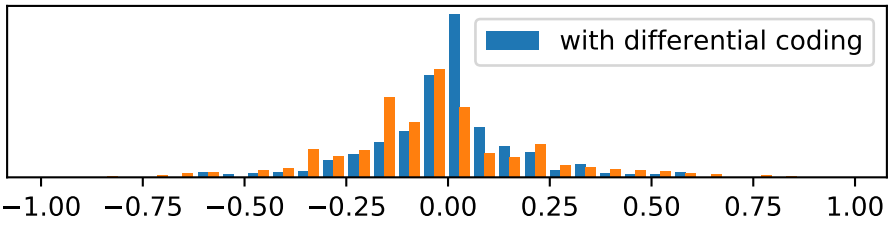


Figure 3.4: Differential coding enables a more centralized distribution

3.2.2 Residual coding

The LPC residual, calculated from the trainable LPC analyzer (Fig. 3.2), is compressed by the 1D-CNN AEs. In this work, we employ differential coding [66] to the output of encoders, $\boldsymbol{h} = [h_0, h_1, \cdots, h_{m-1}]$ where m is the length of code per frame for each AE. Hence, the input scalar to the softmax quantization is $\Delta h_i = h_i - h_{i-1}$. Consequently, the quantization starts from a more centralized real-valued "code" distribution (Fig.3.4). As illustrated in Fig. 3.3, both the quantization of LPC coefficients and residual coding with CMRL are optimized together. With this design, the purpose of LPC analysis is not just to minimize the residual signal energy as much as possible [67], but to find a pivot which also facilitates the residual compression from following CMRL modules.

3.3 Experimental Results

We consider four bitrate cases 9, 16, 20, and 24 kbps, with the sample rate of 16 kHz. The training set contains 2.6 hours of speech from 300 speakers randomly selected from the TIMIT training set. 50 speakers are randomly selected from the test set. At test time, each frame has 512 samples with an overlap of 32 samples. The overlap region is windowed by Hann function (Fig.3.1(c)). For 24 kbps, we cascade the LPC module and two AEs as in [33], but we use only one AE for the LPC residual coding for other three bitrates. For 16 and 20 kbps cases, the code layer is downsampled with a convolutional layer of stride 2; for the 9 kbps case, we use two downsampling layers of stride 2. We use Adam optimizer [68] with the batch size of 128, learning rate of 0.0002 for 30 epochs,

Table 3.1: MOS-LQO scores computed from PESQ-WB

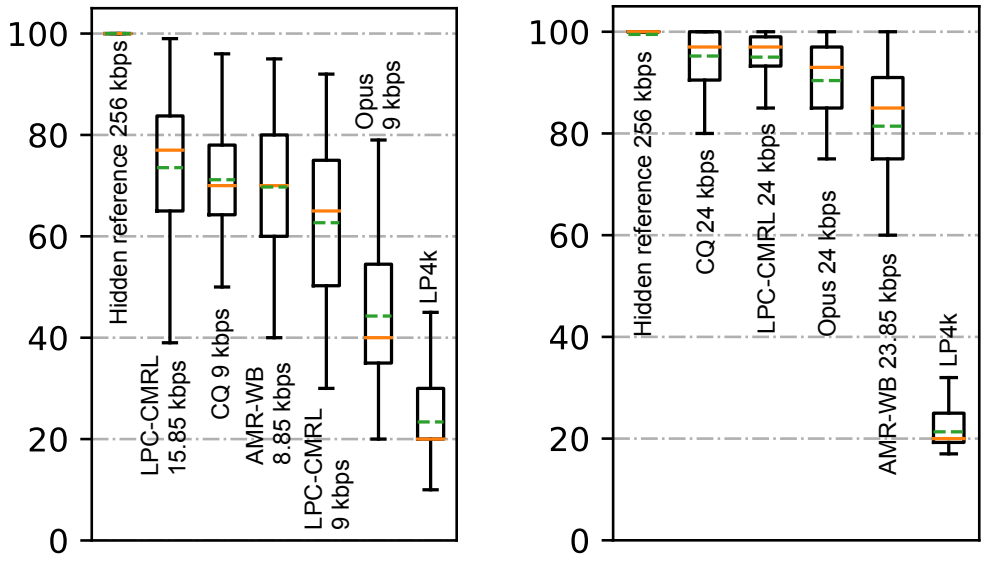
	AMR-WB	Opus	LPC-CMRL	CQ
∼9 kbps	3.48	3.42	3.01	3.69
∼16 kbps	3.99	4.30	3.26	3.98
∼20 kbps	4.09	4.43	3.67	4.08
∼24 kbps	4.17	4.47	4.15	4.17

followed by finetuning until the entropy is within the target range.

Recent works on generative model based speech synthesis systems [62, 59] have reported that PESQ [69] or its successor POLQA [70] cannot accurately evaluate the synthesized speech quality. In fact, we also find that there is a discrepancy between PESQ and the actual MOS. Still, we report MOS-LQO scores in Table 3.1 as the proposed method is based on waveform reconstruction.

We conduct two MUSHRA-like [23] sessions corresponding to two bitrate settings. Each session includes ten trials on gender-balanced utterances randomly chosen from the test set. A low-pass anchor at 4kHz and the hidden reference signal are included in both settings, with eleven audio experts as the subjects. The lower bitrate setting refers to the performance around 9 kbps, including AMR-WB [64] at 8.85 kbps, Opus [55] at 9 kbps, LPC-CMRL [33] at 9 and 16 kbps, and CQ at 9 kbps. The higher bitrate session uses decoded signals from codes with around 24 kbps bitrate. The competing models are AMR-WB at 23.85 kbps, Opus at 24 kbps, the proposed CQ method, and the LPC-CMRL counterpart at 24 kbps.

First, we can see that CQ outperforms LPC-CMRL at the same bitrate, especially in low bitrate setting (Fig. 3.5). In higher bitrate setting, both LPC-CMRL and CQ outperform AMR-WB and Opus in the MUSHRA test. None of these methods add very audible artifacts. One explanation for the result is that AMR-WB does not code all 8kHz wide bandwidth, but up to 7kHz, while our model maintains the energy of decoded signals at high frequencies, and therefore yield less-muffled speech. However, as the bitrate decreases, some human subjects tend to be more negative towards the artifacts (from LPC-CMRL) which become audible than the moderately muffled speech (from AMR-WB). That explains why LPC-CMRL is less favored than AMR-WB at low bitrate. As



(a) ∼9 kbps (b) ∼24 kbps

Figure 3.5: MUSHRA results in box-plots (Orange solid lines represent medians, and green dashed lines represent means).

was expected, when the LPC coefficient quantization is collaboratively learned along with residual coding, the artifact is suppressed—CQ-9 outperforms LPC-CMRL-9 with a noticeable margin[1].

We use the number of trainable parameters to measure the model complexity, as it determines the floating point operation rate, memory space, and execution time, etc. Each AE, used in CQ and CMRL, contains 0.45 million parameters (Table 2.1). We use one AE for CQ-16 and CQ-20 kbps cases, and two AEs for the 24 kbps case. The AE in CQ-9 kbps is 0.67 million parameters as it contains two downsampling and upsampling layers. Admittedly, the decoder of CQ is still more complex than the conventional codecs. but it is much simpler than the other WaveNet vocoder-based coding systems, e.g., the VQ-VAE with WaveNet has 20 million parameters, although it gives impressive speech quality with only 1.6 kbps.

[1]Decoded samples are available at `http://pages.iu.edu/~zhenk/speechcoding.html`

3.4 Summary

In this work, we proposed a lightweight and scalable waveform neural codec. The method incorporates merits from both advanced end-to-end neural network architectures and conventional DSP techniques. With collaborative quantization (CQ), LPC coefficient quantization becomes a trainable component to be jointly optimized with the residual quantization. This helps CQ outperform its predecessor and Opus at 9 kbps, and show comparable performance to AMR-WB at 8.85 kbps. The method is with much lower complexity in terms of the amount of parameters than other competitive neural speech coding models.

Chapter 4

TOWARDS A PERCEPTUAL LOSS: PSYCHOACOUSTICAL CALIBRATION

4.1 Psychoacoustics in Audio Coding

Audio coding, a fundamental set of technologies in data storage and communication, is to compress the original signal into a compact bitstream (encoding) with a minimal bitrate, while not sacrificing the perceptual quality of the recovered one in the waveform domain (decoding) [71, 22]. A conventional lossy audio codec consists of an encoder and decoder, where the encoder converts the input signal into a bitstream and the decoder faithfully reconstructs the signal from the bitstream as the output. In this section, we focus on the lossy codecs, which typically allow a loss of information during the encoding and decoding process only in inaudible audio components. To this end, psychacoustics is employed to quantify the audibility in both time and frequency domains. For example, a successful commercial audio codec, such as MPEG-1 Audio Layer III (also known as MP3), achieves a transparent quality at 128 kbps by using psychoacoustic model (PAM). Its encoder employs a dynamic bit-allocation scheme, which assigns bits first to the more audible frequency subbands [72]. In other words, if a spectral component masked by its adjacent louder one, the encoder saves bits for it allowing more inaudible quantization error. While the MP3 decoder is available to the public, the perceptual encoder of MP3 is kept in the black box.

Although deep neural networks (DNNs) have been a powerful data-driven paradigm in various audio signal processing scenarios [73, 74, 75, 58, 76], it is by no means easy to replace conventional audio codecs without utilizing the domain knowledge but just data. Most recent works have led to a high coding gain in speech coding, but at the cost of increased model complexity: a basic U-Net contains approximately 10 million parameters [77]; in [60] vector quantized variational autoencoders (VQ-VAE) [30] employs WaveNet [58] as a decoder, yielding a competitive speech quality at 1.6

kbps, but with 20 million parameters. Models as such are beyond the capacity of low-power devices, which is a common environment for decoders. While there have been efforts in reducing the model complexity for speech coding [62], a general-purpose neural non-speech audio coding system is yet to come.

To achieve a better trade-off between performance and efficiency, a perceptually meaningful training objective is critical as shown in speech enhancement [78, 79, 80]. For speech coding, there have been efforts to introduce more perceptually inspired, yet objective metrics to train neural networks, such as short-time objective intelligibility (STOI) [81] and perceptual evaluation of speech quality (PESQ) [82], as in [83, 84]. However, these metrics do not faithfully correlate with subjective metrics, such as MUSHRA [23], and does not apply well to audio coding as psychoacoustics does, neither.

Hence, we present psychoacoustic calibration schemes to improve the neural network optimization process, as an attempt to enable an efficient and high-fidelity neural audio coding (NAC) system. With the global masking threshold calculated from the standard PAM [22], we firstly present a weighting scheme that makes the optimization process focus more on audible frequency subbands, where the masking effect is weak, while going easy otherwise. In addition, the scheme modulates the coding artifact such that it is well below the global masking threshold, similarly to MP3's bit allocation algorithm. This is, to our best knowledge, the first method to directly incorporate psychoacoustics to neural audio coding.

4.2 Psychoacoustical Calibration in Neural Audio Codec

Psychoacoustic models (PAM) are used in audio coding to calculate masking effects for the input signal as a function of frequency. Fig. 4.1 shows the logarithmic power spectral density (PSD) curve of a signal and its associated global masking threshold to explain the simultaneous masking effect. We will be based on PAM-1 as defined in [22]. Given an input frame, PAM-1 (a) calculates

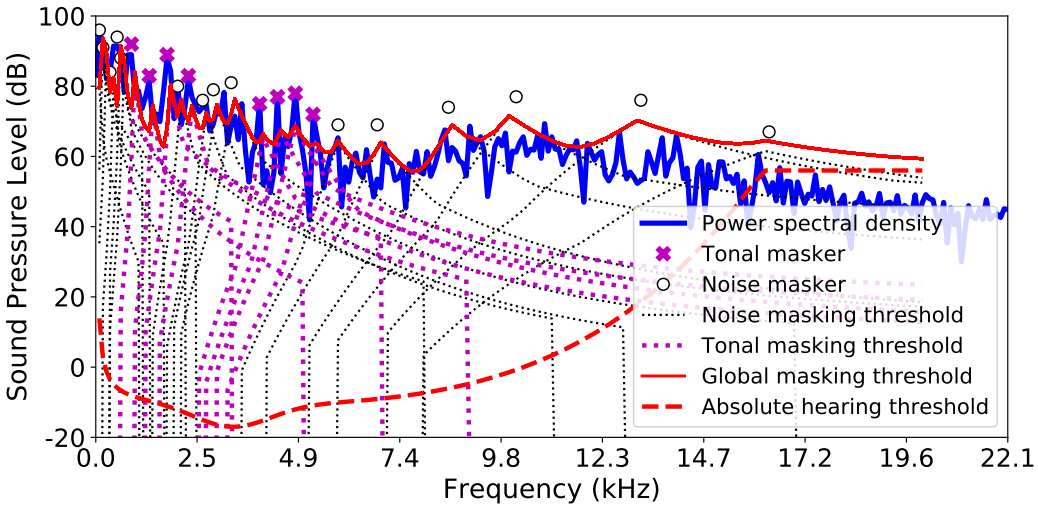


Figure 4.1: Visualization of the masker detection, individual and global masking threshold calculation for an audio input.

the logarithmic PSD $\boldsymbol{p}$; (b) detects tonal and noise maskers, followed by decimation; (c) calculates masking threshold for individual maskers; (d) and accumulates all masking thresholds in (c), along with the absolute hearing threshold, to acquire the global masking threshold $\boldsymbol{m}$.

Signal-to-mask ratio (SMR) is the difference between two curves: $\boldsymbol{p} - \boldsymbol{m}$. Typically, the bit allocation algorithm calculates the mask-to-noise ratio (MNR) by subtracting SMR from Signal-to-Noise Ratio (SNR), where noise incurred from quantization. Hence, a high MNR value means that the quantization noise is masked out by an adjacent loud spectral component. Based on MNR, the encoder prioritizes the iterative bit allocation process: the subband with the lowest MNR is assigned with more bits earlier, such that the minimum MNR is maximized [85, 86, 87]. Note that we are based on the primitive PAM-1 version due to the lack of reproducibility of the advanced commercial PAM implementations. However, our coding gain using PAM-1 implies that a more precise PAM implementation will improve the performance further.

Our proposed perceptual loss terms are derived from PAM-1's masking curves to calibrate the basic loss function. It is a dynamic process as masking curves are contingent upon each specific input frame. We propose two PAM-based terms.

4.2.1 Priority weighting

During training we first estimate the logarithmic PSD $\boldsymbol{p}$ out of an input frame $\boldsymbol{s}$, as well as its corresponding global masking threshold $\boldsymbol{m}$. We define a perceptual weight vector, $\boldsymbol{w} = \log_{10}(\frac{10^{0.1\boldsymbol{p}}}{10^{0.1\boldsymbol{m}}} + 1)$, which is the log ratio between the signal power and the masked threshold, rescaled from decibel. Accordingly, we define a weighting scheme that pays more attention to the unmasked frequencies:

$$\mathcal{L}_3(\boldsymbol{s}||\hat{\boldsymbol{s}}) = \sum_i \sum_f w_f \left(x_f^{(i)} - \hat{x}_f^{(i)} \right)^2, \tag{4.1}$$

where $x_f^{(i)}$ and $\hat{x}_f^{(i)}$ are the f-th magnitude of the Fourier spectra of the input and the recovered signals for the i-th CMRL module. The intuition is that, if the signal's power is greater than its masking threshold at the f-th frequency subband, i.e. $p_f > m_f$, the model tries hard to recover this audible tone precisely: a large w_f enforces it. Otherwise, for a masked tone, the model is allowed to generate reconstruction error. The weights are bounded between 0 and ∞, whose smaller extreme says that the masking threshold is very large and any sound, such as the reconstruction error at that frequency bin, is not audible. Comparing to $\mathcal{L}_2$, which also accents certain subbands over the others, the priority ranking from $\mathcal{L}_3$ is not fixed, but dependent on each input frame.

4.2.2 Noise modulation

The priority weighting scheme can result in an accidentally loud reconstruction noise, exceeding the mask value m_f, when w_f is set to be small. Our second psychoacoustic loss term is to modulate the reconstruction noise by directly exploiting MNR, $\boldsymbol{m} - \boldsymbol{n}$, where $\boldsymbol{n}$ is the logarithmic PSD of the reconstruction residual $\boldsymbol{s} - \hat{\boldsymbol{s}}$. Since the bit allocation algorithm in the MP3 decoder is to maximize MNR iteratively, it is natural for us to employ this concept as a loss term. In particular, we propose

a two-term solution which maximizes both the overall MNR and the lower bound:

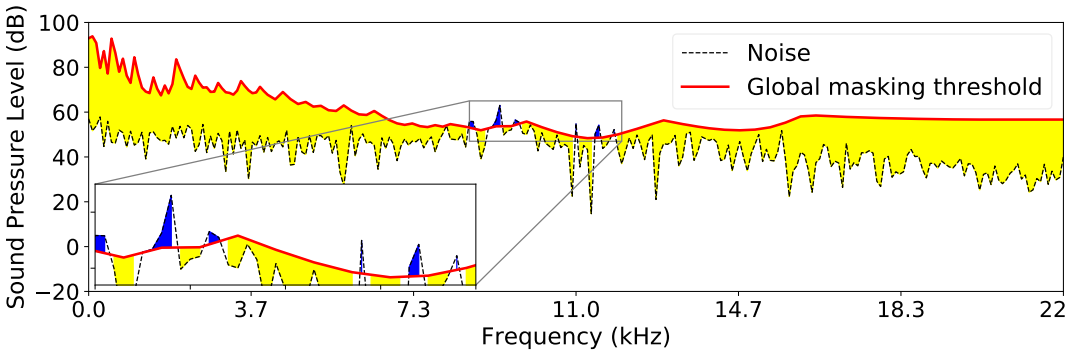
$$\mathcal{L}_4 = \sum_i \Big(\sum_f \big(n_f^{(i)} - m_f^{(i)} \big) \Big) - \min_f \big(m_f^{(i)} - n_f^{(i)} \big). \tag{4.2}$$

Note that the maximization of the sum of MNR turned into loss minimization by changing the sign. Meanwhile, the second term captures the minimum MNR value in the spectrum, which needs to be maximized, too. Fig. 4.2 shows that noise modulation detects the lower bound of MNR and widen the gap between the mask and the noise. Although the SNRs are similar, the audio clip in Fig. 4.2 (b) is of higher perceptual quality as the artifact is completely masked.

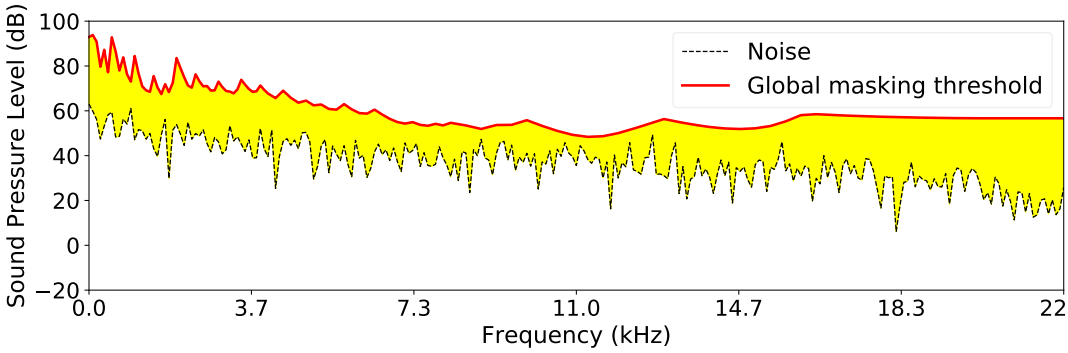
4.3 Experimental Results

Our training dataset consists of 1,000 clips from commercial music, each of which is about 20 seconds long, amounting to about 5.5 hours of play time. 13 genres are covered. All clips are downmixed to a monaural channel. The sampling rate is 44.1 kHz and downsampled to 32 kHz for the lower bitrate setup. Each frame contains $T = 512$ samples with an overlap of 32 samples, where half of a Hann window of 64 samples are used. For training, hyperparameters were found based on validation with another 104 clips; 128 frames for the batch size; $\alpha = 300$ for the initial softmax scaling factor ; 2×10^{-4} for the initial learning rate of the Adam optimizer [68], while 2×10^{-5} for the second cascaded modules; 64 and 32 kernels for the soft-to-hard quantization for low and high bitrate cases, respectively; 50 and 30 for the number of epochs to train the first and the second modules in CMRL, respectively.

To validate the perceptual loss terms, we train our NAC based on four different objective



(a) No noise modulation (Model C)



(b) With noise modulation (Model D)

Figure 4.2: Noise modulation maximizes the average MNR while boosting the lower bound, too.

functions by gradually adding the loss terms to the final loss:

$$\mathcal{L} = \mathcal{L}_1 : \text{Model-A}, \quad \mathcal{L} = \lambda_1\mathcal{L}_1 + \lambda_2\mathcal{L}_2 : \text{Model-B},$$

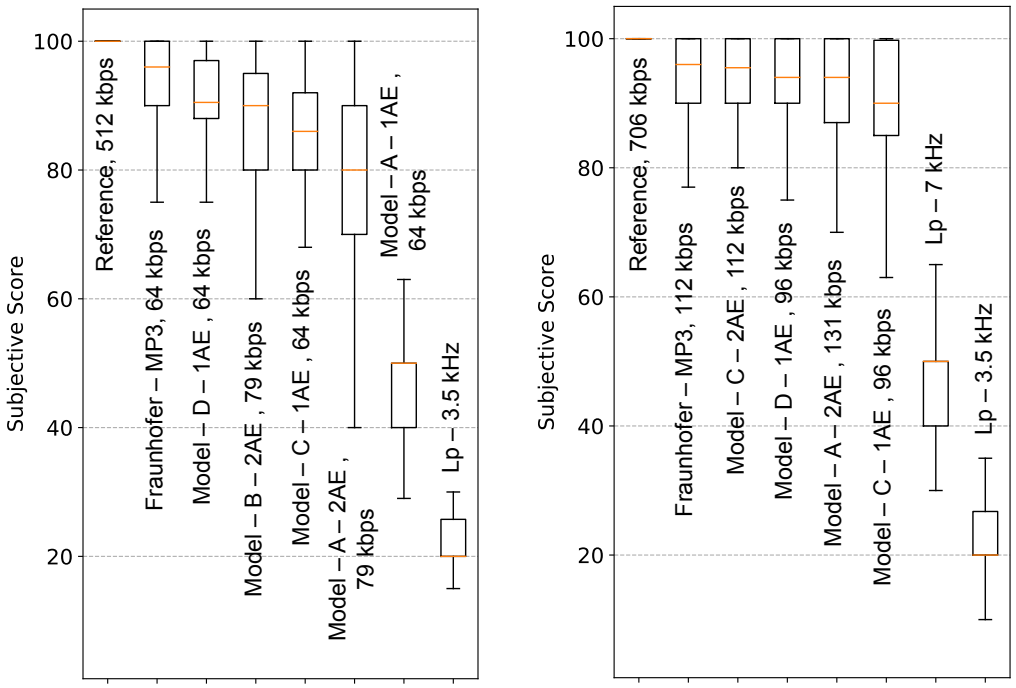
$$\mathcal{L} = \lambda_1\mathcal{L}_1 + \lambda_2\mathcal{L}_2 + \lambda_3\mathcal{L}_3 : \text{Model-C},$$

$$\mathcal{L} = \lambda_1\mathcal{L}_1 + \lambda_2\mathcal{L}_2 + \lambda_3\mathcal{L}_3 + \lambda_4\mathcal{L}_4 : \text{Model-D},$$

where the blending weights are found via validation: $\lambda_1 = 60, \lambda_2 = 5, \lambda_3 = 1, \lambda_4 = 5$.

Each model configuration is also denoted by the number of CMRL modules and the target bitrate, e.g., "Model-D-1AE, 96 kbps" for a model with only 1 autoencoding module (0.45M parameters), trained by all loss terms, whose bitrate is 96 kbps.

Two MUSHRA tests [23] are conducted for low and high bitrate settings by ten audio experts. Each session includes 13 trials, each representing a musical genre, randomly selected from the



(a) Low bitrate case (b) High bitrate case

Figure 4.3: Subjective scores from the MUSHRA tests.

unseen test set. Fig. 4.3 summarizes the test results[1].

The low bitrate session targets at 64 kbps with the sample rate of 32 kHz. Aside from the hidden reference and the anchor (low-pass filtered at 3.5 kHz), we compare the commercial MP3 codec from Adobe Audition® (licensed from Fraunhofer IIS and Thomson), along with our five NAC systems. As illustrated in Fig. 4.3 (a), the performance from models trained purely on MSE in time domain is far from competitive (Model-A); applying the loss term on mel-scale filter banks improves the performance (Model-B-2AE, 79 kbps > Model-A-2AE, 79 kbps); Model-D with both PAM-inspired loss terms got the highest subjective score among the NAC systems. Note that the performance is less appealing without the noise modulation step (Model-D-1AE, 64 kbps > Model-C-1AE, 64 kbps). Lastly, Model-C and D also show that the perceptual loss can reduce the model complexity: they use only one module with 0.45 million parameters and achieve better or equivalent performance than Model-A and B that employ two modules with 0.9 million parameters

[1]Samples are available at `http://kaizhen.us/neural-audio-coding.html`

at a higher bitrates (79 kbps).

The high bitrate session includes the hidden reference, two anchors (filtered at 3.5 kHz and 7 kHz), the commercial MP3 codec at 112 kbps and 44.1 kHz, and four NAC systems. In Fig. 4.3 (b), the basic NAC system (Model-A) yields a decent performance at 131 kbps. With psychoacoustic priority weighting, our model achieves almost transparent quality similar to MP3 at the same bitrate (Model-C-2AE, 112 kbps $\approx$ MP3, 112 kbps). Having both priority weighting and noise modulation, the model with only one CMRL module (thus half of the parameters) at a lower bitrate (96 kbps) competes the basic model at a 36.5% higher bitrate (Model-D-1AE, 96 kbps $\approx$ Model-A-2AE, 131 kbps).

4.4 Summary

We showed that incorporating the simultaneous masking effect in the objective function is advantageous to NAC in terms of the coding gain and model efficiency. Although the system is based on PAM-1, it successfully proved the concept and suggests that a more advanced PAM, e.g., by employing temporal masking, will improve the performance further. We also publicized all source codes[2].

[2]Available at `https://github.com/cocosci/pam-nac/`.

Chapter 5

FUTURE RESEARCH PLAN

5.1 From Wave-U-Net To a Modularized Neural Codec

Based on an orthodox autoencoder concept, U-Net [88] links each component from the encoder side to the corresponding component of the decoder, which is widely used in image segmentation. Fig. 5.1 summarizes a basic topology for U-Net. These days, there have been many variations of U-Net used in speech enhancement. For instance, one variation Wave-U-Net can be considered as an adaption of U-Net for one dimensional time domain audio source separation.

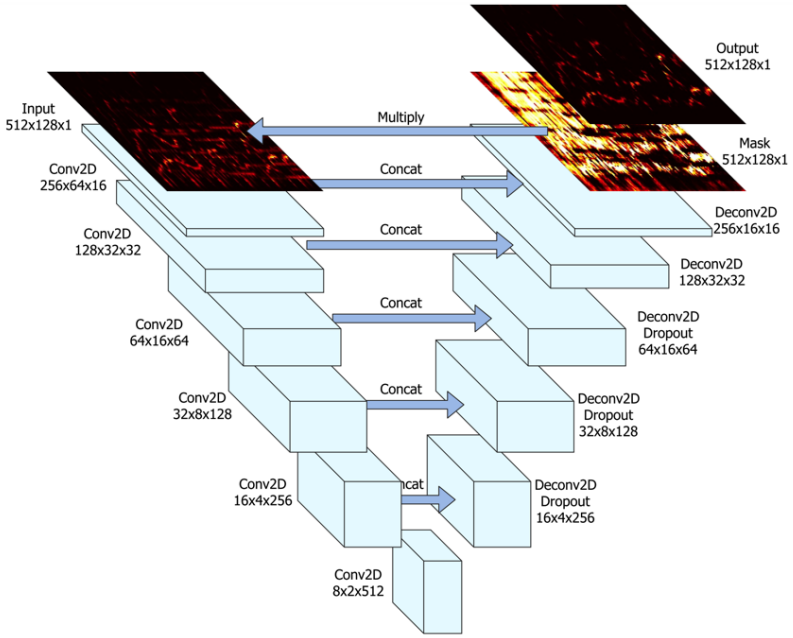


Figure 5.1: A common U-Net topology for source separation [89]

Recently, Wave-U-Net based architecture has been applied to speech and audio coding. Even with promising results, however, we would have to note that the definition of U-Net disqualifies it from being a neural codec. The reason is that the code is not only generated from the bottleneck code layer, but also identity shortcuts linking encoding blocks to the corresponding decoding blocks.

With the encoder and decoder being bridged, the size of the bottleneck can be deceptively small.

Instead of completely discarding U-Net, we plan to fix the identity shortcut issue to make the modualized network work for speech/audio coding, as the architecture has indeed shown impressive performance in multiple end-to-end acoustic signal processing tasks. Clearly these shortcuts are the major reason for the good performance, and can not be simply removed. We wonder if we could replace the shortcut with a neural quantizer. In that case, the shortcuts still exist, but are not identical anymore due to the quantization error. On the other hand, with the information originally transmitted via identical shortcuts being quantized, we may afford sending it to the decoder.

Besides, should the combination of Soft-to-hard quantization and vector quantization in VQ-VAE work, there is no reason to stop at just one-stage hash table lookup: it will be interesting to investigate if a multistaged hashing regime could even further lower the bitrate. Specifically, consider a dual-stage hashing case. As has been proved useful in information retrieval, with the output of the neural encoder (the index), we could fetch the vector from the first codebook, which could be the index of the second, even bigger codebook.

Tentative steps:

(a) Implement a Wave-U-Net for the end-to-end monaural speech enhancement;

  (1) Start from the available source code in TensorFlow. [1]

  (2) Train this Wave-U-Net on our 40-hour speech enhancement dataset used in [90].

  (3) Modify the topology such that the model output is just a denoised segment, such that it can be smoothly converted to a neural codec later.

(b) Convert the speech denoising network to a speech coding network;

  (1) Change the model input and output to the speech coding setting.

[1] https://github.com/frizzid07/Wave-U-Net

(2) Train a model which should deliver "near perfect" results due to those identical shortcuts.

(3) Try to remove the very bottom "code layer", as it may be irrelevant to the good performance. Should that be the case, we would only need to quantize those identical shortcuts.

(c) Ablation test on quantizing each of the identical shortcuts;

(1) From the far-end shortcut to the one near the bottleneck layer, sequentially replace the identical shortcut with the neural quantizer. Measure the performance degradation. I assume that replacing the far-end shortcut gives the most performance drop.

(2) Collect results from a quantized Wave-U-Net where all identical shorcuts are approximated by the neural quantizer with the same setting. From there, we will optimize both the model and the training scheme to improve the result.

(d) Potentially efforts towards better models and results.

(1) Employ bottleneck residual blocks or gated linear units to lower the model complexity.

(2) From scalar quantization to vector quantization: instead of reusing the VQ-VAE idea, we could re-define that vector quantization algorithm with the soft-to-hard technique, so that the quantization is compatible with backpropagation.

5.2 A μ-Law Conversed Loss Function For Speech Coding and Enhancement

For audio coding, we have proposed a more perceptually salient objective function the with psychoacoustical calibration scheme. It is doubtful if this can benefit a speech coding system. According to the psychoacoustic model, the masking curve is mainly from noise maskers, not tonal maskers. We are not sure if a similar amount of noise maskers can be detected from a speech signal. Besides, the lowest sample rate supported by current PAMs is 32kHz, much higher than 16kHz for even wide-band speech signals.

Table 5.1: Error rate comparison between linear and non-linear companding algorithms

samples	after linear companding	error	after non-linear companding	error
33	0	100%	32	3.0%
63	0	100%	62	1.6%
32120	32000	0.4%	31374	2.3%

Again, the idea is inspired from a prevalence companding technique in telecommunication, μ-law algorithm. It can be simply considered as a non-linear transformation for waveform digitization with the encoding process reducing the bit depth and the decoding process expanding the bit depth back. Long story short, μ-law algorithm focuses more at lower amplitudes where it "spreads out" the quantization intervals. The function μ-law compression and expansion is defined in Eq. 5.1.

$$\begin{aligned} \mu_{\text{compress}}(x) &= sign(x)\left(\frac{ln(1+\mu|x|)}{ln(1+\mu)}\right) \\ \mu_{\text{expand}}(x) &= sign(x)\left(\frac{(1+\mu)^{|x|}-1}{\mu}\right) \end{aligned} \tag{5.1}$$

Why is that important? Consider a bit-depth compression task to reduce 16-bit to 8-bit. The original bit depth setting gives a range between -32768 and 32767, which is to be narrowed to the range from -128 to 127. Should the companding algorithm be linear, both 33 and 63 with be zeroed out, with the error rate of 100%. In comparison, μ-law companding is capable of preserving samples even with very low amplitudes, although the error for large samples is slightly higher.

This is reflected in our neural codec: after optimization, most kernels fall into the range of low amplitudes. In other words, our model learns to place more quantization intervals for smaller samples to achieve better performance. But, if this is a known property, why not implementing it in the loss function, instead of resorting to the model to learn by itself? Therefore, we plan to make the mean squared error loss term aligned to μ-law function when measuring the reconstruction error in the time domain. After all, a 0.01 reconstruction error means much more to a sample with the amplitude of 0.1 than the one of 0.99.

Unlike the plan for the Wave-U-Net based neural codec, it is relatively straightforward to conduct a proof-of-concept study to understand if the μ-law based loss measurement is better

than the a plain mean squared error loss term. Aside from the μ-law based loss measurement, I could also try a new neuron convolutional operator with μ-law normalization. The idea is that μ-law companding can also be a non-linear transformation method: before being processed by 1-D convolution, the data can be pre-prosessed by μ-law compression algorithm to accent low-value samples; after convolution, the output is transformed back with μ-law expanding algorithm. To that end, we need to apply the continuous μ-law companding algorithm to the non-linear transformation.

Chapter 6

CONCLUSION

Bibliography

[1] P. Noll, "MPEG digital audio coding," *IEEE signal processing magazine*, vol. 14, no. 5, pp. 59–81, 1997.

[2] K. Brandenburg and G. Stoll, "ISO/MPEG-1 audio: A generic standard for coding of high-quality digital audio," *Journal of the Audio Engineering Society*, vol. 42, no. 10, pp. 780–792, 1994.

[3] K. R. Rao and J. J. Hwang, *Techniques and standards for image, video, and audio coding*, vol. 70. Prentice Hall New Jersey, 1996.

[4] D. O'Shaughnessy, "Linear predictive coding," *IEEE potentials*, vol. 7, no. 1, pp. 29–32, 1988.

[5] B. S. Atal and M. R. Schroeder, "Adaptive predictive coding of speech signals," *Bell System Technical Journal*, vol. 49, no. 8, pp. 1973–1986, 1970.

[6] M. Schroeder and B. S. Atal, "Code-excited linear prediction (CELP): High-quality speech at very low bit rates," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'85.*, vol. 10, pp. 937–940, IEEE, 1985.

[7] J. D. Gibson, "Speech coding methods, standards, and applications," *IEEE Circuits and Systems Magazine*, vol. 5, no. 4, pp. 30–49, 2005.

[8] D. Choudhary and A. Kumar, "Study and performance of amr codecs for gsm," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 3, no. 10, pp. 8105–8110, 2014.

[9] A. S. Spanias, "Speech coding: A tutorial review," *Proceedings of the IEEE*, vol. 82, no. 10, pp. 1541–1582, 1994.

[10] F. Itakura, "Early developments of LPC speech coding techniques," in *icslp*, pp. 1409–1410, 1990.

[11] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding," *Proceedings of the IEEE*, vol. 73, no. 11, pp. 1551–1588, 1985.

[12] M. Kim and P. Smaragdis, "Bitwise neural networks," in *International Conference on Machine Learning (ICML) Workshop on Resource-Efficient Machine Learning*, Jul 2015.

[13] L. Deng, M. L. Seltzer, D. Yu, A. Acero, A. R. Mohamed, and G. Hinton, "Binary coding of speech spectrograms using a deep auto-encoder," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

[14] M. Cernak, A. Lazaridis, A. Asaei, and P. N. Garner, "Composition of deep and spiking neural networks for very low bit rate speech coding," *arXiv preprint arXiv:1604.04383*, 2016.

[15] W. B. Kleijn, F. S. Lim, A. Luebs, J. Skoglund, F. Stimberg, Q. Wang, and T. C. Walters, "Wavenet based low rate speech coding," *arXiv preprint arXiv:1712.01120*, 2017.

[16] S. Kankanahalli, "End-to-end optimized speech coding with deep neural networks," *arXiv preprint arXiv:1710.09064*, 2017.

[17] L. J. Liu, Z. H. Ling, Y. Jiang, M. Zhou, and L. R. Dai, "Wavenet vocoder with limited training data for voice conversion," in *Proc. Interspeech*, pp. 1983–1987, 2018.

[18] Y. L. C. Garbacea, A. v. d. Oord, "Low bit-rate speech coding with vq-vae and a wavenet decoder," in *Proc. ICASSP*, 2019.

[19] B. Bessette, R. Salami, R. Lefebvre, M. Jelinek, J. Rotola-Pukkila, J. Vainio, H. Mikkola, and K. Jarvinen, "The adaptive multirate wideband speech codec (AMR-WB)," *IEEE transactions on speech and audio processing*, vol. 10, no. 8, pp. 620–636, 2002.

[20] J. M. Valin, K. Vos, and T. Terriberry, "Definition of the opus audio codec," *IETF, September*, 2012.

[21] J. M. Valin, G. Maxwell, T. B. Terriberry, and K. Vos, "High-quality, low-delay music coding in the opus codec," *arXiv preprint arXiv:1602.04845*, 2016.

[22] ISO/IEC 11172-3:1993, "Coding of moving pictures and associated audio for digital storage media at up to about 1.5 mbit/s," 1993.

[23] ITU-R Recommendation BS 1534-1, "Method for the subjective assessment of intermediate quality levels of coding systems (MUSHRA)," 2003.

[24] K. Tan, J. T. Chen, and D. L. Wang, "Gated residual networks with dilated convolutions for monaural speech enhancement," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 27, no. 1, pp. 189–198, 2018.

[25] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, pp. 1310–1318, 2013.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[27] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, and A. Graves, "Conditional image generation with pixelcnn decoders," in *Advances in neural information processing systems*, pp. 4790–4798, 2016.

[28] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 933–941, JMLR. org, 2017.

[29] W. Z. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. H. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1874–1883, 2016.

[30] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 6306–6315, 2017.

[31] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, and L. V. Gool, "Soft-to-hard vector quantization for end-to-end learning compressible representations," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 1141–1151, 2017.

[32] S. Kankanahalli, "End-to-end optimized speech coding with deep neural networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2018.

[33] K. Zhen, J. Sung, M. S. Lee, S. Beack, and M. Kim, "Cascaded cross-module residual learning towards lightweight end-to-end speech coding," in *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, 2019.

[34] R. Salami, C. Laflamme, B. Bessette, and J. P. Adoul, "ITU-T G. 729 annex a: reduced complexity 8 kb/s CS-ACELP codec for digital simultaneous voice and data," *IEEE Communications Magazine*, vol. 35, no. 9, pp. 56–63, 1997.

[35] Rec., "ITU-T G.722.2: Wideband coding of speech at around 16 kbit/s using adaptive multi-rate wideband (AMR-WB)," 2003.

[36] M. Neuendorf, M. Multrus, N. Rettelbach, G. Fuchs, J. Robilliard, J. Lecomte, S. Wilde, S. Bayer, S. Disch, C. Helmrich, *et al.*, "MPEG unified speech and audio coding-the iso/mpeg

standard for high-efficiency audio coding of all content types," in *Audio Engineering Society Convention 132*, Audio Engineering Society, 2012.

[37] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "Samplernn: An unconditional end-to-end neural audio generation model," *arXiv preprint arXiv:1612.07837*, 2016.

[38] Y. L. C. Garbacea, A. v. d. Oord, "High-quality speech coding with samplernn," in *Proc. ICASSP*, 2019.

[39] A. Gersho and V. Cuperman, "Vector quantization: A pattern-matching technique for speech coding," *IEEE Communications Magazine*, vol. 21, no. 9, pp. 15–21, 1983.

[40] A. Gersho and R. M. Gray, *Vector quantization and signal compression*, vol. 159. Springer Science & Business Media, 2012.

[41] B. Bhattacharya, W. LeBlanc, S. Mahmoud, and V. Cuperman, "Tree searched multi-stage vector quantization of lpc parameters for 4 kb/s speech coding," in *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 105–108, IEEE, 1992.

[42] K. M. He, X. Y. Zhang, S. Q. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[43] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, and L. V. Gool, "Soft-to-hard vector quantization for end-to-end learning compressible representations," in *Advances in Neural Information Processing Systems*, pp. 1141–1151, 2017.

[44] K. Tan, J. T. Chen, and D. L. Wang, "Gated residual networks with dilated convolutions for supervised speech separation," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 21–25, IEEE, 2018.

[45] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, no. 6, pp. 520–541, 1987.

[46] L. R. Welch and E. R. Berlekamp, "Error correction for algebraic block codes," Dec. 30 1986. US Patent 4,633,470.

[47] R. BS, "ITU-R 1534-1,"method for the subjective assessment of intermediate quality levels of coding systems (MUSHRA)"," *International Telecommunication Union*, 2003.

[48] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, N. L. Dahlgren, and V. Zue, "TIMIT acoustic-phonetic continuous speech corpus," *Linguistic Data Consortium, Philadelphia*, 1993.

[49] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[50] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, vol. 2, pp. 749–752, IEEE, 2001.

[51] K. Brandenburg and G. Stoll, "ISO/MPEG-1 Audio: A generic standard for coding of high-quality digital audio," *Journal of the Audio Engineering Society*, vol. 42, no. 10, pp. 780–792, 1994.

[52] M. Hasegawa-Johnson and A. Alwan, "Speech coding: Fundamentals and applications," *Wiley encyclopedia of telecommunications*, 2003.

[53] D. O'Shaughnessy, "Linear predictive coding," *IEEE potentials*, vol. 7, no. 1, pp. 29–32, 1988.

[54] T. Moriya, R. Sugiura, Y. Kamamoto, H. Kameoka, and N. Harada, "Progress in lpc-based frequency-domain audio coding," *APSIPA Transactions on Signal and Information Processing*, vol. 5, 2016.

[55] J. M. Valin, G. Maxwell, T. B. Terriberry, and K. Vos, "High-quality, low-delay music coding in the opus codec," *arXiv preprint arXiv:1602.04845*, 2016.

[56] J. M. Valin, "Speex: A free codec for free speech," *arXiv preprint arXiv:1602.08668*, 2016.

[57] ITU-T G.722.2:, "Wideband coding of speech at around 16 kbit/s using adaptive multi-rate wideband (AMR-WB)," 2003.

[58] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[59] W. B. Kleijn, F. S. C. Lim, A. Luebs, J. Skoglund, F. Stimberg, Q. Wang, and T. C. Walters, "WaveNet based low rate speech coding," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 676–680, 2018.

[60] Y. L. C. Garbacea, A. van den Oord, "Low bit-rate speech coding with VQ-VAE and a wavenet decoder," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2019.

[61] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. van den Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," *arXiv preprint arXiv:1802.08435*, 2018.

[62] J.-M. Valin and J. Skoglund, "LPCNet: Improving neural speech synthesis through linear prediction," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2019.

[63] J. M. Valin and J. Skoglund, "A real-time wideband neural vocoder at 1.6 kb/s using lpcnet," *arXiv preprint arXiv:1903.12087*, 2019.

[64] B. Bessette, R. Salami, R. Lefebvre, M. Jelinek, J. Rotola-Pukkila, J. Vainio, H. Mikkola, and K. Jarvinen, "The adaptive multirate wideband speech codec (AMR-WB)," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 8, pp. 620–636, 2002.

[65] F. Soong and B. Juang, "Line spectrum pair (lsp) and speech data compression," in *ICASSP'84. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 9, pp. 37–40, IEEE, 1984.

[66] P. Cummiskey, N. S. Jayant, and J. L. Flanagan, "Adaptive quantization in differential pcm coding of speech," *Bell System Technical Journal*, vol. 52, no. 7, pp. 1105–1118, 1973.

[67] K. K. Paliwal and B. S. Atal, "Vector quantization of lpc parameters," *Speech and Audio Coding for Wireless and Network Applications*, vol. 224, p. 191, 2012.

[68] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[69] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in *in Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, pp. 749–752, IEEE, 2001.

[70] J. G. Beerends, C. Schmidmer, J. Berger, M. Obermann, R. Ullmann, J. Pomy, and M. Keyhl, "Perceptual objective listening quality assessment (POLQA), the third generation ITU-T stan-

dard for end-to-end speech quality measurement part i—temporal alignment," *Journal of the Audio Engineering Society*, vol. 61, no. 6, pp. 366–384, 2013.

[71] M. Bosi, K. Brandenburg, S. Quackenbush, L. Fielder, K. Akagiri, H. Fuchs, and M. Dietz, "ISO/IEC MPEG-2 advanced audio coding," *Journal of the Audio Engineering Society*, vol. 45, no. 10, pp. 789–814, 1997.

[72] K. Brandenburg and G. Stoll, "ISO/MPEG-1 audio: a generic standard for coding of high-quality digital audio," *Journal of the Audio Engineering Society*, vol. 42, no. 10, pp. 780–792, 1994.

[73] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "An experimental study on speech enhancement based on deep neural networks," *IEEE Signal Processing Letters*, vol. 21, no. 1, pp. 65–68, 2014.

[74] J. Engel, C. Resnick, A. Roberts, S. Dieleman, M. Norouzi, D. Eck, and K. Simonyan, "Neural audio synthesis of musical notes with wavenet autoencoders," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1068–1077, JMLR. org, 2017.

[75] K. Zhen, M. S. Lee, and M. Kim, "Efficient context aggregation for end-to-end speech enhancement using a densely connected convolutional and recurrent network," *arXiv preprint arXiv:1908.06468*, 2019.

[76] J. Klejsa, P. Hedelin, C. Zhou, R. Fejgin, and L. Villemoes, "High-quality speech coding with SampleRNN," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2019.

[77] D. Stoller, S. Ewert, and S. Dixon, "Wave-u-net: A multi-scale neural network for end-to-end audio source separation," in *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, 2018.

[78] Q. J. Liu, W. W. Wang, P. J. Jackson, and Y. Tang, "A perceptually-weighted deep neural network for monaural speech enhancement in various background noise conditions," in *2017 25th European Signal Processing Conference (EUSIPCO)*, pp. 1270–1274, IEEE, 2017.

[79] Z. Chen, Y. Luo, and N. Mesgarani, "Deep attractor network for single-microphone speaker separation," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pp. 246–250, IEEE, 2017.

[80] K. Zhen, A. Sivaraman, J. M. Sung, and M. Kim, "On psychoacoustically weighted cost functions towards resource-efficient deep neural networks for speech denoising," *arXiv preprint arXiv:1801.09774*, 2018.

[81] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "A short-time objective intelligibility measure for time-frequency weighted noisy speech," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2010.

[82] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 2, pp. 749–752, 2001.

[83] S. W. Fu, T. W. Wang, Y. Tsao, X. G. Lu, and H. Kawai, "End-to-end waveform utterance enhancement for direct evaluation metrics optimization by fully convolutional neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 9, pp. 1570–1584, 2018.

[84] J. M. Martín-Doñas, A. M. Gomez, J. A. Gonzalez, and A. M. Peinado, "A deep learning loss function based on the perceptual evaluation of the speech quality," *IEEE Signal processing letters*, vol. 25, no. 11, pp. 1680–1684, 2018.

[85] D. Salomon, *Data compression: the complete reference*. Springer Science & Business Media, 2004.

[86] C. H. Yen, Y. S. Lin, and B. F. Wu, "A low-complexity mp3 algorithm that uses a new rate control and a fast dequantization," *IEEE Transactions on Consumer Electronics*, vol. 51, no. 2, pp. 571–579, 2005.

[87] S. Zamani and K. Rose, "Spatial audio coding without recourse to background signal compression," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 720–724, IEEE, 2019.

[88] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015.

[89] A. Jansson, E. Humphrey, N. Montecchio, R. Bittner, A. Kumar, and T. Weyde, "Singing voice separation with deep u-net convolutional networks," 2017.

[90] K. Zhen, M. S. Lee, and K. M, "A dual-staged context aggregation method towards efficient end-to-end speech enhancement," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020.

CURRICULUM VITAE

Name: Kai Zhen
Affiliation: School of Informatics Computing and Engineering, Cognitive Science Program
Email: zhenk@iu.edu
Website: `http://kaizhen.us`

Education

Indiana University, Bloomington, IN, USA

- Ph.D., Computer Science and Cognitive Science

Tsinghua University, Beijing, China

- M.S., Computer Science, July, 2015

Xidian University, Xi'an, ShaanXi, China

- B.S., Software Engineering, July, 2012

Industrial Experience

Amazon, Inc.

- Applied Scientist Intern, Alexa Edge ML Team, Pittsburgh, PA 2020 Summer

LinkedIn Corporation

- Machine Learning & Relevance Intern, Mountain View, CA, 2019 Summer
- Machine Learning & Relevance Intern, New York City, NY, 2018 Summer

Peer-reviewed Publications & Patents

1. **Kai Zhen**, Mi Suk Lee, Jongmo Sung, Seungkwon Beack, and Minje Kim, "Efficient And Scalable Neural Residual Waveform Coding with Collaborative Quantization," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Barcelona, Spain, May 4-8, 2020.*
2. **Kai Zhen**, Mi Suk Lee, Minje Kim. "A Dual-Staged Context Aggregation Method towards Efficient End-To-End Speech Enhancement," *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Barcelona, Spain, May 4-8, 2020.*
3. **Kai Zhen**, Jongmo Sung, Mi Suk Lee, Seungkwon Beack, and Minje Kim, "Cascaded Cross-Module Residual Learning towards Lightweight End-to-End Speech Coding," *Annual Conference of the International Speech Communication Association (Interspeech), Graz, Austria, September 15-19, 2019.*
4. Sung, Jongmo, Minje Kim, Aswin Sivaraman, and **Kai Zhen**. "Audio signal encoding method and apparatus and audio signal decoding method and apparatus using psychoacoustic-based weighted error function." *U.S. Patent Application 16/122,708, filed May 30, 2019.*