NEURAL WAVEFORM CODING: SCALABILITY, EFFICIENCY AND PSYCHOACOUSTIC CALIBRATION

Kai Zhen

Submitted to the faculty of the University Graduate School

in partial fulfillment of the requirements for the degree

Doctor of Philosophy

in the Luddy School of Informatics Computing and Engineering

and the Cognitive Science Program,

Indiana University

May 2021

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Doctoral Committee

Minje Kim, Ph.D.

Robert Goldstone, Ph.D.

Donald S. Williamson, Ph.D.

Yi Shen, Ph.D.

Date of Defense: 04/02/2021

To my parents ZHEN Jianwei and LIU Zhaoqing;

my grandparents ZHEN Jide, WU Sanhua, LIU Wanji and LIU Lijuan.

献给我的父亲甄建伟，母亲刘兆清；祖父甄纪德，祖母武三华，外祖父刘万纪，外祖母刘丽娟。

ACKNOWLEDGEMENTS

It has been 6 years since I embarked on my Ph.D. journey. When I look back, I do not feel that time flies but each day counts. To some people, this may indicate a long and arduous experience with misery and suffering. However, in my case it becomes evidence of a solid Ph.D experience which would not be possible without the great support from many great people who have accompanied me through the path.

To begin with, I would like to show my appreciation for my advisor Prof. Minje Kim for his dedication and patience so as to transform me from a junior graduate student to a student researcher. I took both his courses on signal processing and deep learning, which helps building my knowledge base for the following research projects. I joined the lab approximately the same time as he was gifted a daughter, and I feel that it somehow enables some extra care from my advisor on me as a consequence: he would allow junior graduate students like me to make mistakes while keeping feeding us great ideas and challenging us the way to formulate a problem, propose a novel method and validate the effectiveness of it. I am particularly thankful for him being transparent with his students. It helps me iron out the wrinkles in time.

In addition, I also want to thank my dissertation committee members. Among them, Prof. Robert Goldstone informed me of the opportunity of pursuing this combined Ph.D. program, and funded me for the very first month before I was hired as an associate instructor. As an inter-disciplinary program, it offers me with a diverse perspective on the core inquiry of the nature of minds. Although a general-purpose human-like model with a cross-field consensus may still be elusive, being exposed to cognition related theories, even from a distinct discipline, overall benefits my engineering-oriented work as I have been encouraged to address a computational problem from an alternative human learning perspective, which will also benefit me in future days to come. Prof. Donald Williamson has been on my committee since the qualifying exam. He is very supportive of

me writing a journal manuscript even though the turnaround time can be much longer than that of a conference these days. I also feel thankful to have Prof. Shen Yi on my committee especially with his expertise on understanding the mechanism that counts for the difficulty of individuals with hearing loss.

I have also been inspired by many faculties through the program. Prof. David Crandall's course on the introduction of artificial intelligence (AI) is particularly informative and well structured, where I gained hands-on practice on a wide range of methodologies. I still remember his lectures on two declines of AI in the 1960s and late 1990s, respectively, when neural networks simply failed to deliver what could keep people excited about. Gee, if only the third one is not on its way. In addition, I enjoyed Prof. Chris Raphael's data mining course as he would draft highly mathematically symbolized notes every time before offering the lecture. He insisted on having a written exam in my qualifying exam. The preparation for it turned out to be a memorable period of time, in retrospect, for me to stay focused on revisiting the fundamentals of data science, which would not be possible without a requirement as such.

It has been quite a pleasure to work with my smart and diligent colleagues in SAIGE and great people from the industry. During the past 4 years in SAIGE where I have worked as a research assistant, I have got to know Sanna Wager, a well-trained musician and researcher, Sunwoo Kim who is always humble and helpful, Haici Yang who recently joined us as a specially intelligent and optimistic teammate, and of course the versatile Aswin. I wish I could get further familiarized with David Badger with his experience in the army and Darius Petermann recently graduated from Berklee College of Music. Aside from the daily routine of conducting research at Indiana University, my past internships have certainly added color to my life. When reflecting on those summers, I have always been grateful for Xiaoqiang Luo, the Head of Machine Learning and AI in LinkedIn NYC Office, who picked me to be a machine learning intern in his team back in 2018. That is my first internship which ignites my passion for pushing the envelope in algorithms in Automatic

Speech Recognition (ASR), Natural Language Understanding (NLU), and Audio Signal Processing. My intern mentors Haoran Wang (with company standardization group), Lijun Peng (with ads AI group) and Hieu Duy Nguyen (with Alexa machine learning group) have been knowledgeable enough to keep me involved and well developed in the team during 3 wonderful summers.

I feel blessed to have my graduate research experience in Indiana University, Bloomington. This gorgeous, seasonal and lively college town literally features a combination of everything. The hospitable staff in Indiana Memorial Union (IMU), Student Recreational Sports Center (SRSC), the Office of International Services (OIS) and Luddy Hall will always be remembered. I wish my visit to IU could be longer so that I could establish more collaborations and accumulate a longer publication list. But life is short, and it is time to move on.

Thus far, it is still uncertain if I could further navigate my future career path in the United States. The world is changing fast as what we are struggling with, or suffering from, would seem rather unreal just a few years earlier. No matter how, what is in the future simply cannot overwrite what has happened: the timeless memory on the shell island in Panama City Beach with my roommate Karl Beck and his family, the frontier fun in the Yellowstone National Forest, the first Super Bowl Party hosted by Ed Huff and Elan Rajamani, the first internship in the iconic Empire State Building... They are all generous gifts offered while I am here in this country, and becoming inseparable parts of my life!

Kai Zhen

NEURAL WAVEFORM CODING: SCALABILITY, EFFICIENCY AND PSYCHOACOUSTIC CALIBRATION

Acoustic signal compression techniques, converting floating-point waveforms into a bitstream representation, serve a cornerstone 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 aids which do not afford a gigantic neural network in spite of the impressive performance.

We propose a research methodology to not simply discard conventional DSP methods by embracing 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. 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. In summary, the thesis presents a hybrid set of techniques incorporating conventional and domain specific methods into the modern deep learning system, which facilitates an efficient yet powerful solution for speech and audio waveform compression.

Minje Kim, Ph.D.

Robert Goldstone, Ph.D.

Donald S. Williamson, Ph.D.

Yi Shen, Ph.D.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 Thesis Overview

Acoustic waveform coding, where the encoder converts the acoustic signal into bitstreams and the decoder synthesizes reconstructed signal from received bitstreams, serves an important role for various purposes: to secure a voice communication [2][3], to facilitate data transmission [4], 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 kilobits 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. Traditionally, 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) [5], adaptive encoding [6], and perceptual weighting [7]. Stemmed from those coding techniques, Speex, AMR-WB and Opus are some of the industrialized speech codecs. AMR-WB consists of multiple steps including 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, just to name a few.

The design of speech codecs is to address the trade-off among low bitrate, high perceptual quality, low complexity and delay, etc [8, 9]. 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 [10]. Vocoders use few parameters to model the human speech production process, such as vocal tract, pitch frequency,

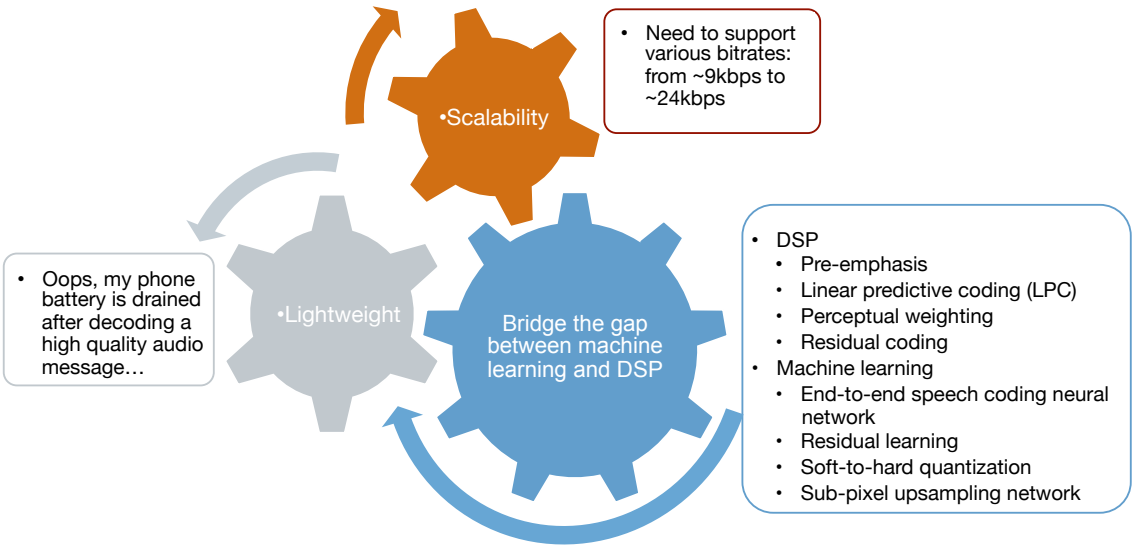


Figure 1.1: Tradeoffs in the design of speech/audio codecs

etc [11], 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 sound as similar to the reference 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 shown 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) [12] or bitwise network techniques [13], etc. Many DNN methods [14][15] 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 [16][17][18][19] model speech signals in time domain directly without T-F transformation. They are referred to as end-to-end methods, yielding competitive performance compared with current speech coding standards, such as AMR-WB [20].

However, a straightforward data-driven approach as such, even comparable or superior to those classical counterparts, may actually be far from realistic to be implemented. 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 [19] outperforms other low bitrate codecs in the subjective listening test, yet with 20 millions parameters, beyond what a resource-constrained device can afford for real-time processing.

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 in industrial products; second, it enables the codec to not only operate at relatively low bitrate cases with decent quality, but also high bitrates with near transparent quality. Overall, it could be seamlessly incorporated with other artificial intelligence tasks that have already heavily relied on deep neural networks. It is not hard to underestimate the role DNNs have played in speech enhancement, recognition and diarization, to name a few, for natural language understanding, where modules are tuned systematically to maximize the performance. Should the speech/audio codec be implemented as a deterministic (non-trainable) algorithm purely based on DSP, it is not possible to be collaboratively tuned along with other neural network components. Therefore, a compact neural waveform codec compatible to low-power devices is a first step towards future intelligent systems for acoustic signal processing.

In this thesis, we propose a cascaded cross-module residual learning (CMRL) pipeline by en-

abling the conventional multi-staged residual coding concept in deep neural network platform for speech and audio coding. In addition, we combine linear predictive coding (LPC) with CMRL and design 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. Aside from the model topology, as another training component, the loss function is critical in supervised learning to navigate the model towards desired behaviors: to that end, we propose a perceptual loss with psychoacoustic calibration, baking human auditory perception into the training of a neural network for audio coding.

In the remainder of the introductory chapter, we firstly review the related conventional techniques for speech and audio compression. Decades-old are many conventional codecs, they are by no means readily to be ditched. In fact, in many rural areas with a relatively limited Internet bandwidth budget, conventional codecs, due to their runtime efficiency, are still highly needed. Therefore, hybrid designs are oftentimes found even from recently proposed codecs as a combination of conventional techniques and contemporary paradigms with deep neural networks. We then provide an overview of related deep learning methods which are building blocks of the proposed model in this thesis. More importantly, we motivate the methodology of designing the neural waveform codec from a cognitive science perspective, by highlighting the role of predictive coding in lightweight and scalable neural codecs, in addition to how human auditory perception can be leveraged in neural audio coding. Finally, we summarize the motivation and outline the topic for each following chapter, respectively.

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.

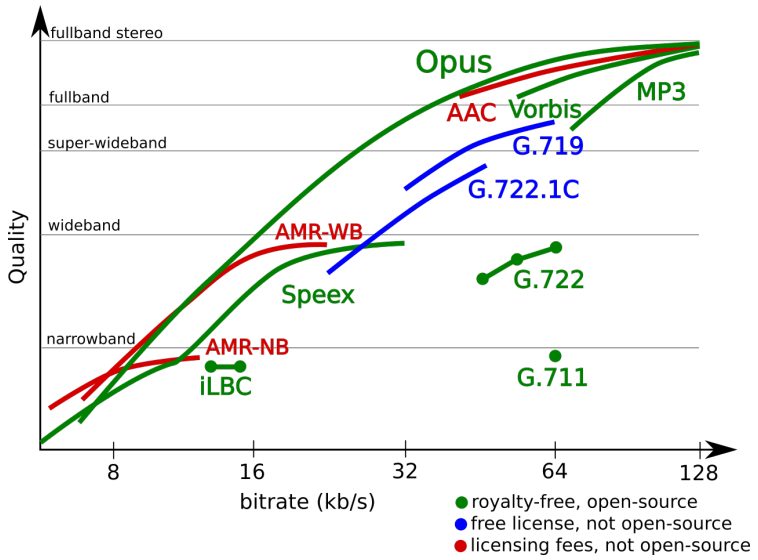


Figure 1.2: The codec comparison on performance, bitrate, and bandwidth [1]

We introduce three speech/audio codecs which have been compared to our proposed systems, AMR-WB [20], Opus [21], and MPEG Audio Layer III (MP3) [22]. Adaptive Multi-Rate Wideband, 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 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 Figure 1.2, the speech quality from Opus at very low bitrate modes is less competitive.

1.2.2 Neural vocoders

The neural vocoder is not the neural network counterpart for classical signal processing based codecs, which should actually be neural waveform codec. Neural vocoders are rather decoders that synthesize the speech only waveform from a few acoustic features, such as mel frequency cepstral coefficients (MFCC), bark frequency cepstral coefficients (BFCC), the fundamental frequency [23]. Neural vocoders are proposed mainly for text-to-speech (TTS), such as WaveNet [24] which can synthesize speech signals of good quality but runs slow. To tackle this issue, WaveRNN [25] is proposed for efficient neural speech synthesis. It only has a single-layer recurrent layer with a dual softmax layer that folds a long sequence into smaller batches so as to generate multiple samples per time. Consequently, WaveRNN runs much faster with comparable speech quality. LPCNet is a variant of WaveRNN by incorporating linear predictive coding (LPC) [5] to sample generation which operates in real time with a bitrate of only 1.6 kbps [26].

1.2.3 Evaluation metrics

1.2.3.1 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). Short-time objective intelligibility (STOI) [27] 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) [28]. 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.

1.2.3.2 Scores from subjective listening tests

Note that the evaluation which is only evidenced by objective measures may not be sufficient to justify the usage of the model for real-world applications, as the discrepancy between these objective scores and mean opinion score (the true reflection of human auditory perception) can sometimes be very noticeable [29]. Therefore, it is highly recommended to report MOS acquired from subjective listening tests. 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 tests 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 tests 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 (who are trained according to [30] and committed to taking the tests).

In our research, we collect subjective listening scores based on MUltiple Stimuli with Hidden Reference and Anchor (MUSHRA) [30] 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. Figure 1.3 shows the interface of a MUSHRA trial.

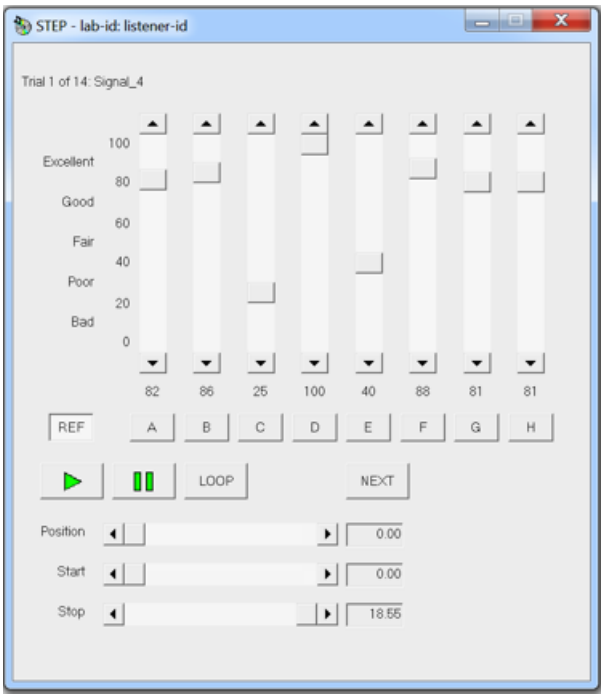


Figure 1.3: The interface for a MUSHRA trial

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;
- 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 neural network techniques from non-linear transformation to trainable quantization, which are building blocks to our proposed systems.

1.3.1 Dilated 1-D convolution

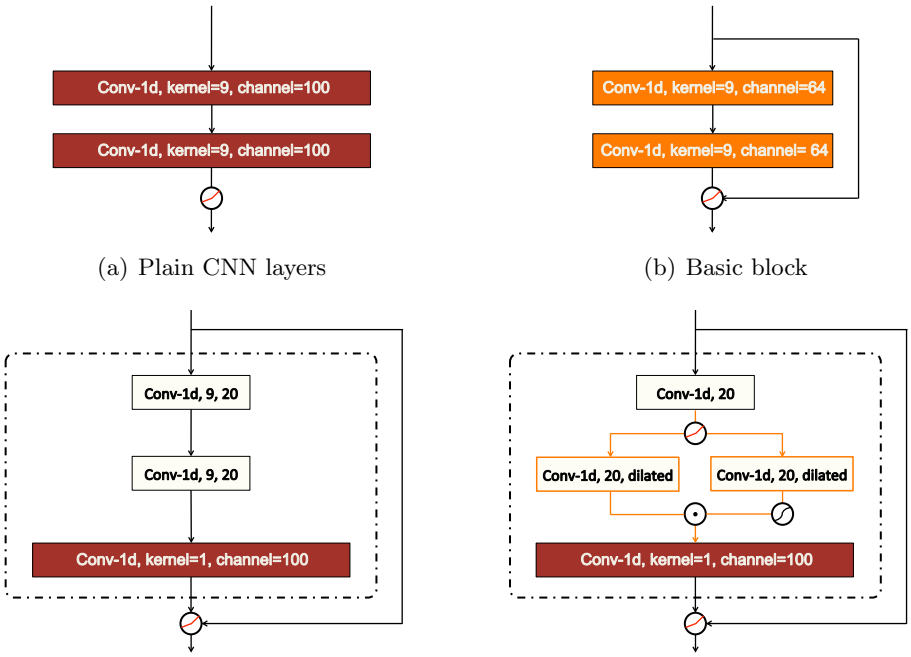
As the input, for end-to-end acoustic signal processing, is the waveform segment, it suffices to use 1-dimension (1-D) convolution. Given the convolution operator denoted as $*$, the 1-D convolution evaluated at p on the signal $\boldsymbol{i}$ with the kernel $\boldsymbol{k}$ is formally defined in equation 1.1, where γ denotes the dilation rate with the default value of 1. For instance, if the kernel size is 3, to convolve $\boldsymbol{i}$ by $\boldsymbol{k}$ at $p = 0$, we calculate $i(-1)k(1) + i(0)k(0) + i(1)k(-1)$. CNNs employ dilated layers to enlarge the receptive field, aggregating contextual information without increasing the kernel size [31].

$$(\boldsymbol{i} *_{\gamma} \boldsymbol{k})(p) = \sum_{s+\gamma t=p} \boldsymbol{i}(s)\boldsymbol{k}(t) \tag{1.1}$$

1.3.1.1 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 [32]. The core idea of residual learning is to add identical shortcuts (Figure 1.4 (b)), such that the layers within a residual learning block will only need to learn the difference (or residual) between the input and output of the block.

It is critical to make residual learning building blocks efficient, as they are repeatedly used in various advanced CNN architectures. Bottleneck residual learning blocks [33] usually replace $\left[\begin{smallmatrix} 9,\ 64 \\ 9,\ 64 \end{smallmatrix}\right]$ type of kernel setting (Figure 1.4 (b)) to $\left[\begin{smallmatrix} 9,\ 20 \\ 9,\ 20 \\ 1,\ 100 \end{smallmatrix}\right]$ (Figure 1.4 (c)). Note that not only the amount of parameters for each block is reduced via the bottleneck design, the dimension of the feature map is increased (from 64 to 100), which is usually found to benefit the overall performance.



(a) Plain CNN layers

(b) Basic block

(c) Bottleneck block

(d) Gated linear unit

Figure 1.4: CNN building blocks

As a variation of the bottleneck block, gated linear unit ([34]) is proposed to boost the gradient flow so as to leverage longer term temporal information. Note that in Figure 1.4 (d), left-hand side component ($\boldsymbol{v}_1$) is without any activation function for the Hamdard product with $\boldsymbol{v}_2$ from the right-hand side (equation 1.2). Consequently, there is no downscaling factor from the left-hand side component in the gradient (equation 1.3). It differs from the long short-term memory (LSTM) style gating scheme in [35] where the tanh function is employed as the activation function for $\boldsymbol{v}_1$ (equation 1.4). In contrast, the LSTM-style gating is subject to gradient vanishing due to downscaling factors in its gradient (equation 1.5).

$$\boldsymbol{y} = \boldsymbol{v}_1 \odot \sigma(\boldsymbol{v}_2) \tag{1.2}$$

$$\nabla[\boldsymbol{v}_1 \odot \sigma(\boldsymbol{v}_2)] = \nabla\boldsymbol{v}_1 \odot \sigma(\boldsymbol{v}_2) + \sigma'(\boldsymbol{v}_2)\nabla\boldsymbol{v}_2 \odot \boldsymbol{v}_1 \tag{1.3}$$

$$\boldsymbol{y} = \tanh(\boldsymbol{v}_1) \odot \sigma(\boldsymbol{v}_2) \tag{1.4}$$

$$\nabla[\tanh(\boldsymbol{v}_1) \odot \sigma(\boldsymbol{v}_2)] = \tanh'(\boldsymbol{v}_1)\nabla\boldsymbol{v}_1 \odot \sigma(\boldsymbol{v}_2) + \sigma'(\boldsymbol{v}_2)\nabla\boldsymbol{v}_2 \odot \tanh(\boldsymbol{v}_1) \tag{1.5}$$

1.3.1.2 Sub-pixel upsampling convolution

Upsampling convolution is essential for our neural codecs as it recovers the data rate, after downsampling with strided convolutions, back to the original one of the input signal. Sub-pixel upsampling is proposed by [36] for super resolution images. Unlike deconvolution, 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 upsampling 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 Figure 1.5. Note that the feature map is learnable during backpropagation, although these feature map transformations are deterministic.

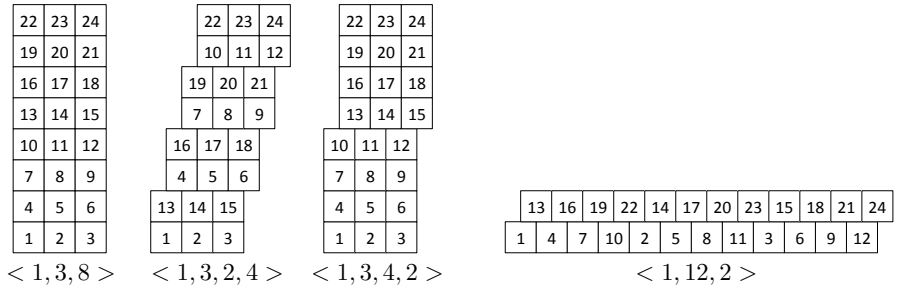


Figure 1.5: An example of 4X sub-pixel upsampling

1.3.1.3 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

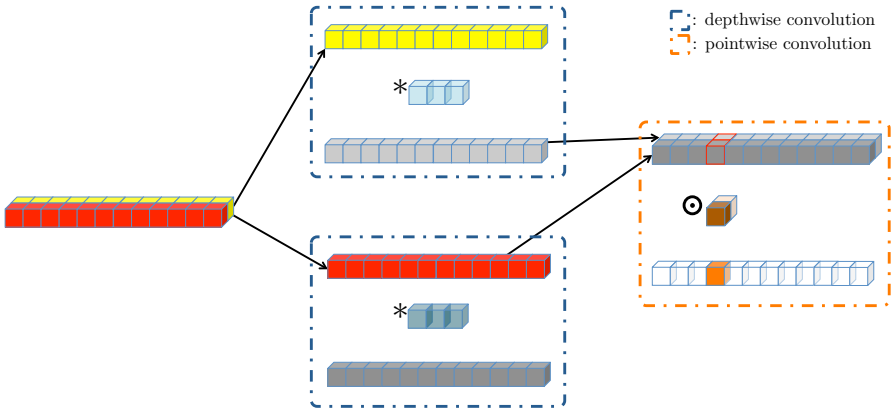


Figure 1.6: An example of 4X upsampling

the last feature map in Figure 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 (as the number of input and output channels will be 2). However, if we decompose this convolution into two steps, by conducting time domain convolution and depthwise convolution separately, we only need $3 \times 2 \times 1 + 1 \times 2 \times 2 = 10$ parameters. This technique is referred to as "depthwise separable convolution". The toy example is described in Figure 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 $k_s \times d_1 \times d_2$ parameters; for depthwise separable convolution, the number can be lowered to $k_s \times 1 \times d_1 + 1 \times d_1 \times d_2$. Note that in a more realistic case when the kernel size is reasonably large, the benefit of this technique in reducing the amount of parameters can be noticeable [37].

1.3.2 Neural discrete representation learning

1.3.2.1 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 asks for 0.5 MB of memory space. Empirical results show that BNN performs well for speech denoising and handwritten digit recognition at minimal computa-

tional cost [13].

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 activation function as in $\boldsymbol{y} = \tanh(\mathbb{W}\boldsymbol{x} + b)$, where $\mathbb{W}$ is the model weight of a network layer, $\boldsymbol{x}$ is the input of that layer and and b is the bias. During the feedforward step, the tanh function is replaced by *sign* function, such that the output can only be either -1 or $+1$. Note that sign function is non-differentiable. To make it compatible with backpropagation during model training, the derivative of the sign function is approximated by that of a tanh function.

The downside of BNN as a discrete representation learning technique is that it is difficult to be coupled with entropy coding. The amount of unique binary vectors increases exponentially, in that the binary vector with the length of l has 2^l permutations. Without a well-defined regularizer on the distribution of these vectors in the high dimensional space, they are expected to be sparsely located, which is not favored by entropy coding procedures.

1.3.2.2 Vector quantized-variational autoencoder (VQ-VAE)

Following the turbulent wave of deep generative models, two winning candidates are arguably to be generative adversarial networks (GAN) [38] and variational autoencoders (VAE) [39]. GAN features a unique adversarial training scheme between two components, where a generator is optimized to create sufficiently "vivid" samples so as to fool a discriminator while the discriminator is optimized to tell the authentic samples from those counterfeit ones. While GAN is empirically found to produce better "synthesized" images, VAE is deemed more relevant to the scope of our work on compressing the waveform, as it focuses on learning the compact representation of the input data. The learned representation can be further discretized via vector quantization (VQ) as in VQ-VAE [40].

As indicated by its name, VQ-VAE has two major components. First, it's a VAE which does not directly learn a compressed representation of the input data but a set of parameters of a Gaussian distribution representing the data in the latent space. In other words, the decoder of a VAE does not consume what the encoder outputs but samples from a Gaussian distribution which is parameterized by the encoder's output. Because of that, VAE differs from the conventional autoencoder, which targets a perfect input reconstruction, in that VAE is capable of synthesizing the input, or generating similar but unseen samples. The other component is the built-in vector quantization operator. The samples from the Gaussian distribution are segmented into frames. Each frame is represented by the corresponding embedding with the highest similarity in the codebook. Suppose the codebook contains 1024 embeddings, it is sufficient to use 10 bits to index all embeddings. Rather than quantizing the samples from the distribution, the encoder simply sends indices of embeddings in a bitstream. The decoder uses the received indices to fetch the actual samples to assemble its inputs.

VQ-VAE has been widely adopted in recent neural speech coding schemes as it can downsample the feature map by up to 64 $\times$ and still deliver decent performance. Note that due to its "generative" nature, the "synthesized" speech does not preserve the F0-related perceptual factors (including pitch) of the speaker. The synthesized speech loses the original structure of the waveform, but still sounds similar. One way to compensate for it is to transmit the fundamental frequency of the speech as auxiliary information to facilitate a more accurate pitch reconstruction [19].

Although VQ-VAE serves an effective neural discrete representation learning scheme, it does not suffice to meet an efficient and general-purpose waveform coding criteria in this thesis. First of all, the model size of VQ-VAE can significantly go beyond what a low power device can afford. What exacerbates the computational overhead is when VQ-VAE is used along with WaveNet, another large auto-regressive neural network. In that case, WaveNet functions as a decoder consuming the code generated from VQ-VAE to synthesize the speech signal. Moreover, VQ-VAE does not fit in

well with audio coding. While synthesized speech can be tolerant even with an altered prosody or accent, the audio signal of an opera performance is expected to be rendered of high fidelity.

1.3.2.3 Soft-to-hard quantization

To compress acoustic waveforms, a core component of an autoencoder (AE) is the trainable quantizer which learns a discrete representation of the code layer in the AE. An alternative of VQ-VAE [40] is termed as soft-to-hard quantization [41] which is employed in the many end-to-end speech coding AEs [42, 43]. 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 of which 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 $\boldsymbol{D} \in \mathbb{R}^{I \times J}$.

$$\boldsymbol{A}_{\text{hard}}(i,j) = \begin{cases} 1 & \text{if } \boldsymbol{D}(i,j) = \min_{j'} \boldsymbol{D}(i,j') \\ 0 & \text{otherwise} \end{cases} . \tag{1.6}$$

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 $\bar{\boldsymbol{h}}$.

Figure 1.7 summarizes the softmax quantization process.

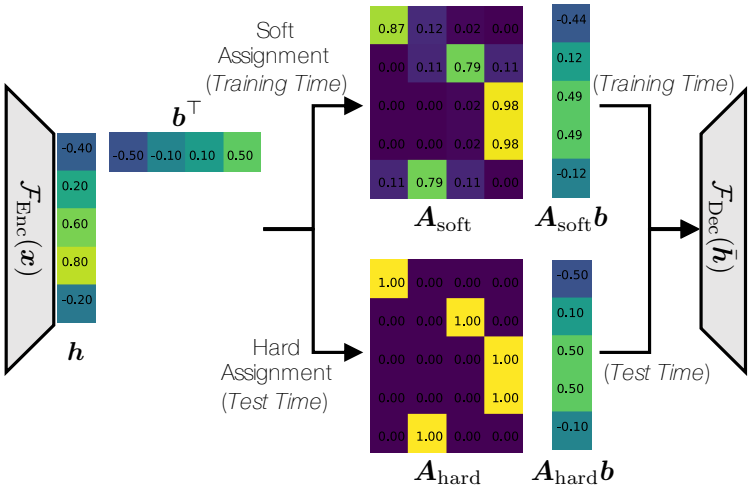


Figure 1.7: An example of the softmax quantization process.

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 inherent temporal structure in waveforms, which could be modeled in vector quantization. However, empirically, we could not implement a vector based soft-to-hard quantization.

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

1.3.2.4 WaveNet

WaveNet [24] was not originally proposed to compress audio signals but to implement an auto-regressive model to generate audio waves as an audio synthesis tool. It has been successfully employed in text-to-speech (TTS), a core functionality in various industrial products such as Google Assistant, Amazon Alexa, Apple's Siri and Microsoft's Cortana.

The major merit to use a WaveNet for speech coding is that it narrows the quality gap between speech synthesis systems and waveform coding systems [44]. Conventionally, a vocoder such as Codec 2 [45] does not generate satisfying decoded speech signals as it operates at a relatively low bitrate. WaveNet vocoder is able to achieve comparable performance with standardized waveform codecs such as AMR-WB [20], although WaveNet consumes much fewer bits per second.

The potential drawbacks for a WaveNet codec is twofold. As mentioned previously, a WaveNet code, like most other neural network codecs, is computationally expensive which limits its potential on devices. In addition, it runs strikingly slow. A WaveNet decoder may take more than 1 minute to decode 1 second of speech, which is far from realistic for real-time communication. One obvious reason is that it is based on an auto-regressive model: for each prediction step, the model only predicts one single sample.

1.4 Motivation from a Cognitive Science Perspective

Understanding the nature of human minds, which is oftentimes approached via a resilient and multi-disciplinary perspective from philosophy, anthropology, linguistics, neuroscience, computer science, psychology and beyond, is the fundamental mission of cognitive science [46]. More specifically, between the field pair of computer science and cognitive science, efforts are made bidirectionally [47]: virtues of human learning can be employed in a computational model with a concrete engineering goal, such as to achieve field transcending performance; on the other hand, the gist of a working computational model can be distilled to cast light on decoding the underlying mechanism of human

cognitive functionalities such as attention, memory and representation, to name a few. The focus of this thesis is paradigmatic of the first direction with the emphasis on proposing a human learning inspired and computationally favored model, for acoustic waveform coding.

Recent computational methods for several major audio / speech processing tasks have evidenced that it is critical to understand the mechanism that facilitates human auditory perception. For example, human learning features a multimodal nature of cognition as opposed to deliver multiple advanced functionalities independently [48]. Consider three auditory processing tasks: speech enhancement, automatic speech recognition (ASR), and natural language understanding (NLU), which involves multiple types of multimodal sensory inputs: such as auditory and visual inputs. Suppose you are having a casual conversation with your friend in a bar. Even with the background noise, your brain exerts extra attention on the person you are listening to; visual stimuli such as the gesture and lip movement usually offer additional cues to facilitate the dialog. Moreover, the prior knowledge for the language and the topic at that moment will also be utilized. In contrast, till recently, computational auditory processing has considered these tasks separately. Speech enhancement models aim to suppress the added noise at the best effort. Assuming the speech is perfectly enhanced, ASR models are to minimize the recognition error. Likewise, NLU are rooted in a well functioned ASR front end. However, this can be sub-optimal. For example, machine learning algorithms for speech separation and enhancement are usually to be deployed for the benefit of a speech recognition system, with the assumption that a cleaned-up signal will lower the word error rate (WER) which indicates a higher accuracy of the recognition model. However, as the noise is removed, the speech is often distorted and sounds less natural, leading to even inferior recognition performance.

Concretely, for speech and audio waveform coding, human cognition serves a critical role in both conventional and contemporary deep learning based codecs. In this section, we briefly explain it from two aspects: the design of speech codecs and the loss function for neural audio codecs.

1.4.1 The role of predictive coding in acoustic waveform compression

1.4.1.1 Residual coding

Predictive coding (PC) can be interpreted from a perspective in variety of fields such as cognitive psychology [49] and neuroscience [50], etc. The insight of PC is that human brains care less about learning the representation of the actual reality out there in the world, than minimizing the discrepancy between the reality and the human brain's prediction [51]. In other words, human brain does not process the whole qualia of the sensory input from the environment, but only a small subset of it such that the synthesized prediction from it can lead to a minimized residual error compared to what actually happens later on [52].

The virtue of PC is shown in residual coding, a well-formalized technique applied in compressing videos [53], images [54] and audio signals [55]. As an instance of PC for data compression, residual coding is to not encode the actual input but the residual signal which is the difference between the actual input and what the codec predicts. In practice, with residual coding, the model complexity can be reduced as we do not require the "perfect prediction" to be made from one single codec: the "prediction error" will be processed by another data compression component which conducts residual coding. Moreover, the prediction is usually more accurate as residual coding is to "minimize the discrepancy". Beside, the theory suggests that the update of the states should be conducted locally as a global backward sweep in canonical backpropagation algorithm is not biologically plausible in human brains. Inspired of that, we also propose a local optimization scheme to train our neural residual coding framework, detailed in Chapter. 2.

1.4.1.2 Source-filter modeling

Perhaps one of the most popular PC instances in speech processing is linear predictive coding (LPC) [5]. LPC is a linear regression model to use only a few bits to represent a sampled speech waveform. As a concrete example, LPC assumes that a speech sample at time step t is predicted by

a linear combination of a few samples in the past. The linear coefficients are learned to minimize the prediction error. As indicated by the predictive coding theory, LPC can be integrated in our hierarchical residual coding framework where it performs a specific functionality of autoregression, with the prediction error being processed by other hierarchical layers.

As detailed in Chapter. 3, in digital signal processing, LPC is based on speech production theory [56] which suggests the speech signal is generated from sources which are then filtered by the vocal tract. The way LPC is employed in the analysis of speech signal enables a source-vocal tract separation where the characteristics of the vocal tract can be leveraged in many speech processing tasks, such as automatic speech recognition (ASR).

1.4.2 How auditory perception is not reflected in neural audio codecs

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: if the objective function is ill-defined, will the output with a satisfying objective score be correspondingly preferred by human listeners? 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 listeners will also be likely to consider the decoded speech to be with transparent quality; on the other hand, if the objective score is extremely low, human listeners will not find those samples to be acceptable. The tricky case, however, lies somewhere in the middle: if the

SNR is 15 dB, will that decoded signal be with high or low perceptual quality? In fact, for audio signals, it is very uncertain. While some transparent decoded audio samples may have a relatively low SNR of less than 10 dB, other decoded samples with over 20 dB SNR may still contain audible artifacts. This is, to a certain degree, mitigated by using surrogates of human auditory perception, such as STOI and PESQ both of which are shown to be better correlated to subjective listening scores than SNR. Still, discrepancy exists between these surrogates and the ground truth feedback from human listeners, not to mention that they cannot be directly used in the loss function for model training.

To address the inconsistency between objective and subjective metrics, conventional audio codecs leverage masking effects featured in human auditory perception when compressing the audio signals [57, 58]. The rule of thumb is to only assign bits to encode the signal if otherwise the consequence quantization error will become audible (or not masked). As psychoacoustic studies have pointed out, various masking effects can make acoustic stimuli including those from quantization imperceptible. For example, the acoustic stimulus would be considered inaudible if the interfering stimulus is with a close enough frequency, because they are perceived as one tune; if the acoustic stimulus is with a too low (below 20 Hz) or high (above 20 kHz) frequency, it will be inaudible to human ears; in addition, even the frequency of an acoustic stimulus is within the audible frequency range, of its sound pressure level is not sufficiently high, it will still be inaudible. These psychoacoustic features can help lowering the bitrate and codec complexity without leading to audible performance degradation. 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.

1.5 Summary and Thesis Outline

In this chapter, I presented the problem of acoustic waveform coding; discussed the potential opportunity and challenge of using deep neural networks for this problem; outlined related techniques

in both the conventional and contemporary domains; and motivated the thesis from a cognitive science perspective. We argue that a better consideration of the mechanism of human auditory perception can benefit the design of our computational model for acoustic waveform compression, although the solution is not easy to be achieved: how to apply the theory of predictive coding to the practice of waveform coding to achieve efficiency and scalability? Is it possible to integrate experimental evidence from psychoacoustics into the training procedure of a neural audio codec so as to improve the model performance without at the expense of complexity? As an attempt to answer those questions, we structure the remainder of the thesis as follows:

- **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 detail, 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 propose an algorithm similar to LPC-Net but for waveform quantization. Instead of modeling the whole speech generation process in neural networks, 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 finds a better pivot between LPC module and neural network residual quantization module, such that the performance is more robust even at lower bitrates.

- Having realized the importance of a perceptually salient objective function, 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 an energy consuming 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 summarize our contributions in **chapter 5** and provide a potential direction for future work to better address a set of open ended questions such as the usage of the learned compact representation of clean speech signals in speech recognition and enhancement.

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) [12] or bitwise network techniques [13], etc. Many DNN methods [14][15] 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 [16][17][18][19][59] 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 [20].

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 [60][61][62], but with a high model complexity. A WaveNet based variational autoencoder (VAE) [19] outperforms other low bitrate codecs in the listening test, however, with 20 millions parameters, not realistic for real-time processing in a resource-constrained device. Similarly, codecs built on SampleRNN [63][64] can also be energy-intensive.

Conventional codecs achieve model efficiency by decomposing the whole task of speech signal quantization and reconstruction into multiple stages [65, 66]. For instance, [67] adopts multi-stage vector quantization (MSVQ) to discretize LPC coefficients in 4 kbps. 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

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 framework to host multiple modules, we describe the component module which is aimed to be compact in terms of the model size.

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) [17]. We propose our component model, with much fewer model parameters, that consists of bottleneck residual learning [68], soft-to-hard quantization [69], and sub-pixel convolutional neural networks for upsampling [36]. 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 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 the

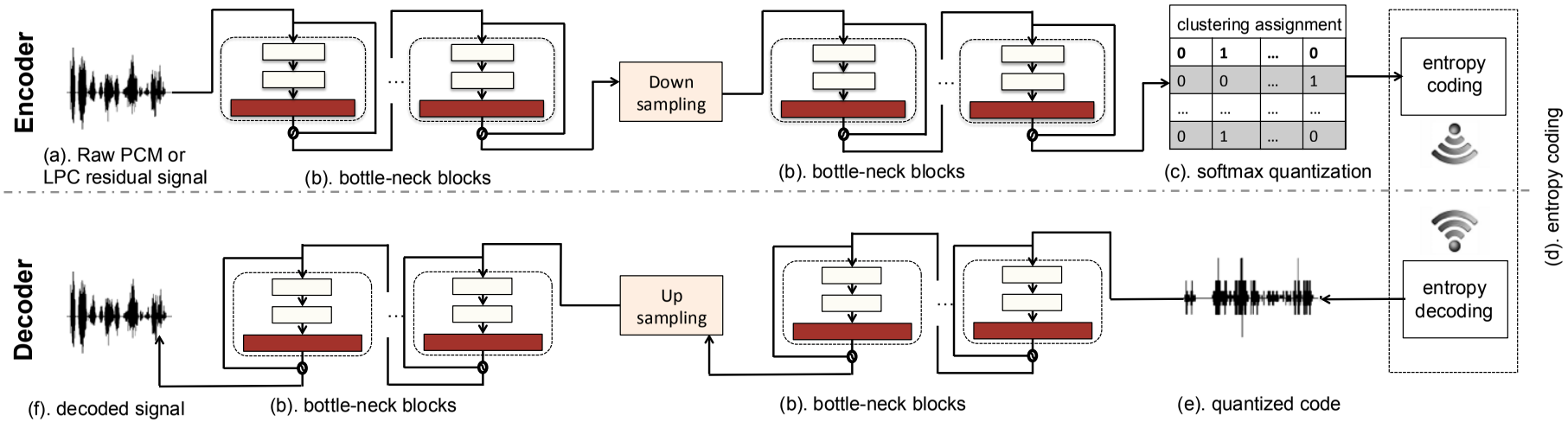


Figure 2.1: Cross-module residual learning pipeline

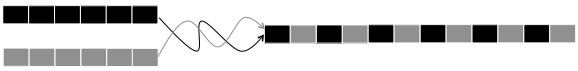


Figure 2.2: The interlacing-based upsampling process.

recently proposed sub-pixel convolution [69], which forms the upsampling operation. The super-pixel convolution is done by interlacing multiple feature maps to expand the size of the window (Figure 3.14). 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 [70], 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.

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

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

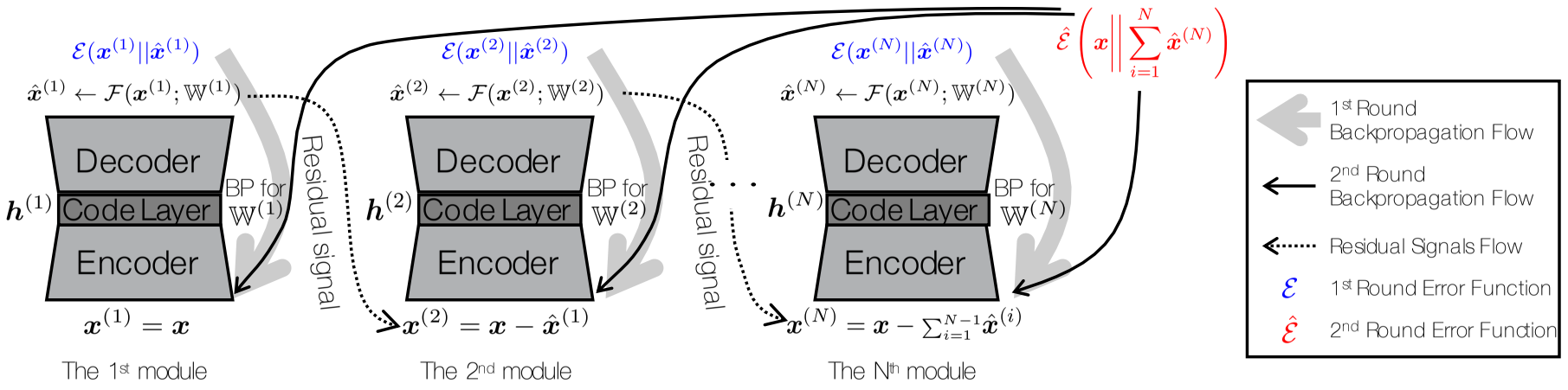


Figure 2.3: Cross-module residual learning pipeline

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, $\mathcal{F}_{\text{dec}}(\boldsymbol{x}^{(i)}; \mathbb{W}_{\text{dec}}^{(i)})\ \forall N$, 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 Training loss justification

2.3.2.1 Mel-scaled loss

To better reflect human sound perception, we follow the common steps to conduct Mel-scaled filter bank analysis. Starting from calculating the power spectrum from short-term Fourier transform (STFT), we subsequently compute the energy in each Mel-scaled filter. The filter bank consists of a set of triangular filters: the response reaches the maximum of 1 at its center frequency and

decreases linearly to 0 at center frequencies of adjacent filters. In other words, the distorted MSE measures the energy difference in each frequency band in Mel scale. The filter bank size defines the granularity level of the comparison. Following [17], we conduct a coarse-to-fine filter bank analysis by setting four filter bank sizes, $F = \{8, 16, 32, 128\}$, to measure the Mel-scaled MSE as shown in Fig.2.4.

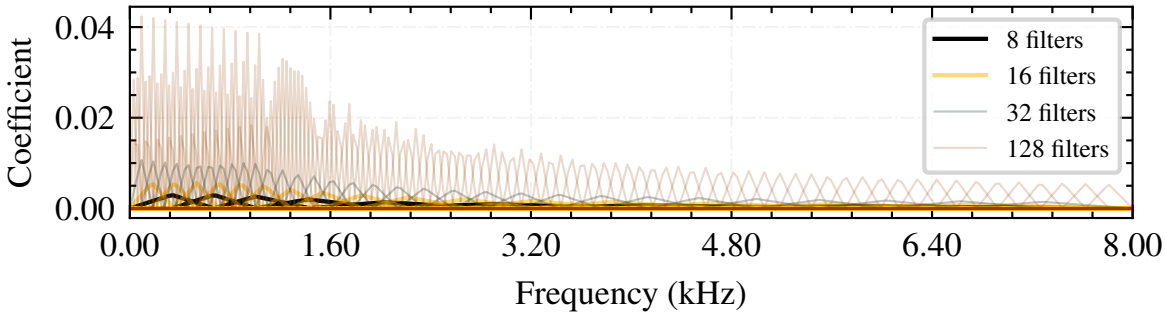


Figure 2.4: Coarse-to-fine filter bank analysis in Mel scale

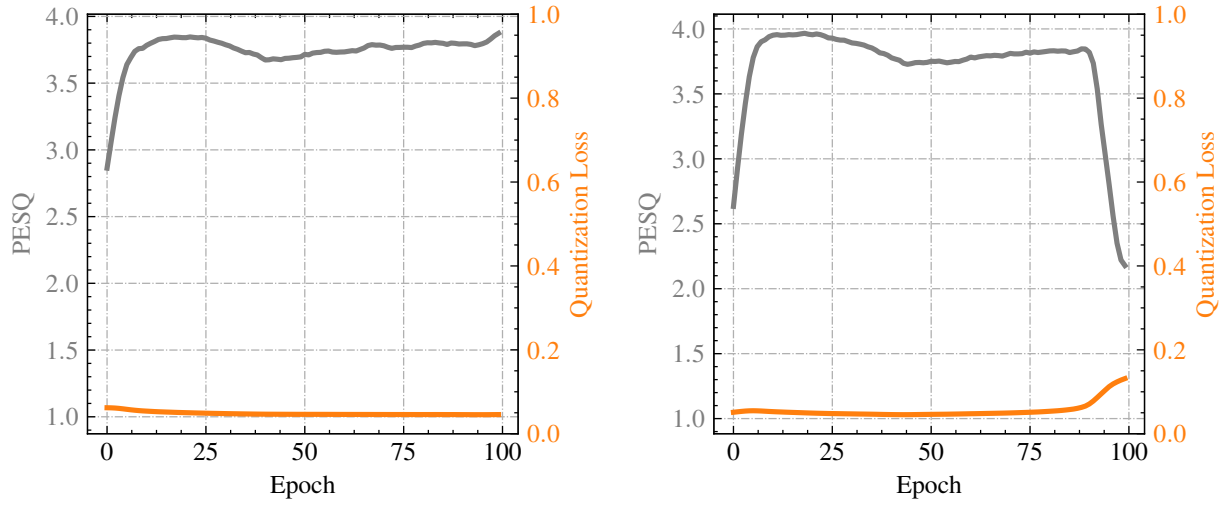
2.3.2.2 Quantization loss

We define the perplexity degree of the soft class assignment $\mathcal{P}(\boldsymbol{A}_{\text{soft}})$ in Eq.2.4.

$$\mathcal{P}(\boldsymbol{A}_{\text{soft}}) = \frac{1}{m}\sum_{m}(\sum_{n}\sqrt{\boldsymbol{A}_{\text{soft}}[m][n]} - \sum_{n}\sqrt{\boldsymbol{A}_{\text{hard}}[m][n]}), \tag{2.4}$$

where the perplexity degree is maximized when the value in each row is $\frac{1}{n}$. In that case, a severe performance degradation is expected at the test time when $\boldsymbol{A}_{\text{soft}}$ is replaced by $\boldsymbol{A}_{\text{hard}}$. $\mathcal{P}(\boldsymbol{A}_{\text{soft}})$ will be 0 when there is only one position with the probability of 1.0 in all rows. To better approximate $\boldsymbol{A}_{\text{hard}}$, the scalar α can be gradually increased during training such that the probability vector in $\boldsymbol{A}_{\text{soft}}$ resembles a one-hot vector. This is termed as "soft-to-hard annealing" which is accompanied by the issue of hand tweaking the scheduler to alter the value of α properly.

In this work, we resort to a regularizer which only keeps the $\boldsymbol{A}_{\text{soft}}$ component in Eq.2.4 to minimize the perplexity degree while setting α to be a moderately large constant for simplicity. As



(a) With quantization regularizer

(b) No quantization regularizer

Figure 2.5: Effects of quantization regularizer on PESQ during model training

shown in Eq.2.5,

$$\frac{\partial \mathcal{P}(\boldsymbol{A}_{\text{soft}})}{\partial \mathbb{W}} = \frac{\partial(\frac{1}{m}\sum_{m}(\sum_{n}\sqrt{\boldsymbol{A}_{\text{soft}}[m][n]}))}{\partial \mathbb{W}}, \tag{2.5}$$

since each row of $\boldsymbol{A}_{\text{hard}}$ in Eq.2.4 is an one-hot vector, summed up to a constant. The derivative of the regularizer with respect to the model weight is the same with that of the perplexity degree.

Generally, the scalar value $\alpha = 300$ should be large enough to initiate the training process, which means the corresponding quantization loss is close to 1. Still, the quantization regularizer is necessary to guarantee the level of spikiness for each row in the soft assignment matrix. As shown in Figure 2.5, without the quantization regularizer, the PESQ score will drop abruptly as the quantization loss surges.

2.3.3 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:

$$\arg\min_{\mathbb{W}^{(1)}\ldots\mathbb{W}^{(N)}} \hat{\mathcal{E}}\left(\boldsymbol{x}\middle|\middle|\sum_{i=1}^{N}\mathcal{F}\big(\boldsymbol{x}^{(i)};\mathbb{W}^{(i)}\big)\right). \tag{2.6}$$

During the finetune 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.

2.3.4 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 sample rate, respectively, the bitrate 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}=2.4 kbps, 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 [71] [72]. 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

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)	[(9, 100, 20), (9, 20, 20), (9, 20, 100)] ×2	(512, 100)
Downsampling	(512, 100)	(9, 100, 100)	(256, 100)
2nd bottleneck	(256, 100)	[(9, 100, 20), (9, 20, 20), (9, 20, 100)] ×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)	[(9, 100, 20), (9, 20, 20), (9, 20, 100)] ×2	(256, 100)
Upsampling	(256, 100)	(9, 100, 100)	(512, 50)
2nd bottleneck	(512, 50)	[(9, 50, 20), (9, 20, 20), (9, 20, 50)] ×2	(512, 50)
Change channel	(512, 50)	(9, 50, 1)	(512, 1)

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 [17], where the use of LPC was not tested. Therefore, for the subjective quality, we perform MUSHRA tests [73] to investigate how the performance from CMRL with an LPC residual input is compared with AMR-WB and OPUS

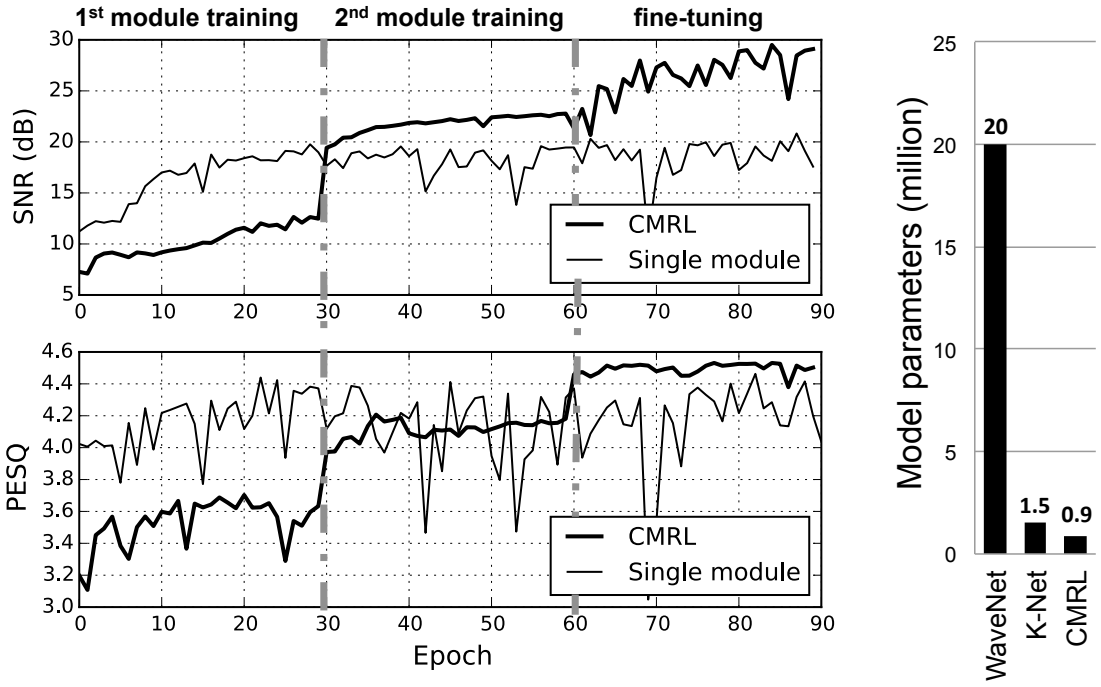
at high bitrates.

300 and 50 speakers are randomly selected from TIMIT [74] training and test datasets, respectively. 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. The fine-tuning step uses 0.00002 as the learning rate, too. Each window contains 512 samples with the overlap size of 32. We use Adam optimizer [75] with the batch size of 128 frames. Each module is trained for 30 epochs followed by the fine-tuning 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) [28]. Figure 2.6 (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 that are unaltered. For both SNR and PESQ, the plot shows a noticeable performance jump as



(a) (b)

Figure 2.6: (a) SNR and PESQ per epoch (b) model complexity

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

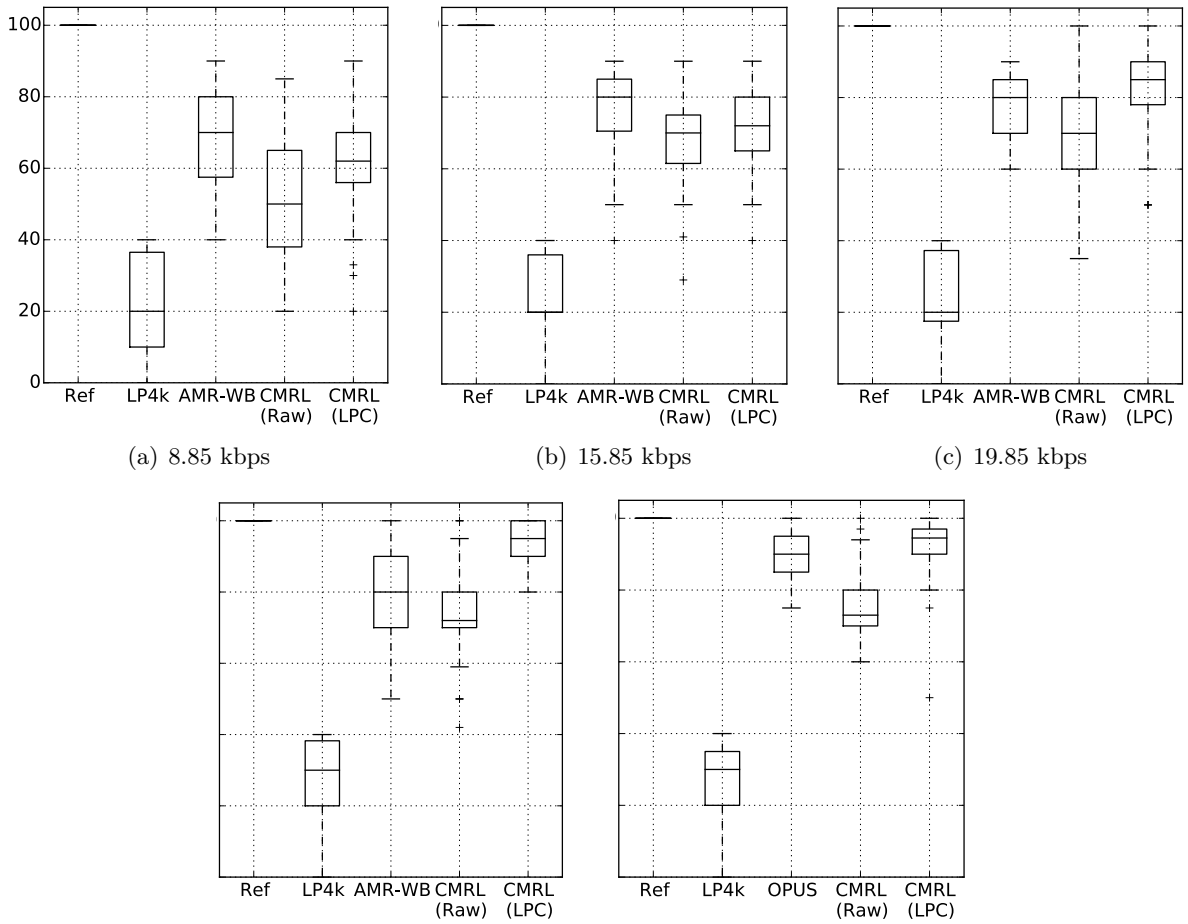
Metrics	SNR (dB)				PESQ			
Bitrate (kbps)	8.85	15.85	19.85	23.85	8.85	15.85	19.85	23.85
AMR-WB	9.82	11.93	12.46	12.73	3.41	3.99	4.09	4.13
K-Net	-	-	-	-	3.63	4.13	4.22	4.30
CMRL	**13.45**	**16.35**	**17.18**	**17.33**	**3.69**	**4.21**	**4.34**	**4.42**

the second module is included, followed by another jump by a round of fine-tuning.

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

2.4.2 Subjective test

Figure 2.7 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



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

(d) 23.85 kbps (e) 23.85 kbps

Figure 2.7: 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 23.85 kbps.

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 μ law companding

When it comes to bit-depth reduction for speech compression, μ law companding serves a well known algorithm to lower the bit-depth from 16 bits to 8 bits in telecommunication standards.

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

The algorithm consists of two reciprocal transformations: μ law compressing and μ law expanding: the former transformation warping the signal via a logarithm based function while the latter transformation converting the warped representation back to the original scale. With μ law companding, the process of bit-depth reduction is less subject to quantization loss comparing with the linear quantization counterpart. In addition, it facilitates the optimization of deep neural network based auto-regressive models.

2.4.3.1 The role of μ law companding in 8-bit PCM codecs

Consider a speech waveform in the 16-bit fixed point representation. By definition, the representation has 65536, or 2^{16}, distinct integer values, ranging in $(-32768, +32767)$. Suppose each sample in the waveform is only allowed to be represented by an 8-bit fixed point, it has to be quantized into one out of the 256, or 2^8, integer values, ranged in $(-128, +127)$. How to design the quantization algorithm to limit the quantization loss is critical to preserve the speech quality.

One straightforward way to conduct the quantization is via linear mapping. For the highest positive value, +32767, when converting it to the 8-bit format, it becomes +127: the process is to divide +32767 by +256, with the remainder 255 being the quantization error. In other words, when expanding +127 back to the 16-bit format, it will be 32512. Comparing to its original value, the error rate is $|\frac{32767-32512}{32767}| = 0.0078\%$. The problem occurs when those sample values to be quantized are close to 0. Suppose we are compressing +128: with linear mapping, it will be rounded to 0 and not possible to be recovered. In fact, this issue applies to all positive samples ranging in $(-255, -1)$ and $(+1, +255)$, with the corresponding error rate of 100%. Unfortunately, most samples in speech waveforms are with small amplitudes, which makes linear mapping undesirable.

In contrast, μ law companding focuses more on samples with smaller amplitudes by non-linearly mapping the waveform with a logarithm based function before conducting quantization. The μ law compressing function is defined in Eq.2.7, where $\mu \in (0, \inf)$ determines the non-linearty of the

Table 2.3: Error rate comparison between μ law companding and linear mapping

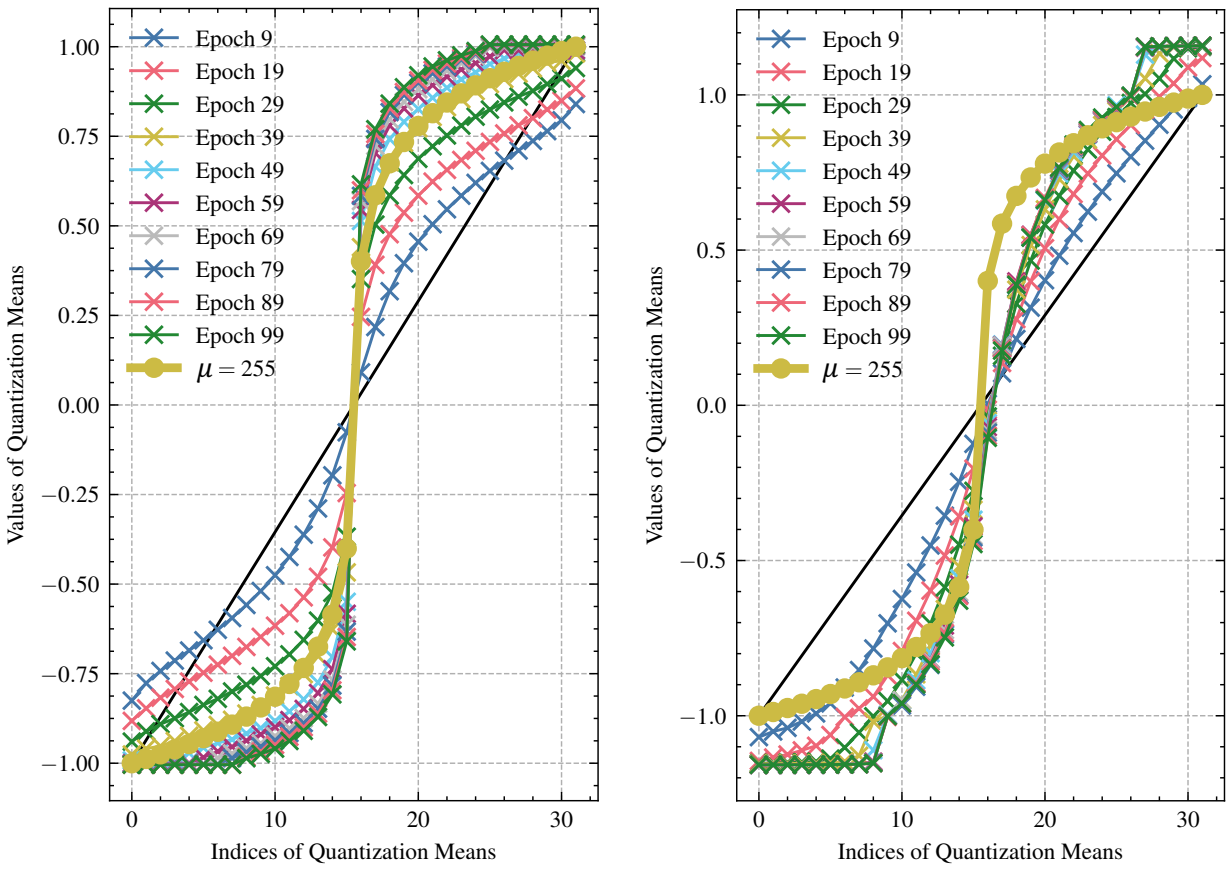
Sample values	200	400	800	1600	3200	6400	12800	25600
Normalized values	0.0061	0.0122	0.0244	0.0488	0.0977	0.1953	0.3906	0.7813
μ law warped values	0.1693	0.2550	0.3566	0.4687	0.5869	0.7084	0.8316	0.9557
Quantization values	21	32	45	59	75	90	106	122
Recovered values	191	386	775	1528	3183	6214	12555	25239
Error rates	**0.0450**	**0.0350**	**0.0312**	0.0450	**0.0053**	0.0291	0.0191	0.0141
Error rates (linear)	1.0000	0.3600	0.0400	**0.0400**	0.0400	**0.0000**	**0.0000**	**0.0000**
Coverage rate	0.7312	0.8555	0.9422	0.9847	0.9986	0.9999	1.0000	1.0000

function and $\boldsymbol{x} \in (-1, 1)$ is the input waveform sample. The function is almost linear when μ gets close to 0.

$$\mathcal{F}_{\text{compression}}(\boldsymbol{x}, \mu) = \text{sign}(\boldsymbol{x}) \frac{ln(1 + \mu|\boldsymbol{x}|)}{ln(1 + \mu)} \tag{2.7}$$

By default, μ is set to be 255. Consider the same case where +128 is to be quantized. With μ law companding, it is firstly converted as $\mathcal{F}_{\text{compression}}(\frac{128}{32767}, 255) = 0.1247$, as opposed to 0.0039 with linear transformation; in 8-bit scale, 0.1247 is quantized as $\textit{floor}(0.1247 * 128) = 15$. The μ law expanding algorithm recovers the sample in 16-bit scale as $\textit{ceil}(\mathcal{F}^{-1}_{\text{compression}}(\frac{15}{128}, 255)) = 118$. The error rate is $|\frac{128-118}{128}| = 0.0781\%$, as opposed to the previous 100% from the linear mapping approach.

Table.2.3 compares error rates from μ law companding and linear companding: unless sample values are relatively small, there is no clean evidence that μ law companding is superior to linear companding unless when the sample values are relatively small, which, as stated before, is indeed the case for speech signals. We calculate the coverage rate of samples from 20 TIMIT test utterances, where the coverage rate is defined as a percentage, quantifying the portion of samples with the absolute value less than the threshold: over 85% samples are bounded in $(-400, 400)$.



(a) μ law transformed before quantization

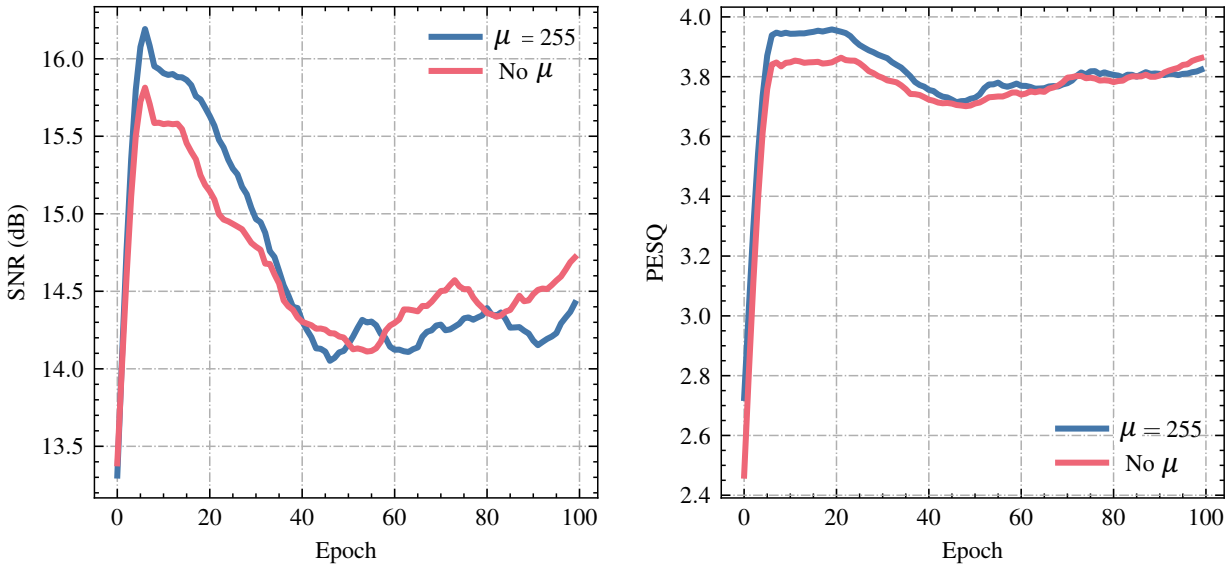
(b) Direct encoder output quantization

Figure 2.8: Effects of μ law transformation on quantization means, SNR and PESQ

2.4.3.2 μ law companding in neural codecs

μ law companding is adopted in WaveNet, an end-to-end auto-regressive network, where the model is optimized as a classification problem. Without μ law companding, each model output by default is in 16-bit fixed point representation. Having that, it is formulated as a vector with the length of 16. During model training, cross entropy is employed to measure the discrepancy between ground truth vectors and those generated from the model. The challenge lies in the high dimensional and sparse vector space, which makes the optimization almost intractable. To tackle that, WaveNet uses μ law companding to achieve the bit-depth reduction from 16 bits to 8 bits, as applied in PCM codecs, making the model training a much more feasible process.

The proposed waveform codec, in contrast, does not reduce the bit-depth via μ law companding



(a) The validation SNR curve during training

(b) The validation PESQ curve during training

Figure 2.9: Effects of μ law transformation on quantization means, SNR and PESQ

but soft-to-hard quantization by leveraging entropy coding. To investigate if μ law companding will benefit the quantization loss, we apply μ law compressing algorithm to the encoder output and μ law expansion algorithm to the decoder input, while training the model at 12.85 kbps. The SNR and PESQ from the validation dataset during model training is compared to the counterpart without μ law companding, in Figure 2.9. It is observed that μ law companding does not noticeably improve the speech quality especially in terms of PESQ, when the bitrate target is reached (at the 99-th epoch). During the beginning of model training when the entropy penalty is not as strict, μ law curves are relatively higher as the encoder outputs are zoomed in via μ law compressing algorithm, which means those samples are quantized with a relatively higher resolution. However, with the entropy regularizer navigating the model towards the target bitrate, the gain starts to disappear as those zoomed in samples do not have as many bits to represent. Therefore, we do not apply μ law companding to our proposed system.

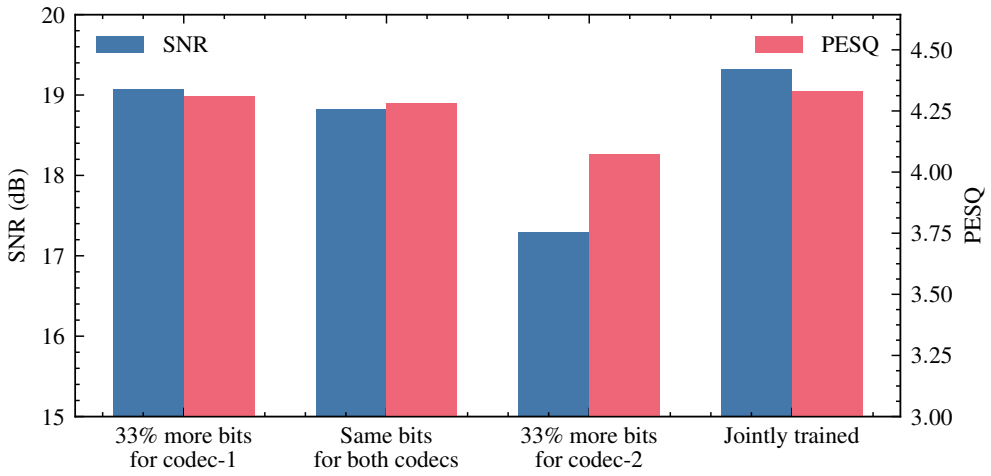


Figure 2.10: Objective measures under different bit allocation schemes

2.4.4 Bit allocation between neural codecs

To answer the question "how does these two neural codecs share bits", we first manually specify the target bitrate for each codec: in trial-i, the 1^{st} codec (codec-1) gains 33% more bits than the 2^{nd} codec (codec-2); in trial-ii, both codecs are assigned with the same amount of bits; and in trial-iii, codec-2 gains 33% more bits than codec-1. It is observed that better objective measures in terms of SNR and PESQ are found in trial-i, which is aligned with our intuition as codec-2 only compresses residuals. To figure out the optimal bit allocation scheme, we jointly train both codecs and eventually find the highest SNR and PESQ scores when 16% more bits are assigned to codec-1, as shown in Figure 2.10.

2.4.5 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 [19], and outperforms Kankanahalli-Net with 40% less model parameters. Figure 2.6 (b) summarizes the comparison.

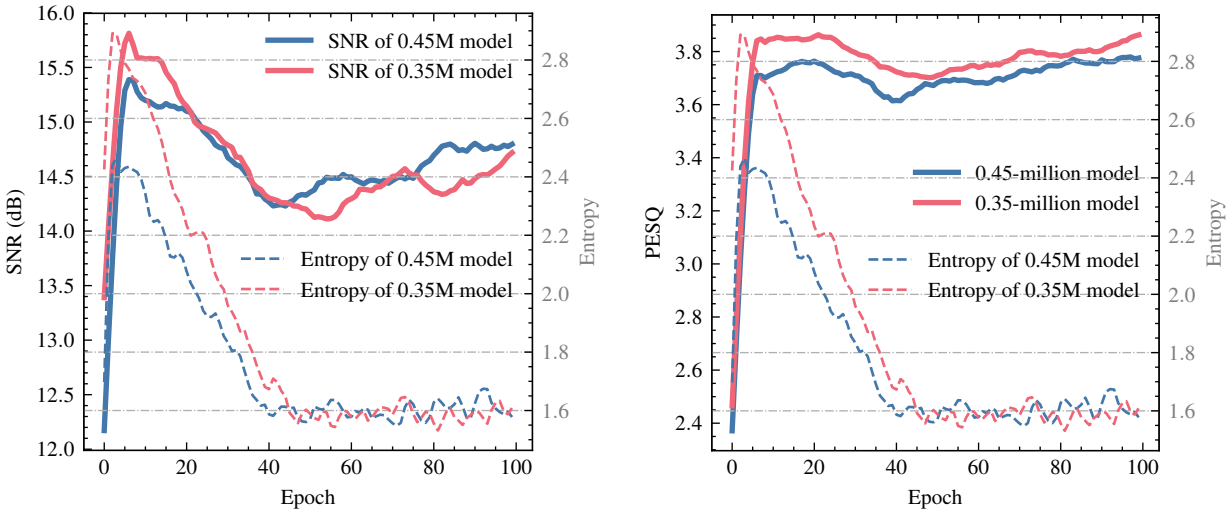
Table 2.4: Architecture of the alternative neural waveform codec: input and output tensors are shaped as (sample, channel), while the kernel is represented as (kernel size, in channel, out channel).

Layer	Input shape	Kernel shape	Output shape
Change expansion	(512, 1)	(55, 1, 100)	(512, 100)
1st bottleneck	(512, 100)	[(1, 100, 20), (15, 20, 20)†, (15, 20, 20)†, (9, 20, 100)] ×2	(512, 100)
Downsampling	(512, 100)	(9, 100, 100)	(256, 100)
2nd bottleneck	(512, 100)	[(1, 100, 20), (15, 20, 20)†, (15, 20, 20)†, (9, 20, 100)] ×2	(512, 100)
Change reduction	(256, 100)	(9, 100, 1)	(256, 1)
Change expansion	(256, 1)	(9, 1, 100)	(256, 100)
1st bottleneck	(256, 100)	[(1, 100, 20), (15, 20, 20)†, (15, 20, 20)†, (9, 20, 100)] ×2	(256, 100)
Upsampling	(256, 100)	(9, 100, 1), (1, 100, 100)	(512, 50)
2nd bottleneck	(512, 50)	[(1, 50, 20), (15, 20, 20)†, (15, 20, 20)†, (9, 20, 50)] ×2	(512, 50)
Change reduction	(512, 50)	(55, 50, 1)	(512, 1)

2.4.5.1 An alternative codec with 0.35M parameters

Thus far, the neural codec involved in our system consists of 0.45 million parameters, which is already simplified from 1.6 million parameters in [17]. From there, we could further reduce the model size by using advanced residual learning blocks and convolutional operations. Table.2.4 details the topology of a neural codec with only 0.35 million parameters. Comparing to its predecessor, the residual block in this alternative adopts gated linear unit [31] and depth-wise separable convolution [76] for upsampling.

Figure 2.11 shows that the performance does not degrade by a large margin when the model size (parameters) is reduced from 0.45 million to 0.35 million. Particularly, the more compact model design leads to a slightly higher PESQ score. Both models are trained with the same hyperparameter values with the blending weights for the time domain loss, mel-scaled frequency



(a) The validation SNR curve during training

(b) The validation PESQ curve during training

Figure 2.11: Performance degradation in terms of SNR and PESQ when the amount of model parameters decreases from 0.45 million to 0.35 million.

Table 2.5: Execution time ratio during model inference: the ratio is defined as the execution time to encode and decode the test signals divided by the duration of those signals.

Model type	0.45M	0.35M	CMRL (0.45M×2)	CMRL (0.35M×2)
Execution time ratio (%) on one Tesla V100	12.49	13.38	20.69	21.12
Execution time ratio (%) on one Tesla K80	24.45	22.53	39.42	38.82
Execution time ratio (%) on eight CPU cores	20.76	18.91	35.17	33.80
Execution time ratio (%) on one CPU core	46.88	42.44	87.38	80.21

domain loss, quantization regularizer and entropy regularizer being $50, 5, 10$ and 0, respectively.

2.4.5.2 Delay and execution time

The delay of our system is approximately $16ms$, or a duration of a half wave frame. This is because the system conducts frame-wise speech coding with convolutional neural network layers: to decode a sample, the system needs to look at the future samples with the size of 256 in this work. To remove the delay as such, the current convolutional layer can be implemented as causal convolution as it does not require future samples to make predictions, which is at the expense of the speech quality.

While the system delay is defined by the design of the codec, the execution time is related to the model complexity and also a factor to be considered: the bottom line is that the encoding and decoding time for a signal is expected to not exceed the duration of that signal to facilitate real-time telecommunication. WaveNet codec does not introduce the system delay due to its causal convolutional layers, but its execution time, though not reported, can be rather high as it is an auto-regressive model with 20 million parameters, predicting every single sample by looking back a large receptive field of samples. Table.2.5 lists the execution time ratio of our models. The execution time ratio for Kankanahalli-Net is 15.93% for GPU and 71.40% for CPU [2]. On both CPU [3] and GPU, our models (0.45M and 0.35M) run faster due to the significantly reduced model size. Even with CMRL to enable residual coding among 2 codecs, our models still achieve lower execution time ratios on CPU. However, the ratio comparison results between the 0.45M model and 0.35M model are not consistent. We believe this is because the internal implementation of different Residual learning blocks in TensorFlow may lead to various runtime optimization effects.

2.4.5.3 Network compression

It is worthy to note that network compression is a more straightforward way, than reducing the model size, for the feedforward speedup during model inference. The goal of network compression is boiled down to reducing the amount of multiplicative operations during inference. To that end, there are following major methodologies:

- Model quantization aims to reduce the bit-depth of the model weights to 8 bits or even fewer, as the hardware may only support integer arithmetic operations for run-time inference.

- Model pruning is to reduce the amount of non-zero weights by setting a threshold: all weights with the absolute value below the threshold are to be pruned, or set to zero. The pruning

[2] Kankanahalli-Net [17] claims that it requires $4.78ms$ to encode and decode a $30ms$ signal on a GeForce GTX 1080 Ti GPU, and $21.42ms$ on an Intel i7-4970K CPU (3.8GHz).

[3] The model is Intel Xeon Processor E5-2670 V3 (2.3GHz).

can be conducted either in a structured or unstructured way: (a) model selection/structured pruning is to choose a model structure with pruned layers/channels and small performance degradation [77, 78]; and (b) zero-weight compression/sparse pruning is to prune small-value weights to zero [79, 80, 81]. Model selection differs from sparse pruning in that it deletes entire channels or layers, showing a more efficient speedup during inference, yet with a more severe performance degradation [77, 78]. These two types of methods are usually complementary: after being structurally pruned, a model can also undergo further zero-weight compression to improve the inference speed.

In addition to network compression methods mentioned above, there are other alternatives such as knowledge distillation to transfer what is learned from a large model to a small model via teacher-student learning scheme, low-rank factorization to approximate matrices by reducing redundancy, etc.

2.5 Summary

In this chapter, 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. We omitted the details of LPC but simply shoehorned it from AMR-WB into our neural codec to show a proof-of-concept. The mechanism of LPC is articulated in Chapter 3 where a trainable LPC analyzer is also proposed.

Chapter 3

COLLABORATIVE QUANTIZATION: BRINGING LPC TO DNN

3.1 Motivation: Why Modeling Everything from Scratch

Ubiquitous are end-to-end neural networks: finally we could out-source the nerve-wracking feature engineering process and throw as much raw data as possible into a neural engine while "hoping for the best". It seems that we, engineers and researchers, are substantially spoiled in the era where big data is trending. In fact, in Chapter. 2, we proposed an end-to-end neural codec as such, consuming the raw waveform and reconstructing the input by minimizing an objectively defined loss function. While many end-to-end models have indeed achieved their smash debut in various domains, such as image recognition, speech recognition, etc, the methodology of proposing new models becomes a matter of scaling up the data and computational resources, in addition to great patience. Having a sufficient amount of nodes, you could incubate numerous deep neural networks harvesting the big training dataset. After waiting for hours, days or weeks, terminate those not promising and initiate new training processes by adding variations to those well-performing networks. Usually with months of struggling, you are supposed to have found the winning model and supportive experimental results.

This would be with no problem if the premise, that the data and computational resources are sufficient, was true. However, in most under-developed areas in the world, resources such as super-computers and the bandwidth of the Internet may not support the fruit of recent end-to-end neural engines. In terms of data telecommunication [82, 83, 8, 9], Google's WaveNet speech codec operates at a high computational cost which is only possible if it is on high end GPUs. In other words, it will still be a long way before the WaveNet codec being used by the world citizens. Besides, learning from big data to find the logic or prior knowledge that have been well investigated for decades at

the expense of computational overhead is not reasonable. This is not to discredit the development of DNN, but to rethink ways to host previous scientific findings on neural networks to achieve both high performance and efficiency.

In terms of speech coding which quantizes speech signals into a compact bitstream for efficient transmission and storage, it would be beneficial to consider how speech is produced when designing neural codecs. The speech production model is based on source-filter theory where linear predictive coding (LPC) [84], an all-pole linear filter, serves a critical component. LPC can efficiently model power spectrum with only a few coefficients through Levinson-Durbin algorithm [11]. In fact, among two major categories of speech codecs, vocoders use few parameters to model the vocal tract response leading to very low bitrates and minimal computational complexity. For waveform coders, such as Opus [85], Speex [86] and AMR-WB [87], the residual is directly compressed to the desired bitrate before being synthesized to the decoded signal.

LPC should be applied to neural speech codecs as well, considering that they have greatly improved the synthesized speech quality [24] at the cost of model complexity during the decoding process [44]. 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 [88]. To make such a system more efficient, LPC can still unload computational overheads from neural networks. LPCNet combines WaveRNN [89] and LPC to shrink down the complexity to 3 GFLOPS which enables real-time coding [26, 90]. 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 [43] uses LPC as a pre-processor and a variation of [42] 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 [91]. 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 Preliminaries of Linear Predictive Coding (LPC)

It is almost not possible to steer clear of terminologies in digital signal processing (DSP) even when the influence of neural networks on speech processing is getting pervasive. This section is therefore dedicated to revisiting some of the fundamental concepts and terminologies in DSP to facilitate the discussion of how our neural codec incorporates the speech production model for scalable and efficient waveform compression. To simplify the deliberation on the preliminaries while serving the purposes, the section is based on a set of concrete examples.

3.2.1 Speech Production Modeling with Source and Filter

A filter, or a system, is a function that is applied to the input signal to generate the output signal. Suppose the discrete time input signal, parameterized by the time step t, $\boldsymbol{x}[t]$ is filtered by system τ to output $\boldsymbol{y}[t]$, the process is formulated as $\boldsymbol{y}[t] = \tau(\boldsymbol{x}[t])$, or $\boldsymbol{x}[t] \overset{\tau}{\mapsto} \boldsymbol{y}[t]$. In DSP, the term "filter" and "system" are often used interchangeably.

3.2.1.1 z-transform

Aside from the time domain (or n-domain) and frequency domain (or ω-domain), a third domain, named z-domain, is also widely used in describing mathematical analysis and synthesis of signals and systems due to its algebraic nature.

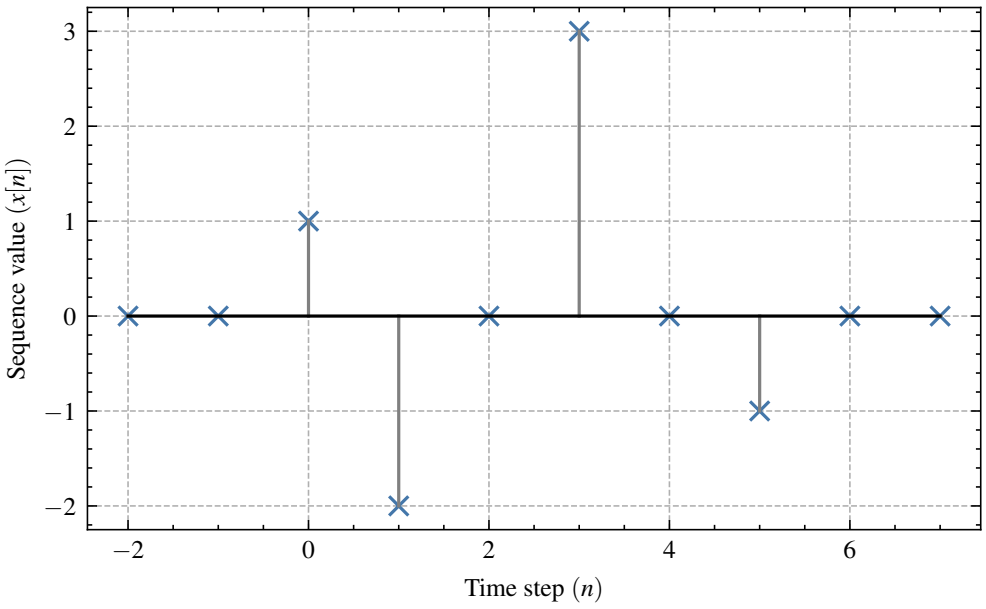


Figure 3.1: An example of a discrete time signal

Consider a time series signal shown in Figure.3.1[1], the signal value is zero if $n < -2$ or $n > 7$. To represent it, we firstly consider a time-domain representation in terms of impulse sequences as equation 3.1

$$x[n] = \delta[n] - 2\delta[n-1] + 3\delta[n-3] - \delta[n-5], \tag{3.1}$$

where $\delta[n-k]$ is a unit impulse with the value being 1 when $n = k$ and 0, otherwise. Equivalently, for this specific example, the z-transform, $X(z)$, is given as equation 3.2, where z is a complex number.

$$X(z) = 1 - 2z^{-1} + 3z^{-3} - z^{-5} \tag{3.2}$$

In z-domain, signal values are represented as polynomial coefficients. In fact, the major advantage of converting a signal from time-domain to z-domain is that the system can be described as polynomials and rational functions, which simplifies the analysis of linear time invariant (LTI) systems. For example, a system is represented as $X(z) = 1 + \frac{1}{2}z^{-1} + \frac{1}{2}z^{-2} - z^{-3}$. It can be factored as $X(z) = (1 + z^{-1})(1 - \frac{1}{2}z^{-1} + z^{-2})$, in which the z value from the second term can be evaluated as

[1]This example is derived from the book DSP First [92].

$\frac{1}{4} \pm j\frac{\sqrt{15}}{4}$, according to the quadratic formula.

Finding these values are of significant importance to understand properties of an LTI system. The roots from the numerator are called zeros and the denominator poles, which are displayed in the pole-zero plot.

3.2.1.2 All-pole filter for vocal tract formant modeling

Linear predictive coding (LPC) is based on a bold assumption that a given speech sample at time n, $s[n]$, can be approximated by k previous samples as $\boldsymbol{x}[n] \approx \sum_{i=1}^{k} a_i \boldsymbol{x}[n-i]$. Suppose the estimation error, termed as the residual or excitation, is defined as a normalized excitation, $\delta[n]$, and a gain (scalar), G, the LPC analysis process is essentially an auto-regressive algorithm as equation 3.3

$$\boldsymbol{x}[n] = \sum_{i=1}^{k} a_i \boldsymbol{x}[n-i] + G\delta[n]. \tag{3.3}$$

In z-domain, this is re-written as equation 3.4 which is equivalent to equation 3.5 where $A(z) = 1 - \sum_{i=1}^{k} a_i z^{-i}$. Since the system for LPC analysis, $X(z)$, has a constant value of 1 for the numerator, it does not have any zeros: as all roots are from the denominator, the LPC analyzer is therefore an all-pole filter.

$$S(z) = \sum_{i=1}^{k} a_i z^{-i} S(z) + G\Delta(z) \tag{3.4}$$

$$X(z) = \frac{S(z)}{G\Delta(z)} = \frac{1}{A(z)} \tag{3.5}$$

One reason that $X(z)$ is an all-pole filter is because of the fact that samples for $s[n]$ estimation are all past samples. For speech processing, this is usually found sufficient for feature extraction such as in speech coding and recognition. In addition, the algorithm to evaluate the filter coefficients, a_i, and the gain is more computationally efficient with the design as such. Algorithms to evaluate the LPC filter can be based on autocorrelation (such as the Levinson-Durbin Algorithm) or covariance [93].

The speech signal is generated as a collaboration among the lung, glottis, mouth and novel cavities. This production process can be formulated as a source-filter model, where the lung and glottis are the source yielding the impulse sequence or some white noise, while the latter serves a vocal tract or a resonator, responding to certain frequencies. Due to the physical structure of the vocal tract, certain frequencies are "favored" where signals being bounced by surfaces of the vocal tract are being added up. In other words, in the time-frequency domain, the resonance can be visualized from the envelope of the spectrogram in that formants are the peaks in the spectral envelope (Figure 3.2).

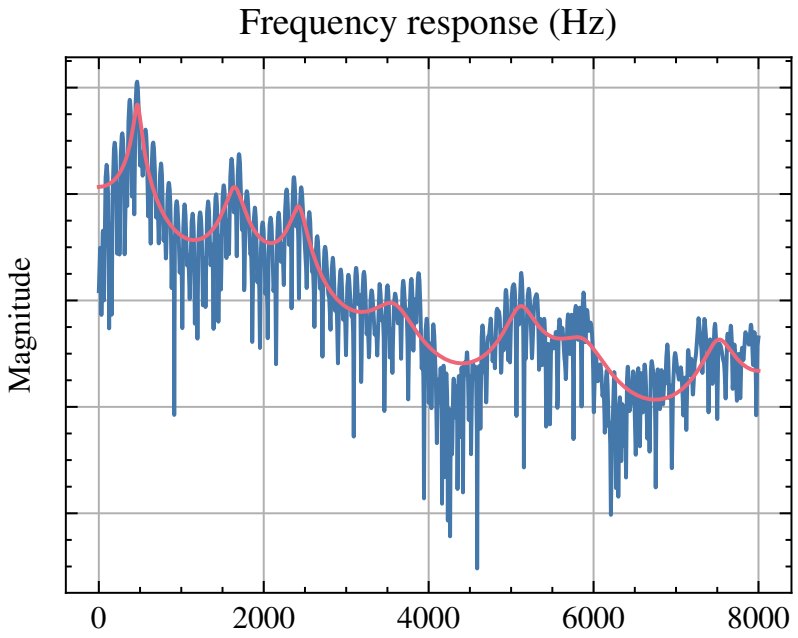
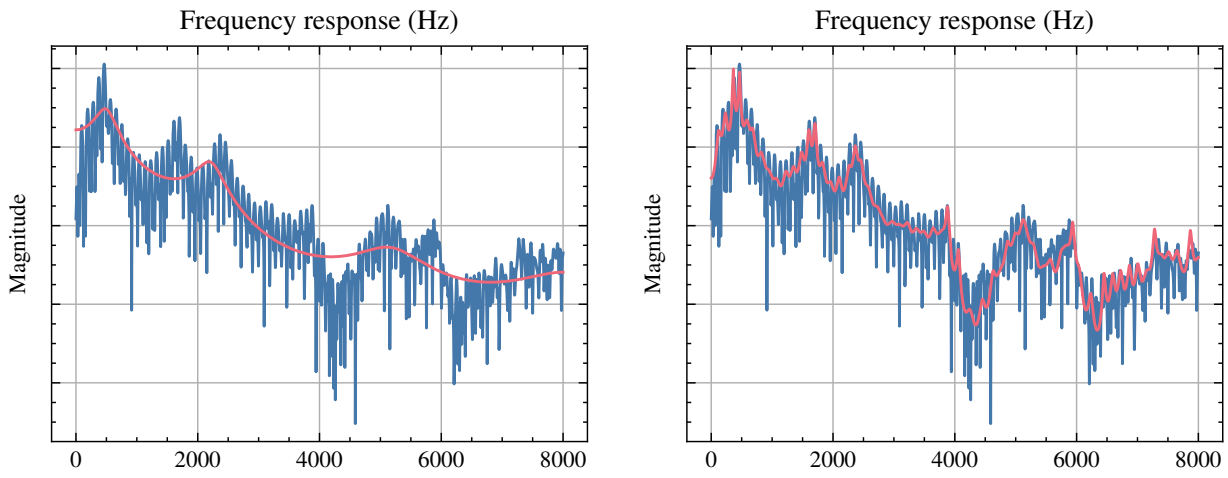


Figure 3.2: Estimating the spectral envelope with LPC algorithm: the scale in the y axis is omitted for the alignment between the power spectral density curve (in blue color) and the envelope (in red color).

The order of LPC, k, defines the amount of samples it needs to look back for an auto-regressive estimation. From a machine learning's perspective, the order number is related to the level of granularity for the curve fitting: if the order is too low, the algorithm may miss quite a few formants and only reflect the "general trend" of the curve (Figure.3.3(a)); on the other hand, even with a large order, the algorithm cannot perfectly estimate the power spectral density curve, due



(a) LPC order of 8

(b) LPC order of 128

Figure 3.3: The estimated spectral envelope with either a too low (a) or high (b) LPC order.

to the fact that it is a linear auto-regression algorithm (Figure.3.3(b)). Empirically, the order of LPC is related to the sample rate of the speech signal: for signals in TIMIT corpus with 16000 samples per second, a 16-pole LPC model is usually sufficient.

3.2.2 LPC in conventional speech vocoding

While the dissertation focuses on waveform coding, it is worth mentioning speech vocoding as another major type of codec. Unlike waveform coding, where the goal is to recover the waveform as exact as possible, a vocoder does not care as much about reconstructing the waveform. Instead, it is mainly used to achieve very low bitrates, usually less than 4 kbps. How is that possible? A vocoder does not compress the waveform directly but extracts just a few parameters, or "cues", that could characterize the speech signal. Since those parameters, after being quantized, can be transmitted at a very low bitrate, the vocoder is mainly to synthesize the speech from the parameters. In other words, there are 2 major tasks for a well performing vocoder: finding the right set of parameters, and the right synthesis algorithm. Therefore, a speech vocoder is also named parametric codec.

To use image compression as an analogy: suppose an image is to be compressed and transmitted to the receiving side. The image is with 501×501 pixels and only has one black circle in the very

middle with the radius of 200 pixels. If we compress it in a parametric way, we would propose a "circle production model" that a circle is well-specified when its center and radius are defined (which is actually the case). Having that, what we need to encode is not the entire image but those parameters about the image size, circle's center and radius, line width, etc, which could significantly reduce the bitrate. Obviously, this assumption does not apply to all image objects. The speech production model, on the other hand, has actually played a critical role in vocoders since the 1970s.

Fortunately, for speech vocoding, the parameters have almost always been the pitch, voicing level, and spectral envelope. The pitch carries quite a lot of information on "how" the speech is being made in terms of the prosody and emotion, etc. Spectral envelope represents the vocal tract, or the system of the input signal to be filtered, telling "what" speech is being generated. Usually the speech pattern represented by these parameters alter every 20 ms, and therefore vocoders operate frame-wise. Voicing level determines if the corresponding frame is voiced or un-voiced based on the accumulated energy of that frame. The voiced frame is usually modeled as a glottal impulse while the un-voiced signal is modeled as white noise. For speech vocoders, the LPC algorithm estimates the spectral envelope yielding a set of LPC coefficients, given the cepstrum of each input speech frame.

3.2.3 LPC in neural speech synthesis: LPCNet

The neural auto-regressive model, such as WaveNet [24], WaveRNN [25], SampleRNN [63], has been quite a speech synthesis smash. However, the motion to apply it into speech coding is much less well-received, even though text-to-speech (TTS) synthesis and low bitrate speech vocoding are two highly correlated domains (for TTS, speech parameters are extracted from text via other types of neural networks). Among many reasons, the model complexity is the major concern. While TTS synthesis can be a cloud service, speech vocoding is usually with low-power devices where resources are at a premium.

Therefore, for neural speech vocoding system, the most cited recent paper is not to boast the capacity of speech modeling via neural networks but introduce LPC to a compact network (consequently termed as LPCNet [26]) lowering the test time computational overhead. The high

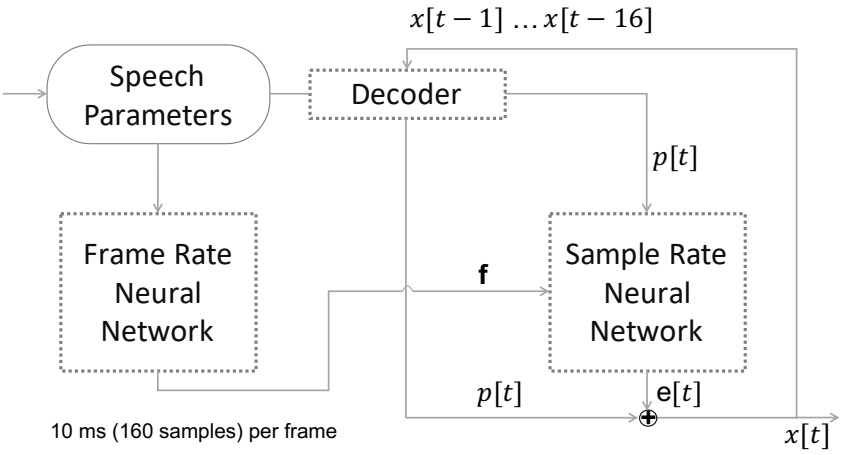


Figure 3.4: A high level diagram of LPCNet

level diagram of LPCNet is shown in Figure.3.4, where the whole proposed model serves a decoder of a parametric codec. As mentioned previously, the compression of speech from a parametric codec is achieved from the assumption that the speech can be synthesized by just a few "speech parameters". In other words, the quantization is just applied to those parameters. The major distinction of LPCNet against full neural network synthesis systems is that it incorporates LPC into the speech synthesis process. In Figure.3.4, the frame rate network distills a vector $\boldsymbol{f}$ for each 10 ms frame. The sample at t time step, s_t is estimated via linear prediction as p_t by looking back 16 previous samples. The current prediction p_t and frame-wise vector $\boldsymbol{f}$, along with predictions in $t-1$ step (omitted in Figure.3.4 for clarity), are the input of a sample rate RNN to estimate the excitation, e_t, which is added to p_t to be s_t. LPCNet claims to operate at 1.6 kbps in real-time with a much better quality against conventional vocoders and much less complexity against pure neural vocoders.

3.2.4 Analysis by synthesis (ABS) in LP codecs

While LP based vocoders synthesize speech signals at a very low bitrate, the quality is usually less satisfying due to audible artifacts. Two major reasons are that:

(a) LPC vocoder depends on a strict voice/unvoice frame classification which can be ambiguous for some frames.

(b) The spectral envelope only matches the magnitude of the spectrogram with phase information being omitted.

Admittedly, human listeners are not as sensitive to phase information which is still related to the overall naturalness of the synthesized speech.

These major limitations of LP based vocoders have been properly addressed in later codecs many of which adopt a technique named analysis-by-synthesis (AbS). The idea of ABS is to embed a decoder during the encoding process to optimize the excitation signal that minimizes the difference between the input signal and the synthesized signal in the time domain via a closed loop, as shown in Figure 3.5. Since the optimization is conducted in the time domain, AbS compensate for the

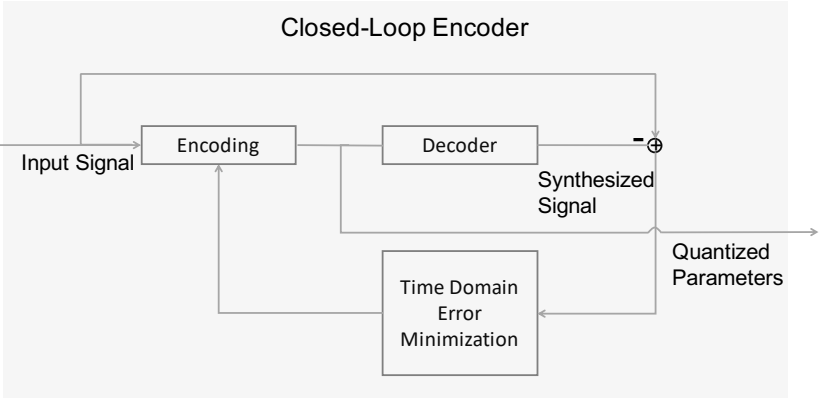


Figure 3.5: A block diagram of the closed-loop encoder with the analysis-by-synthesis approach

loss of phase information modeling, leading to a significant speech quality gain in codecs such as CELP [94] and MELP [95], etc. We note that the benefit from AbS is mainly observed in vocoders, which can be resembled by optimizing residuals with neural codecs in waveform coding.

3.2.5 LPC in speech recognition

Before end-to-end neural networks held sway in automatic speech recognition, speech feature extraction is a critical step to achieve robust ASR performance. Two major features widely employed in ASR are Mel Frequency Cepstral Coefficients (MFCC) derived from Fast Fourier Transform (FFT) based algorithms and resonant modeling coefficients which are acquired from LPC. Compared to LPC, although MFCC is used in many phoneme recognition systems, it is based on FFT with a uniform resolution in the time-frequency space. Consequently, detecting sudden bursts of a "slowly-varying" speech signal can be challenging [96].

Till these days, even with well defined neural engines including but not limited to Connectionist Temporal Classification (CTC) [97] and RNN-Transducer (RNN-T) [98], LPC, along with other features such as MFCC, still serves important auxiliary information in speaker verification and many other ASR tasks [99].

3.3 Trainable LPC Analyzer in Neural Waveform Coding

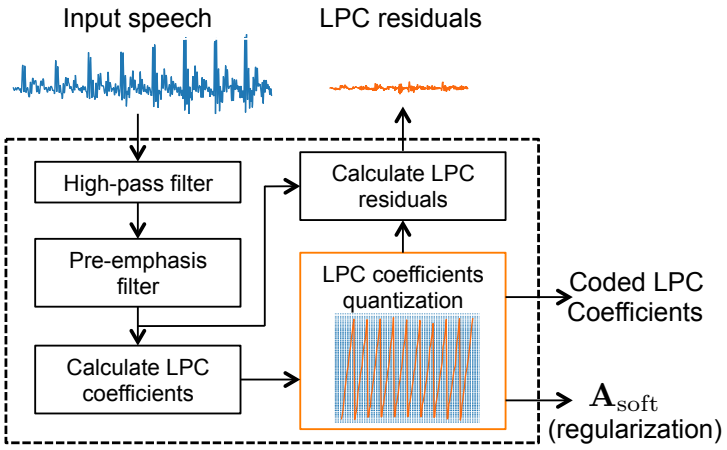


Figure 3.6: The trainable LPC analyzer

A natural way to include LPC in neural speech waveform coding is to take a LPC module off the shelf, such as the one from AMR-WB. Consequently, the LPC module is used to conduct spectral

envelope estimation at a constant rate (usually around 2.4 kbps). The corresponding residual is then quantized and encoded by the neural speech codec introduced in Sec. 4.2.2.

From there, we go one step further to make the LPC module a trainable component which is optimized along with the neural codec for residual signals. After all, it does not make sense to use a deterministic LPC module designed for some other residual coding model in our neural waveform codec. For example, our waveform codec is based on frames, each of which has 512 samples. We argue that the bit allocation between LPC and residual coding networks should be contingent on the content of that frame which can be voiced or unvoiced. Even for voiced frames, the optimal bit allocation may still vary for example, for different vowels which the frame represents.

To achieve a frame-wise dynamic and optimized bit allocation with the LPC component involved, we need to re-model LPC so as to make it "deep-learning" compatible or trainable. Concretely, we conduct frame-wise LPC analysis followed by LPC coefficients conversion and quantization, and then residual calculation, as illustrated respectively.

3.3.1 Waveform pre-processing and framing

We adopt the common pipeline for LPC analysis with several necessary adjustments to make it compatible with neural network computational paradigm.

3.3.1.1 High-pass filtering and pre-emphasizing

Given the input speech waveform, we first adopt high-pass filtering and pre-emphasizing as in [100]. A high-pass filter

$$\boldsymbol{H}_{\text{highpass}} = (\frac{0.989502 - 1.979004z^{-1} + 0.989592z^{-2}}{1 - 1.978882z^{-1} + 0.979126z^{-2}}),$$

is employed with a cut off frequency of 50 Hz. The pre-emphasis filter is defined as a first-order filter

$$\boldsymbol{H}_{\text{emp}}(z) = 1 - \alpha_{\text{emp}} z^{-1},$$

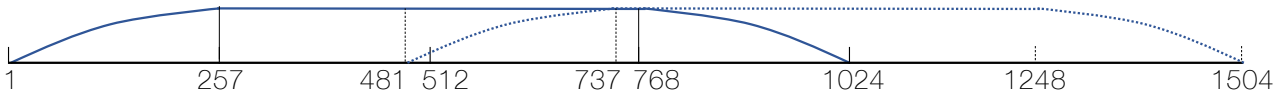


Figure 3.7: LPC windowing schemes: Cross-frame windowing

to boost the high frequency energy which would lead to undesired high spectral tilt, otherwise. The inverse filter, namely the de-emphasis filter is employed to reshape artifacts so as to make them less audible. The free hyperparameter, α_{emp}, can be tuned in multiple ways given a specific sample rate and is chosen to be 0.68 for good results.

3.3.1.2 Data windowing for LPC coefficients calculation

After pre-processing, the speech waveform is segmented to frames, each of which contains 1024 samples. Each frame is windowed before LPC coefficients are calculated. As shown in Figure 3.7, 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. For the prediction of the t-th sample, $\hat{\boldsymbol{x}}[t] = \sum_i a_i \boldsymbol{x}[t-i]$, 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 [11] to calculate LPC coefficients.

3.3.2 Quantization of LPC coefficients

LPC coefficients do not quantize well in that even a small quantization error can lead to noticeable spectral distortion. This is due to several reasons: first, we cannot assume that LPC coefficients are homogeneous as empirically it is observed that higher order coefficients tend to be more sensitive than lower order coefficients; in addition, they do not interpolate easily. Given LPC coefficients at 2 different time steps, the coefficient in between can not be estimated accurately, as shown in Figure 3.8.

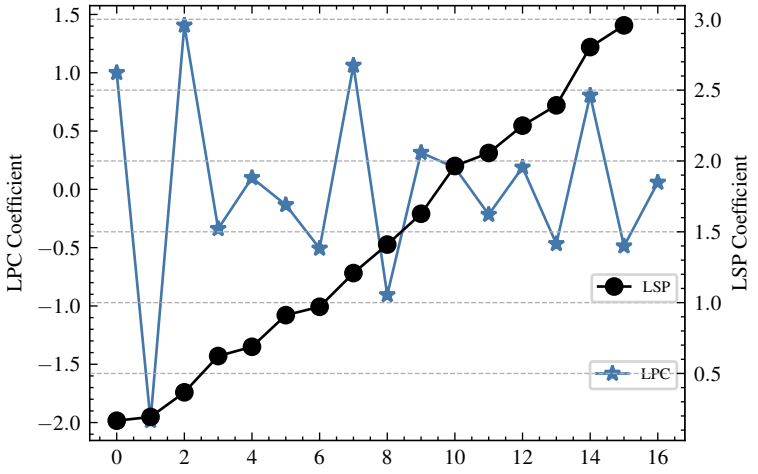


Figure 3.8: An example of the LPC coefficient set with an order of 16, and its LSP representation

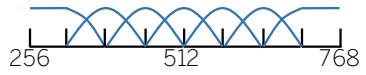
3.3.2.1 Line spectral pairs (LSP)

Efforts have been made to find identical representation of LPC coefficients so as to quantize them more easily. In practice, the line spectral pair (LSP) [101] is the rescue: when LPC coefficients are represented as LSP, they are most robust to quantization.

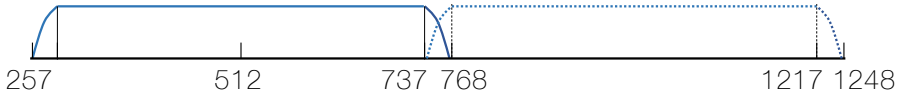
In a nutshell, the LSP algorithm is to find a pair of polynomials (one is palindromic and the other one is antipalindromic) to represent the polynomial of LPC, such that all zeros of LSP are on the unit circle. In fact, the zeroes from the palindromic polynomial and those from antipalindromic polynomial are intertwined on the unit circle, which facilitates the interpolation process (See Figure 3.8: compared to LPC coefficients, in LSP, the coefficients are mono increasing by LPC orders). In addition, since those zeroes are on the unit circle, they are specified uniquely by the angle which corresponds to the LPC frequency response, and therefore LSP is also termed as linear spectral frequencies (LSF).

3.3.2.2 Soft-to-hard quantization for LSP coefficients

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 $\boldsymbol{x}$, $\boldsymbol{h}_{\text{LPC}} = \mathcal{F}_{LPC}(\mathbf{x})$



(a) Sub-frame windowing



(b) Synthesis windowing

Figure 3.9: LPC windowing schemes for residual calculation (a) and synthesis (b)

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. The centroids are initialized via a Gaussian Mixture Model (GMM): for LSP coefficients from each order, 16 clusters are learned (Figure 3.10).

3.3.3 LPC residual calculation

We use a sub-frame windowing technique to calculate residuals (Figure 3.9 (a)). 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 Figure 3.9 (a). 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.

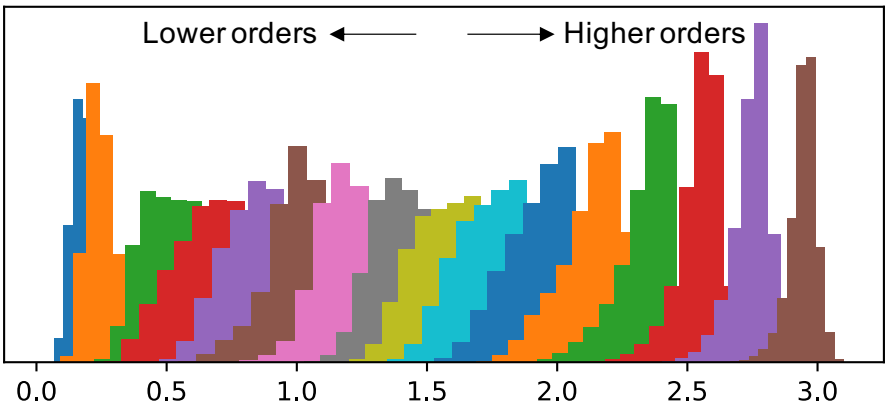


Figure 3.10: Centroid initialization via Gaussian mixture model for the LSP coefficient quantization

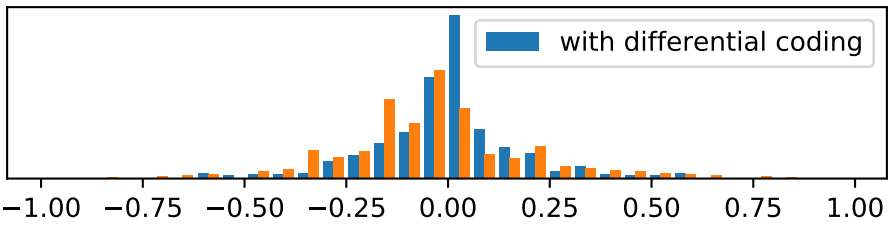


Figure 3.11: Differential coding enables a more centralized distribution

3.3.4 Signal flow

The LPC residual, calculated from the trainable LPC analyzer (Figure 3.6), is compressed by the 1D-CNN AEs. In this work, we employ differential coding [102] 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 (Figure 3.11). As illustrated in Figure 3.12, 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 [103], but to find a pivot which also facilitates the residual compression from following CMRL modules.

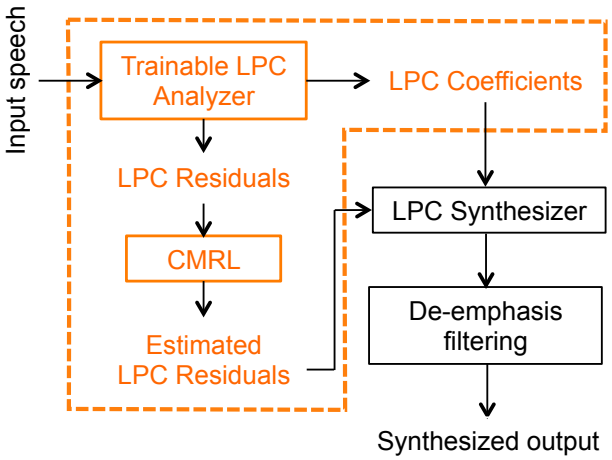


Figure 3.12: Overview of the neural codec with a collaboratively quantized (CQ) LPC analyzer

3.4 Evaluation

3.4.1 Dataset and hyperparameters

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 (Figure 3.7(b)). For 24 kbps, we cascade the LPC module and two AEs as in [43], 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 [104] with the batch size of 128, learning rate of 0.0002 for 30 epochs, followed by the fine-tuning until the entropy is within the target range.

3.4.2 Objective test

Recent work on generative model based speech synthesis systems [26, 44] have reported that PESQ [105] or its successor POLQA [106] 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

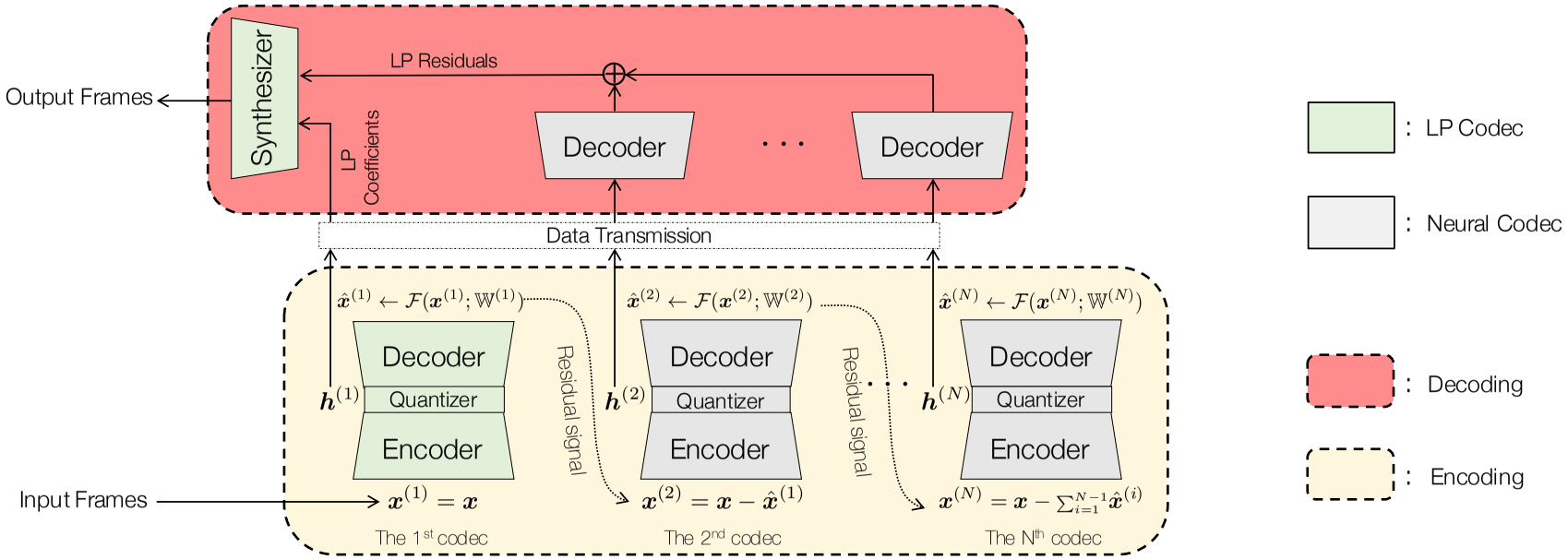


Figure 3.13: Detailed signal flow at test time

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

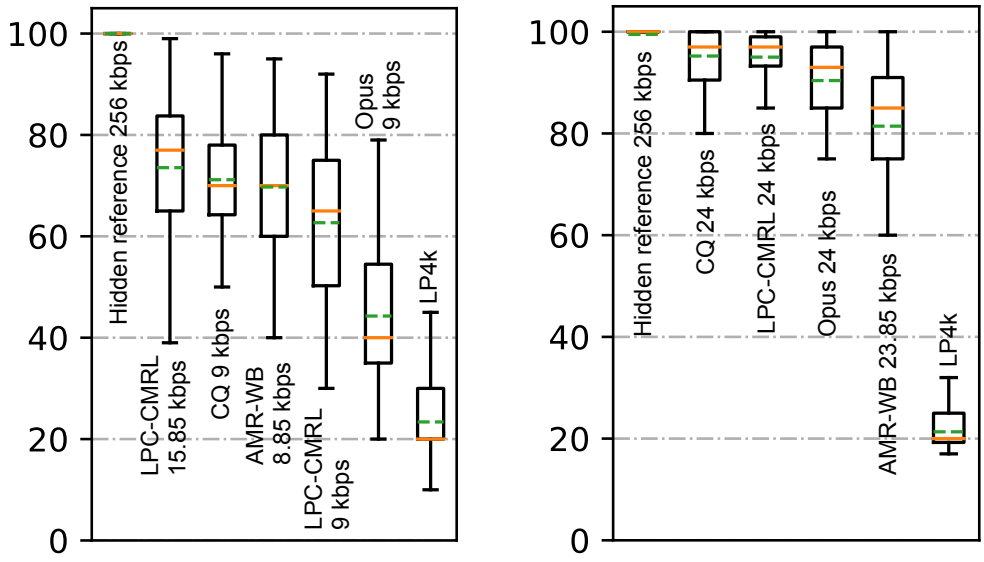
MOS-LQO scores in Table 3.1 as the proposed method is based on waveform reconstruction.

Firstly, consider Opus and AMR-WB both are standardized codecs where AMR-WB is for speech only signals and Opus can be generalized to audio signals. Opus is derived from speech-oriented codec SILK which is based on LPC at 9 kbps and achieves a similar PESQ score compared to AMR-WB. For higher bitrates, Opus uses CELP instead, without relying on the source-filter model, for a more faithful waveform coding. As a consequence, the PESQ score is the highest among all competing models. The fact that using LPC in speech coding does not lead to very high PESQ scores is also observed in our neural codecs. However, with collaborative quantization (CQ), the PESQ score is noticeably improved compared to the model without CQ, especially in lower bitrates.

3.4.3 Subjective test

We conduct two MUSHRA-like [30] 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 [100] at 8.85 kbps, Opus [85] at 9 kbps, LPC-CMRL [43] 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 the low



(a) ∼9 kbps

(b) ∼24 kbps

Figure 3.14: MUSHRA results in box-plots (Orange solid lines represent medians, and green dashed lines represent means).

bitrate setting (Fig. 3.14). In the 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 yields 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 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[2].

[2]Decoded samples are available at `http://kaizhen.us/speechcoding`

3.4.4 Extended experiments with the 0.35-million codec

To further lower the model size, we replace the neural waveform codec with the one including only 0.35 million parameters as introduced in Chapter. 2. Based on that compact neural codec, we consider following neural waveform coding models at 3 bitrate modes: 12kbps for the low-range setup, 20kbps for the mid-range setup, and 32kbps for the high-range setup.

- Model-I: the neural codec with 0.35 million parameters as a baseline;
- Model-II: the baseline with a non-trainable LPC codec;
- Model-III: the baseline with a trainable LPC codec;
- Model-IV: Model-III with CMRL which is evaluated in 32 kbps for performance scaling.

Similarly, regarding the standard codecs, AMR-WB [20] and Opus [21] are considered for comparison.

We firstly compare all models with respect to objective measures, while being aware of the inconsistency between these surrogate measures and the subjective score. We then evaluate these codecs in two rounds of MUSHRA-like subjective listening tests: the neural codecs are compared in the first round; the winner is compared with other standard codecs in the second round.

Table.3.2 reports Signal-to-Noise Ratio (SNR) and PESQ-WB [28] from all considered waveform coding systems. AMR-WB in the low-range bitrate setting operates at 12.65 kbps and 23.05 kbps for the mid-range. Among neural codecs, for both SNR and PESQ, simply Model-I outperforms others in all three bitrate setups. Even compared with AMR-WB and Opus, Model-I is the winner except for the low bitrate case where Opus achieves the highest PESQ score. It is also observed that with CQ, Model-III gains higher SNR and PESQ scores especially when the bitrate gets lower; and that the performance scales up noticeably for Model-IV with 2 neural codecs cascading residuals in CMRL.

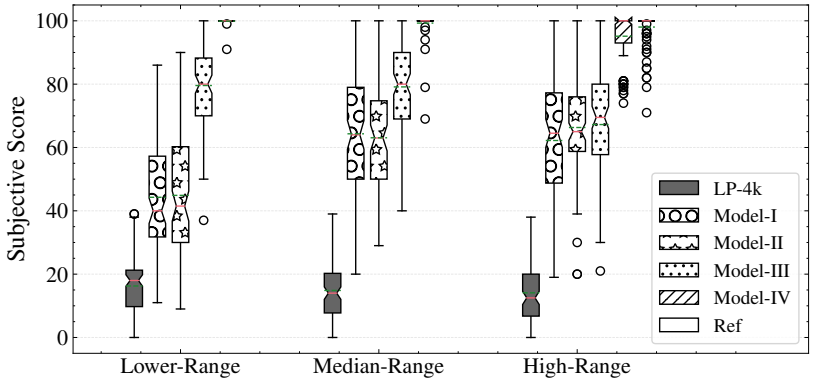
Table 3.2: Objective measurements for neural codec comparison under three bitrate (kbps) cases.

Bitrate	SNR (dB)						PESQ-WB					
	Model-I	Model-II	Model-III	Model-IV	AMR-WB	Opus	Model-I	Model-II	Model-III	Model-IV	AMR-WB	Opus
12	**12.37**	10.69	10.85	–	11.60	9.63	3.67	3.45	3.60	–	3.92	**3.93**
20	**16.87**	10.73	13.65	–	13.14	9.46	**4.37**	3.95	4.01	–	4.18	**4.37**
32	**20.24**	11.84	14.46	17.11	–	17.66	**4.42**	4.15	4.18	4.35	–	4.38

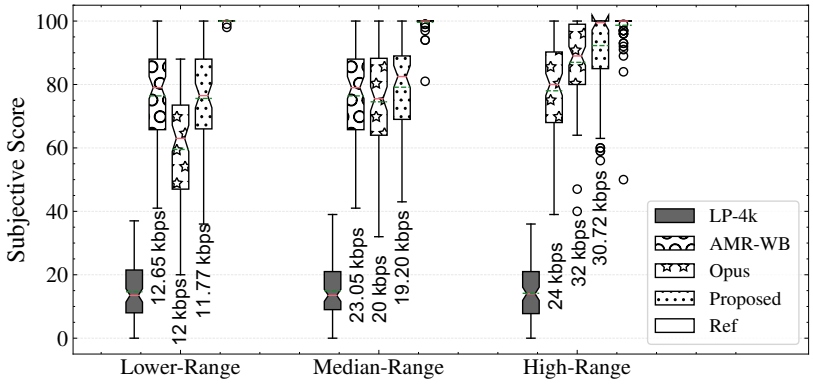
Perhaps, we would have stopped proposing other models if the superior objective score could faithfully indicate higher speech quality. However, this is barely the case. Admittedly, there have been efforts towards a better alignment to human auditory perception [107][108] by improving the model's "perceptual salience". Unfortunately, when it comes to supervised learning based neural waveform coding, we have to rely on "objective assessment" to approach "subjective satisfaction". Therefore, we still report results in Table.3.2 for future reference.

The subjective scores are rendered in Figure 3.15 as boxplots. Each box ranges from the 25% percentile to 75% percentile with a 95% confidence interval. The mean and median are presented as the green dotted line and pink hard line, respectively. Outliers are included in circles as well.

In a nutshell, as shown in Figure 3.15 (a), with a jointly trained LPC codec, Model-III is much more preferred than the pure end-to-end Model-I and Model-II with the non-trainable LPC codec. The advantage gets more significant for lower bitrates, which is quite contradictory to the observation in Table.3.2 where Model-I is mostly favored. The insight is indicated in Figure 3.17 where both the decoded and artifact signals are presented. Comparing Figure 3.17 (g) and (i), the artifact of Model-III is more structured than that of Model-I: the artifact is mainly accumulated where the reference signal is strong and rather soft, otherwise. In contrast, the artifact of Model-I is quite blurry even when the bitrate is up to 32 kbps. Compared with Model-II, the jointly trained LPC codec yields LP residuals which are more robust to quantization distortion especially when the entropy decreases (Figure.3.16). Another observation is that the performance for Model-III at 32 kbps is not better than that at 20 kbps, but rather dwarfed by Model-IV. This clearly shows the effectiveness of CMRL on performance scaling.



(a) Neural waveform codecs comparison



(b) Comparison between proposed methods and standard codecs

Figure 3.15: MUSHRA-like subjective listening test results.

For the lower-range bitrate case, our model is on a par with AMR-WB, both of which outperform Opus. The major concern for Opus is the loss in high frequencies. In the median range, our model at 20 kbps is comparable to Opus at the same bitrate and AMR-WB at 23.05 kbps. Our model is superior in high bitrate cases, compared to Opus at 32 kbps. As a sanity check, AMR-WB outperforms Opus at 12.65 kbps; more outliers from the hidden reference are detected for higher bitrates as decoded samples sound more similar to the hidden reference.

3.4.5 Ablation analysis on the blending weights

Now that we have shown the effectiveness of CQ and CMRL for speech quality improvement, we investigate how different blending ratios between loss terms can alter the performance. The training loss function contains two major terms aside from the regularizers. The MSE term holds

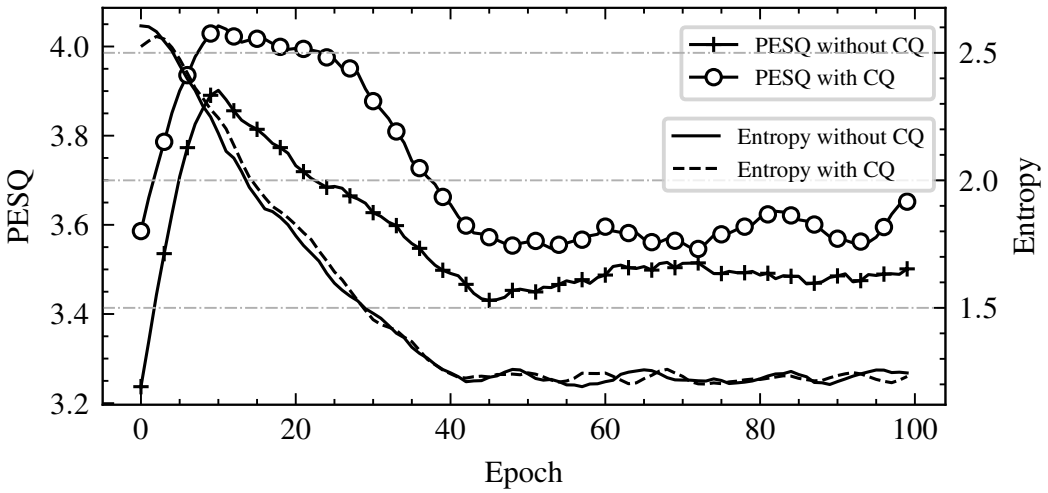
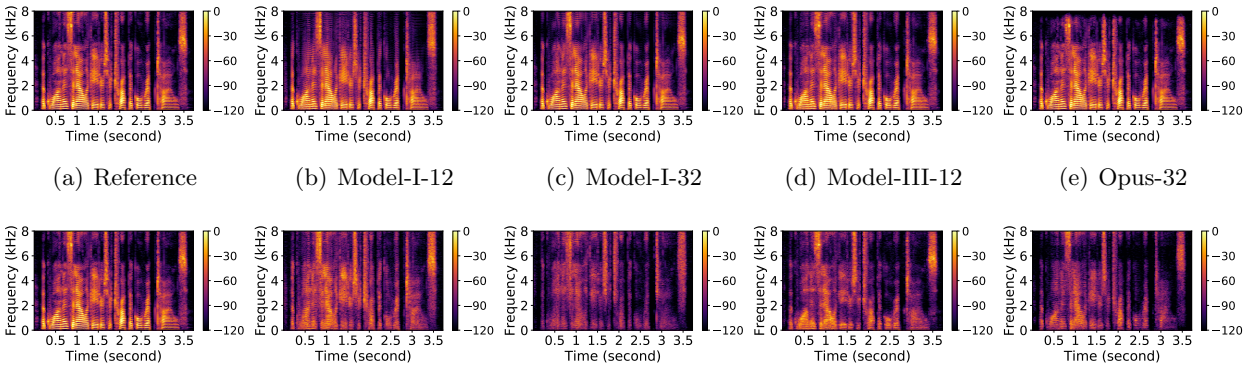


Figure 3.16: Ablation analysis on CQ: with CQ, our neural codec shows less performance degradation (measured in PESQ from the validation data) during training to lower the bitrate.



(a) Reference (b) Model-I-12 (c) Model-I-32 (d) Model-III-12 (e) Opus-32

(f) Reference (g) Model-I-12 (h) Model-I-32 (i) Model-III-12 (j) Opus-32

Figure 3.17: Spectrogram comparison: (b) - (e) are from decoded signals and (g) - (j) are from the corresponding artifact signals. All spectrograms use the Hann window with the size of 1024 and 50% overlap.

sway in the end-to-end neural waveform coding system, while the Mel-scaled loss prioritizes certain frequency bands over the others. In Table.3.3 (a), the SNR reaches the highest when there is only the MSE term, which leads to the lowest PESQ score. By only keeping the Mel-scaled loss term, the PESQ score is decent while there is no alignment between the decoded signal and the reference, as suggested by the SNR. Similarly in Table.3.3 (b) where the input of the neural codec is LP residuals, the single MSE loss term yields the highest SNR for residual reconstruction, which, however, does not benefit the synthesized signal even in terms of SNR. Although the highest SNR

(a) Neural codec only

Blending Ratio	Waveform SNR (dB)	Waveform PESQ
1:0	**18.12**	3.67
0:1	0.16	4.23
1:1	6.23	4.31
10:1	16.88	**4.37**

(b) Collaboratively trained LPC codec and neural codec

Blending Ratio	Residual SNR (dB)	Waveform SNR (dB)	Waveform PESQ
1:0	**9.73**	14.25	3.84
0:1	1.79	17.23	4.02
1:1	7.11	**17.82**	**4.08**
10:1	8.26	17.55	4.01

Table 3.3: Ablation analysis on blending weights

and PESQ are from the blending ratio of 1:1, for simplicity and consistency, all neural codecs in objective and subjective tests are trained via the blending ratio of 10:1.

3.4.6 Frame-wise bit allocation

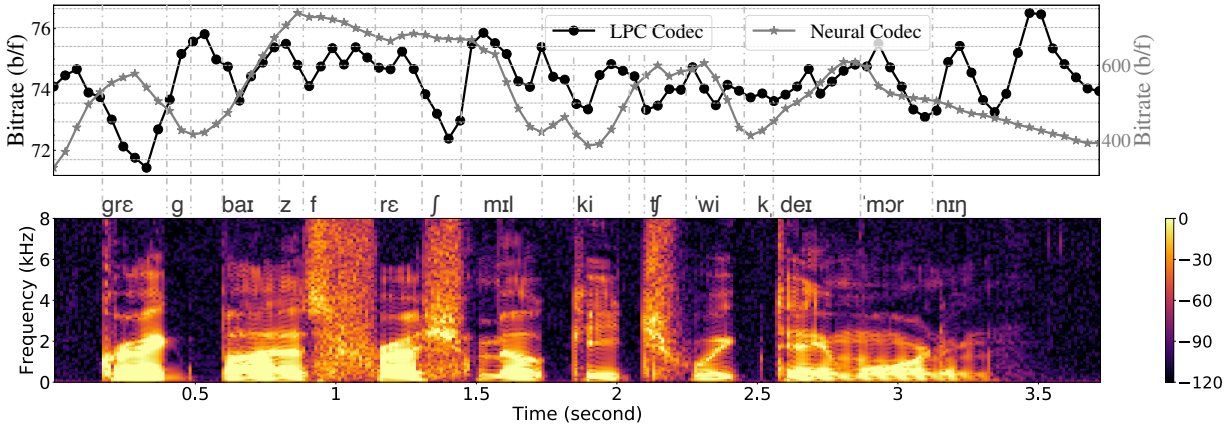


Figure 3.18: Frame-wise bit allocation analysis

In addition, we show the frame-wise bit allocation between the LPC codec and neural codec at mid-range bitrate setting in Figure 3.18. The superimposed plot is for the amount of bits for each frame (b/f) from either the LP coefficient compression (in grey color) and residual compression (in orange color). In terms of the residual quantization and encoding, we observe that the bitrate is proportional to the energy of the corresponding frames. Those consumes more bits are fricatives, such as "f" and "$\int$", and affricates such as "$t\int$", both of which are quasi-periodic. For near silent

frames, LPC codec consumes more than average bits while the neural codec achieves a higher coding efficiency. In other words, the LPC codec versus neural codec bit ratio is relatively large when the corresponding frame has low energy. However, it still requires a significant amount of bits to even represent those near silent frames, which can be further optimized.

3.4.7 Complexity analysis

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.

3.4.8 Discrepancy between subjective scores and surrogate measures

As observed in the objective test and subjective test, the pain lies in the discrepancy between them: our neural network is only possible to be trained when the loss function is defined. However, the loss function does not faithfully reflect upon the auditory perception of sound in human ears. In other words, the model that delivers the best decoded samples measured in terms of SNR or PESQ may not be favored by listeners. Although, multiple loss terms such as scale invariant signal-to-noise ratio (Si-SNR) and perceptual metric for speech quality evaluation (PMSQE) are proposed, they are still far from the actual mean opinion scores (MOS) but only better objective measures more correlated to MOS.

3.5 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: PSYCHOACOUSTIC CALIBRATION

4.1 Motivation

Audio coding, a fundamental set of technologies in data storage and communication, compresses the original signal into a bitstream with a minimal bitrate (encoding) without sacrificing the perceptual quality of the recovered waveform (decoding) [109, 22]. In this paper we focus on the lossy codecs, which typically allow information loss during the process of encoding and decoding only in inaudible audio components. To this end, psychoacoustics is employed to quantify the audibility in both time and frequency domains. For example, MPEG-1 Audio Layer III (also known as MP3), as a successful commercial audio codec, achieves a near-transparent quality at 128 kbps by using a psychoacoustic model (PAM) [22]. Its bit allocation scheme determines the number of bits allocated to each subband by dynamically computing the masking threshold via a PAM and then allowing quantization error once it is under the threshold [110].

Recent efforts on deep neural network-based speech coding systems have made substantial progress on the coding gain [111, 24, 112]. They formulate coding as a complex learning process that converts an input to a compact hidden representation. This poses concerns for edge applications with the computational resource at a premium: a basic U-Net audio codec contains approximately 10 million parameters [113]; in [88], vector quantized variational autoencoders (VQ-VAE) [40] employs WaveNet [24] as a decoder, yielding a competitive speech quality at 1.6 kbps, but with 20 million parameters. In addition, recent neural speech synthesizers employ traditional DSP techniques, e.g., linear predictive coding (LPC), to reduce its complexity [26]. Although it can serve as a decoder of a speech codec, LPC does not generalize well to non-speech signals.

Perceptually meaningful objective functions have shown an improved trade-off between perfor-

mance and efficiency. Some recent speech enhancement models successfully employed perceptually inspired objective metrics, e.g. perceptual attractors [114], energy-based weighting [115], perceptual weighting filters from speech coding [116], and global masking thresholds [117] [118], while they have not targeted audio coding and model compression. Other neural speech enhancement systems implement short-time objective intelligibility (STOI) [27] and perceptual evaluation of speech quality (PESQ) [119] as the loss [120, 121]. These metrics may benefit speech codecs, but do not faithfully correlate with subjective audio quality. Meanwhile, PAM serves as a subjectively salient quantifier for the sound quality and is pervasively used in the standard audio codecs. However, integrating the prior knowledge from PAM into optimizing neural audio codecs has not been explored.

In this paper, we present a psychoacoustic calibration scheme to improve the neural network optimization process, as an attempt towards efficient and high-fidelity neural audio coding (NAC) [122]. With the global masking threshold calculated from a well-known PAM [123], the scheme firstly conducts priority weighting making the optimization process focus more on audible coding artifacts in frequency subbands with the relatively weaker masking effect, while going easy otherwise. The scheme additionally modulates the coding artifact to ensure that it is below the global masking threshold, which is analogous to the bit allocation algorithm in MP3 [22]. Experimental results show that the proposed model outperforms the baseline neural codec twice as large and consumes 23.4% more bits per second. With the proposed method, a lightweight neural codec, with only 0.9 million parameters, performs near-transparent audio coding comparable with the commercial MPEG-1 Audio Layer III codec at 112 kbps. This is, to our best knowledge, the first method to directly incorporate psychoacoustics to neural audio coding.

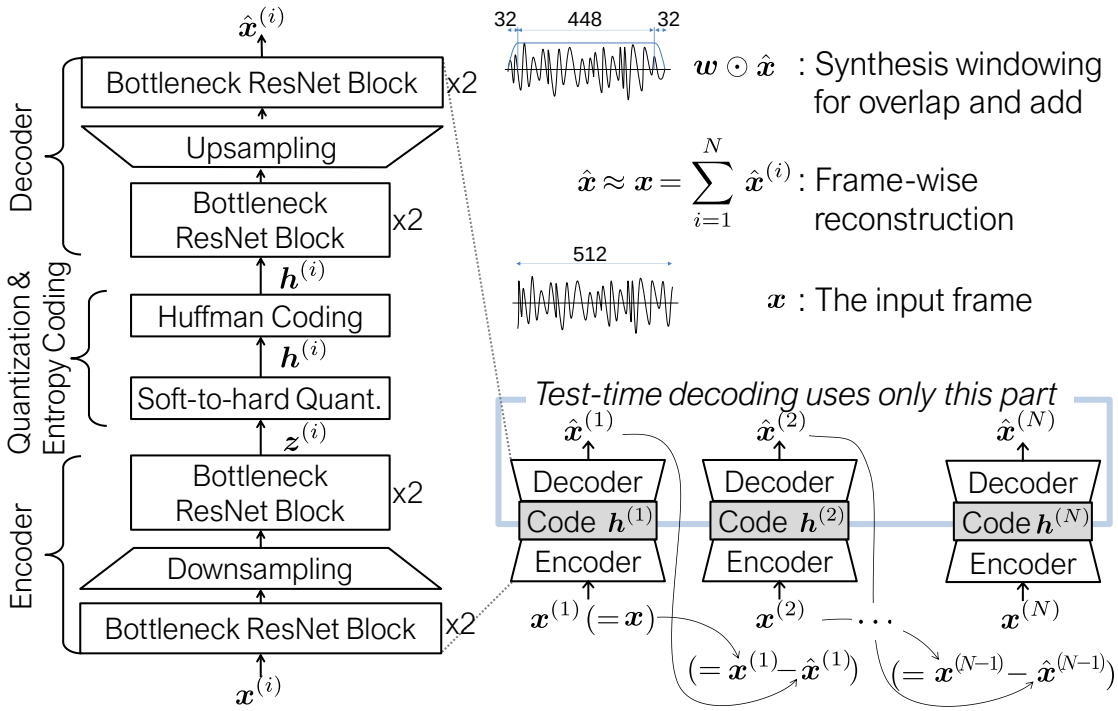


Figure 4.1: Schematic diagrams for NAC. The residual coding pipeline for CMRL consists of multiple NAC autoencoding modules. Training and test-time encoding uses all blocks while the test-time decoding uses only the decoder portion.

4.2 End-to-End Neural Audio Codec

4.2.1 Lightweight NAC module

Given that neural codecs can suffer from a large inference cost due to their high model complexity, one of our goals is to demonstrate the advantage of the proposed pychoacoustic loss function on model compression. To that end, we choose a compact neural audio coding (NAC) module as the building block. The NAC module is a simplified version of a convolutional neural network (CNN)-based autoencoder [42] with only 450K parameters. As shown in Figure 4.1, it consists of a stack of bottleneck blocks as in [124], each of which performs a ResNet-style residual coding [33]. The code vector produced by its encoder part is discretized into a bitstring via the soft-to-hard quantization process originally proposed in [41] for image compression. We detail the description as follows.

4.2.1.1 Encoder

The CNN encoder maps an input frame of T time-domain samples, $\boldsymbol{x} \in \mathbb{R}^T$ to the code vector, i.e., $\boldsymbol{z} \leftarrow \mathcal{F}_{\text{enc}}(\boldsymbol{x})$. Striding during the 1D convolution operation can downsample the feature map. For example, $\boldsymbol{z} \in \mathbb{R}^{T/2}$ when the stride is set to be 2 and applied once during encoding.

4.2.1.2 Soft-to-hard quantization

Quantization replaces each real-valued element of the code vector $\boldsymbol{z}$ with a kernel value chosen from a set of K representatives. We use soft-to-hard quantizer [41], a clustering algorithm compatible with neural networks, where the representatives are also trainable. During training, in each feedforward routine, the c-th code value z_c is assigned to the nearest kernel out of K, $\boldsymbol{b} \in \mathbb{R}^K$, which have been trained so far. The discrepancy between z_c and the chosen kernel $h_c \in \{b_1, b_2, \cdots, b_K\}$ (namely the quantization error) is accumulated in the final loss, and then reduced during training via backpropagation (i.e., by updating the means and assignments). Specifically, the cluster assignment is conducted by calculating the distance, $\boldsymbol{d} \in \mathbb{R}^K$, between the code value and all kernels, and then applying the softmax function to the negatively scaled distance to produce a probabilistic membership assignment: $\boldsymbol{a} \leftarrow \text{softmax}(-\alpha \boldsymbol{d})$. Although we eventually need a hard assignment vector $\mathbf{a}$, i.e., a one-hot vector that indicates the closest kernel, during training the quantized code $\boldsymbol{h}$ is acquired by a soft assignment, $\boldsymbol{a}^\top \boldsymbol{b}$, for differentiability. Hence, at the test time, $\mathbf{a}$ replaces $\boldsymbol{a}$ by turning on only the maximum element. Note that a larger scaling factor α makes $\boldsymbol{a}$ harder, making it more similar to $\mathbf{a}$. Huffman coding follows to generate the final bitstream.

4.2.1.3 Decoder

The decoder recovers the original signal from the quantized code vector: $\hat{\boldsymbol{x}} = \mathcal{F}_{\text{dec}}(\boldsymbol{h})$, by using an architecture mirroring that of the encoder. For upsampling, we use a sub-pixel convolutional layer proposed in [125] to recover the original frame length T.

4.2.1.4 Bitrate analysis and control

The lower bound of the bitrate is defined as $|\boldsymbol{h}|\mathcal{H}(\boldsymbol{h})$, where $|\boldsymbol{h}|$ is the number of down-sampled and quantized features per second. The entropy $\mathcal{H}(\boldsymbol{h})$ forms the lower bound of the average amount of bits per feature. While $|\boldsymbol{h}|$ is a constant given a fixed sampling rate and network topology, $\mathcal{H}(\boldsymbol{h})$ is adaptable during training. As detailed in [41], basic information theory calculates $\mathcal{H}(\boldsymbol{h})$ as $-\sum_k p(b_k)\log_2 p(b_k)$, where $p(b_k)$ denotes the occurrence probability of the k-th cluster defined in the soft-to-hard quantization. Therefore, during model training, $\mathcal{H}(\boldsymbol{h})$ is added to the loss function as a regularizer navigating the model towards the target bitrate. Initiated as 0.0, the blending weight increases by 0.015 if the actual bitrate overshoots the target and decreases by that amount otherwise. Because this regularizer is well defined in the literature [41][42][43], we omit it in following sections for simplicity purposes.

4.2.2 NAC cascading

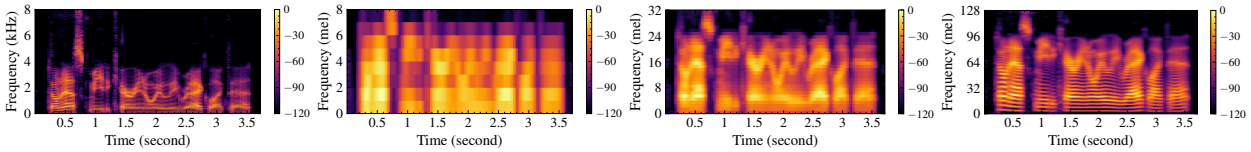
To scale up for high bitrates, cross-module residual learning (CMRL) [43, 126] implants the multistage quantization scheme [127] by cascading residual coding blocks (Figure 4.1). CMRL decentralizes the neural autoencoding effort to a chain of serialized low complexity coding modules, with the input of i-th module being $\boldsymbol{x}^{(i)} = \boldsymbol{x} - \sum_{j=1}^{i-1} \hat{\boldsymbol{x}}^{(j)}$. That said, each module only encodes what is not reconstructed from preceding modules, making the system scalable. Concretely, for an input signal $\boldsymbol{x}$, the encoding process runs all N autoencoder modules in a sequential order, which yields the bitstring as a concatenation of the quantized code vectors: $\boldsymbol{h} = \left[\boldsymbol{h}^{(1)^\top}, \boldsymbol{h}^{(2)^\top}, \cdots, \boldsymbol{h}^{(N)^\top}\right]^\top$. During decoding, all decoders, $\mathcal{F}_{\text{dec}}(\boldsymbol{h}^{(i)}) \quad \forall i$, run to produce the reconstructions that sum up to approximate the initial input signal as $\sum_{i=1}^{N} \hat{\boldsymbol{x}}^{(i)}$.

4.3 Psychoacoustic Calibration in Neural Audio Codec

4.3.1 "Perceptual" loss in the frequency domain

It is obvious that a loss function defined by mean squared error (MSE) or sum of squared error (SSE) in the time domain is not optimal, although these time domain loss terms are used in end-to-end audio coding neural networks. In this work, the baseline model uses the SSE defined in the time domain: $\mathcal{L}_1(\boldsymbol{x}||\hat{\boldsymbol{x}}) = \sum_{i=1}^{N}\sum_{t=1}^{T}\left(\hat{x}_t^{(i)} - s_t^{(i)}\right)^2$.

In addition, it is in fact a common practice to measure and compare the energy in the time-frequency domain with a perceptual scale [128]. Two widely used perceptual scales are mel scale [129] and bark scale [130], both are to warp the frequency space for the gain of perceptual salience. Therefore, we also consider the loss defined in the mel-scaled frequency domain to weigh more on the low frequency area, as the human auditory system does, $\mathcal{L}_2(\boldsymbol{y}||\hat{\boldsymbol{y}}) = \sum_{i=1}^{N}\sum_{l=1}^{L}\left(y_l^{(i)} - \hat{y}_l^{(i)}\right)^2$, where $\boldsymbol{y}$ stands for a mel spectrum with L frequency subbands. The effect to warp the frequency space in mel scale is shown in Figure 4.2: the frequency space in mel scale accents lower frequency bands which usually carry more useful speech-related information. Besides, a coarse-to-fine mel-scale frequency analysis can be conducted by choosing smaller filter bank size such as 8 and larger size such as 128.



(a) Reference (b) 8 mel filters (c) 32 mel filters (d) 128 mel filters

Figure 4.2: Spectrograms (b) - (d) in mel scale discard more information in higher frequencies while preserving more information in lower frequencies. The frequency resolution increases when there are more mel filters.

Without loss of generality, we choose a basic psychoacoustic model (PAM) that computes simultatenous masking effects for the input signal as a function of frequency, while the temporal masking effect is not integrated. According to PAM-1 defined in [22], for an input frame, it (a)

calculates the logarithmic power spectral density (PSD) p; (b) detects tonal and noise maskers, followed by decimation; (c) calculates the masking threshold for individual tonal and noise maskers $\boldsymbol{U} \in \mathbb{R}^{F\times R}, \boldsymbol{V} \in \mathbb{R}^{F\times B}$, where R and B are the number of maskers. The global masking threshold at frequency bin f is accumulated from each individual masker in (c) along with the absolute hearing threshold $\boldsymbol{Q}$ [123], as $m_f = 10\log_{10}\left(10^{0.1Q_f} + \sum_r 10^{0.1U_{f,r}} + \sum_b 10^{0.1V_{f,b}}\right)$. The procedure is detailed in the following section.

4.3.2 Global masking threshold calculation in PAM-I

4.3.2.1 Spectral analysis and SPL normalization

The first step is to perform time-frequency analysis on the input signal. For a standard CD quality pop music sampled at 44.1 kHz with 16 bits per sample, the PSD estimation is conducted via a 512-point fast Fourier transform (FFT) with the $(\frac{1}{16})^{\text{th}}$ overlapped Hann window. p is then normalized to the sound pressure level (SPL) in equation 4.1, where the power normalization term PN is fixed at 90.302 dB.

$$p_k = PN + 10\log_{10}|\text{STFT}(k)|^2 \tag{4.1}$$

4.3.2.2 Tonal and noise maskers identification

Once the PSD is normalized, tonal maskers (TM) are then detected by finding local maximum values of p. Concretely, tones are defined as local maxima exceeding their neighbours by at least 7 dB (equation 4.2).

$$S_T = \left\{ p_k \middle| \begin{array}{l} p_k > p_{k\pm 1}, \\ \\ p_k > p_{k\pm\Delta(k)} + 7, \end{array} \right\} \tag{4.2}$$

where S_T denotes the tonal set and

$$\Delta(k) \in \begin{Bmatrix} 2, & 2 < k < 352 & ,(0.03 - 5.5 \text{ kHz}) \\ [2,3], & 352 < k < 512 & ,(5.5 - 8 \text{ kHz}). \end{Bmatrix} \tag{4.3}$$

The tonal masker (U) is then calculated by combining energy from three adjacent spectral components around the peak as in equation 4.4.

$$U_k = 10 \log_{10} \sum_{j=-1}^{1} 10^{0.1 p_{k+j}} \tag{4.4}$$

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 (V). 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 equation 4.5, where $\bar{k}$ is the geometric mean of the critical band, $(\prod_{j=l}^{u} j)^{1/(l-u+1)}$.

$$V_{\bar{k}} = 10 \log_{10} \sum_{j} 10^{0.1 p_j}, \forall p_j \notin \{U_k, U_{k\pm1}, U_{k\pm\Delta_k}\} \tag{4.5}$$

Not all tonal and noise maskers are needed. If the masker is below the absolute hearing threshold, it will be dropped. Additionally, if more than one maskers are detected within a sliding window of 0.5 Bark bandwidth, only the strongest one will be preserved.

4.3.2.3 Individual masking thresholds

The individual masking threshold of a tonal masker is specified in equation 4.6

$$U_{f,r} = U_r - 0.275 z(r) + \text{SF}(f, r) - 6.025, \tag{4.6}$$

where U_r denotes SPL of the tonal masker at the frequency bin r, $z(r)$ is the Bark frequency of the r-th bin, and $\text{SF}(i, j)$ is a spreading function modeling the spread of masking from f-th bin to

r-th bin, equation 4.7.

$$\mathrm{SF}(f,r) = \begin{cases} 17\Delta_z - 0.4U_r + 11, -3 \leq \Delta_z < -1 \\ (0.4U_r + 6)\Delta_z, -1 \leq \Delta_z < 0 \\ -17\Delta_z, 0 \leq \Delta_z < 1 \\ (0.15U_r - 17)\Delta_z - 0.15U_r, 1 \leq \Delta_z < 8, \end{cases} \tag{4.7}$$

where $\Delta_z = z(f) - z(r)$. The piece-wise function (4.7) echoes the fact that the frequency selectivity of human ears decreases towards higher frequencies. For simplicity, the spread of masking is constrained to a 10-Bark neighborhood [123].

Likewise, the masking threshold from the b-th noise masker at the f-th frequency bin is given in equation 4.8

$$V_{f,b} = V_b - 0.175z(b) + \mathrm{SF}(f,b) - 2.025. \tag{4.8}$$

4.3.2.4 Global masking threshold

The global masking threshold () combines individual masking curves from the tonal and noise maskers, along with the absolute threshold of hearing in equation 4.9

$$m_f = 10\log_{10}\left(10^{0.1Q_f} + \sum_r 10^{0.1U_{f,r}} + \sum_b 10^{0.1V_{f,b}}\right), \tag{4.9}$$

where Q_f is the absolute hearing threshold at f-th frequency bin, $U_{f,r}$ and $U_{f,b}$ are individual masking thresholds of r-th tonal masker and b-th noise masker at f-th frequency bin, respectively. Fig. 4.3 shows an example of p of a signal and its global masking threshold based on the simultaneous masking effect.

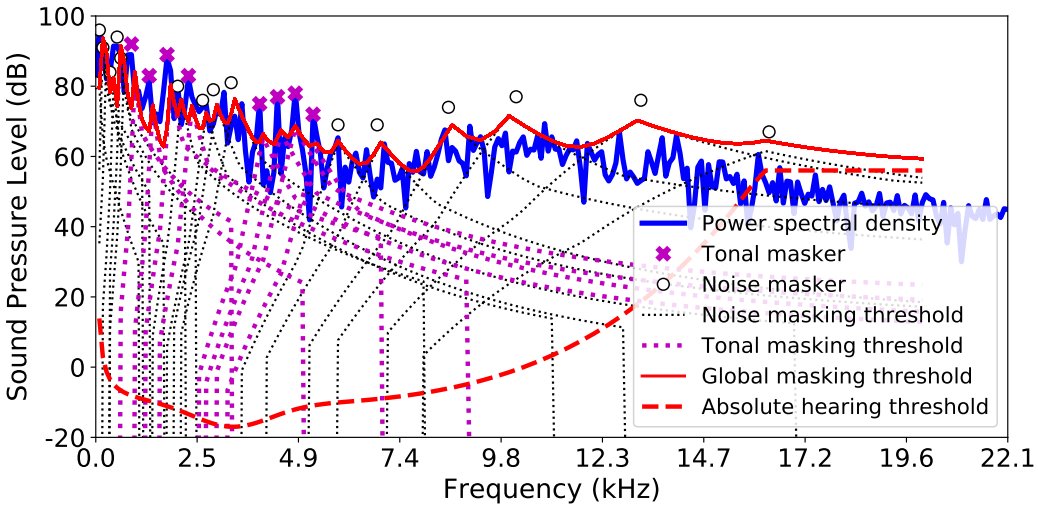


Figure 4.3: Visualization of the masker detection, individual and global masking threshold calculation for an audio input

4.3.3 Signal-to-mask ratio and mask-to-noise ratio

Signal-to-mask ratio (SMR) is the difference between two curves: $p-m$. Typically, the bit allocation algorithm calculates the mask-to-noise ratio (MNR) by subtracting SMR from Signal-to-Noise Ratio (SNR), where noise is 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 [131, 132, 133]. 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.

Global masking threshold as discussed is used in various conventional audio codecs to allocate minimal amount of bits without losing the perceptual audio quality. Typically, the bit allocation algorithm optimizes n_f/m_f (NMR), where n_f denotes the power of the noise (i.e., coding artifacts) in the subband f and m_f is the power of the global masking threshold. In an iterative process, each time the bit is assigned to the subband with the highest NMR until no more bit can be allocated

[131, 132, 133]. The global masking curve acquired via PAM-1 comprises both input-invariant prior knowledge as in the absolute hearing threshold and input-dependent masking effects. We propose two mechanisms to integrate PAM-1 into NAC optimization: priority weighting and noise modulation.

4.3.4 Priority weighting

During training we estimate the logarithmic PSD p out of an input frame $\boldsymbol{x}$, as well the global masking threshold to define a perceptual weight vector, $\boldsymbol{w} = \log_{10}(\frac{10^{0.1p}}{10^{0.1}} + 1)$: 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:

$$\mathcal{L}_3(\boldsymbol{x}||\hat{\boldsymbol{x}}) = \sum_i \sum_f w_f \Big(x_f^{(i)} - \hat{x}_f^{(i)} \Big)^2, \tag{4.10}$$

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 bin, i.e. $p_f >_f$, 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 some reconstruction error. The weights are bounded between 0 and ∞, whose smaller extreme happens if, for example, the masking threshold is too large compared to the sufficiently soft signal.

4.3.5 Noise modulation

The priority weighting mechanism can accidentally result in audible reconstruction noise, exceeding the mask value m_f, when w_f is small. Our second psychoacoustic loss term is to modulate the reconstruction noise by directly exploiting NMR, n_f/m_f, where $\boldsymbol{n}$ is the power spectrum of the reconstruction error $\boldsymbol{x} - \sum_{i=1}^{N} \hat{\boldsymbol{x}}^{(i)}$ from all N autoencoding modules. We tweak the greedy bit

allocation process in the MP3 encoder that minimizes NMR iteratively, such that it is compatible to the stochastic gradient descent algorithm as follows:

$$\mathcal{L}_4 = \max_f \left(\text{ReLU} \left(\frac{n_f}{m_f} - 1 \right) \right). \tag{4.11}$$

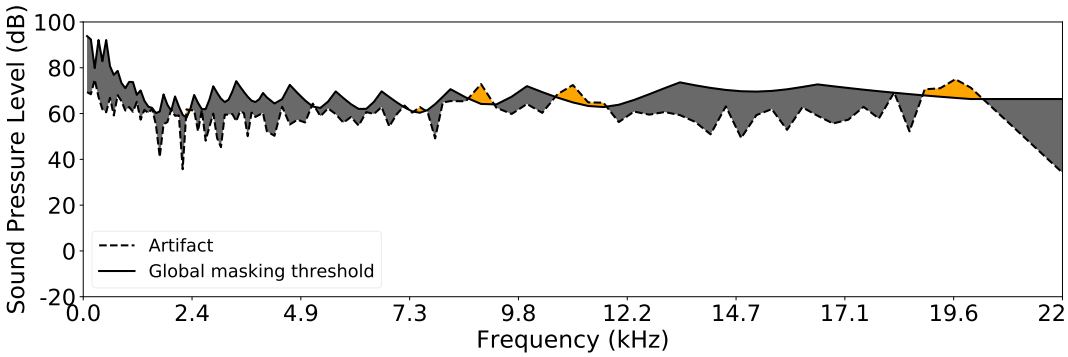
The rectified linear units (ReLU) function excludes the contribution of the inaudible noise to the loss when $n_f/m_f - 1 < 0$. Out of those frequency bins where the noise is audible, the max operator selects the one with the largest NMR, which counts towards the total loss. The process as such resembles MP3's bit allocation algorithm, as it tackles the frequency bin with the largest NMR for each training iteration.

4.4 Evaluation

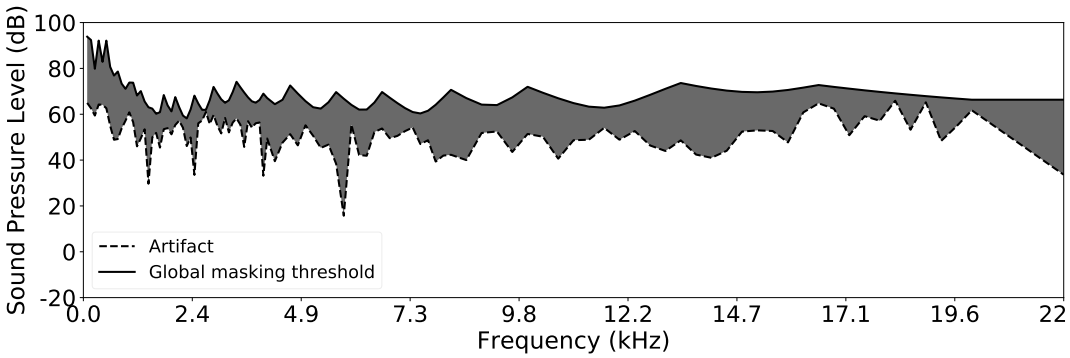
4.4.1 Experimental setup

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 [104], 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). Noise can exceed the mask (orange).



(b) With noise modulation (Model-D)

Figure 4.4: The effect of the proposed noise modulation loss

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

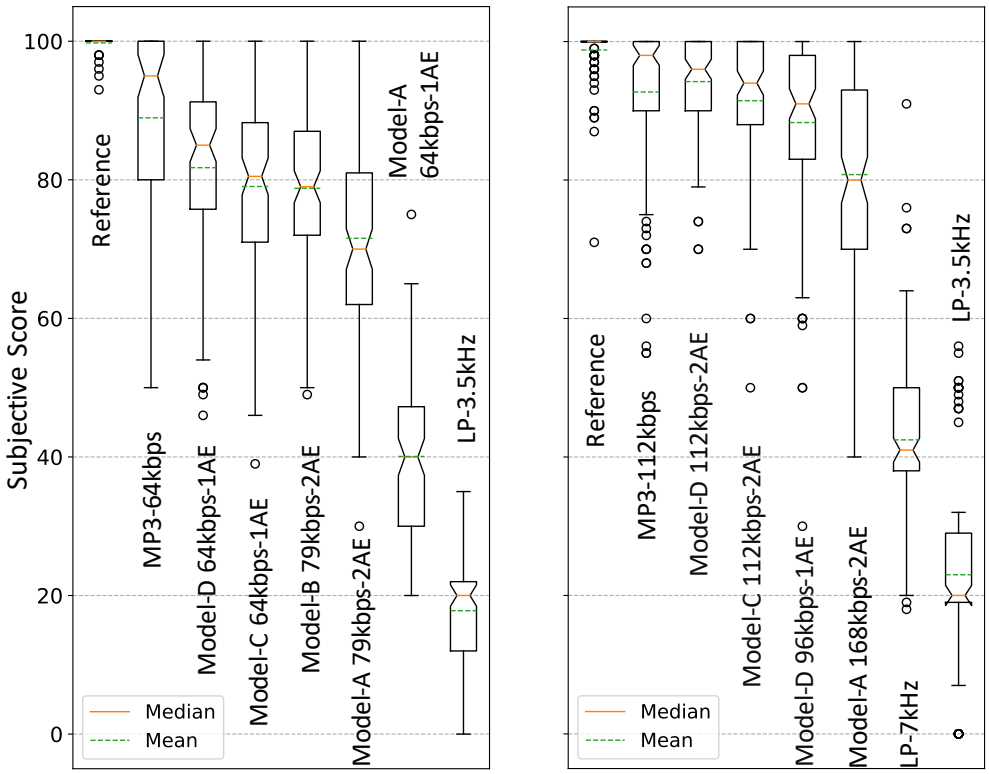
$$\mathcal{L} = \mathcal{L}_1 : \text{Model-A}, \quad \mathcal{L} = \lambda_1\mathcal{L}_1 + \lambda_2\mathcal{L}_2 : \text{Model-B},$$

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

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

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.



(a) Low bitrates (b) High bitrates

Figure 4.5: Subjective scores from the MUSHRA tests

4.4.2 Subjective listening test

Two MUSHRA tests [30] 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 unseen test set. Figure 4.5 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 Figure 4.5 (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

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

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 at a higher bitrate (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 Figure 4.5 (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).

4.5 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

CONCLUSION

5.1 Thesis Summary

Waveform coding, an indispensable speech/audio processing component, serves a critical role in data storage and communication. A waveform codec converts a speech/audio waveform into a highly compact bitstream, to be transmitted to the receiver side where the waveform is reconstructed with the least possible perceptual discrepancy from the uncompressed counterpart. Since waveform codecs are usually deployed on low-power devices for real-time communications, recent data-driven approaches may not be feasible in practice. For example, the user-perceived latency can go beyond tolerance along with the overwhelming runtime complexity, which can be energy-consuming. The dissertation introduced a scalable and efficient methodology to compress speech and audio waveforms. While reflecting upon recent advancements in deep learning, it integrates knowledge and techniques from digital signal processing and psychoacoustics into the model design and training procedure.

In Chapter 2, we proposed a scalable framework termed as cascaded cross-module residual learning (CMRL) derived from multistage vector quantization (MSVQ) to facilitate the model optimization and achieve scalability. Ideally, this would not be necessary if a sufficiently large DNN could be properly optimized. In practice, this is often not the case as optimizing such a complex model is challenging. To mitigate this issue, CMRL does not count on achieving the good result by updating all parameters together. Instead, it takes turns to optimize each module from a whole DNN in a greedy manner: the error from the module that is not perfectly optimized will be the input of succeeding modules where it can be further minimized. By training each module from a DNN sequentially where the error is cascaded to the next module as its input, our model

performance scales well, as more bandwidth is available.

In addition to scalability, efficiency is also highly desired in waveform codecs to run on low-power electronics, which unfortunately poses a major challenge to DNN based methodologies. In Chapter 3, we outsourced the resonance modeling task to linear predictive coding (LPC) by leveraging the speech production theory and employing DNN only to where it is necessary. We didn't just take it from the shelf but remodeled a conventional LPC analyzer as a trainable and efficient preprocessor for spectral envelope estimation. Having that, it is not limited to a pre-calculated fixed bitrate but optimized along with the neural waveform codec for a frame-wise dynamic bit allocation. Our coding performance can be scaled up to a transparent level superior to that from conventional codecs. Compared with recent neural network based generative models, our model size is significantly smaller.

Finally, neural waveform coding is applied to audio signals in Chapter 4 where we addressed the model efficiency issue from another perspective: we proposed a novel loss function that can more faithfully reflect upon human auditory perception by integrating psychoacoustics. Compared to speech signals, the audio frequency can be greater than 20 kHz to render the timbre of various musical instruments, which entails a significantly higher sample rate. Having that, it is usually unrealistic to completely avoid any artifact during audio coding, even with neural networks. As a consequence, a more feasible goal is to prioritize the artifact removal only if the artifact is audible. In other words, if the artifact is otherwise inaudible, there is no need to remove it as it will not lead to any perceptual degradation. To determine whether the artifact is audible or not during model training, we presented psychoacoustic loss terms which are specifically based on simultaneous auditory masking of human ears: we calculated global masking threshold by summing up the tonal and non-tonal masking curves along with the absolute hearing threshold, and used that to let the model focus on suppressing the audible artifact only. As a result, the proposed psychoacoustic calibration leads to noticeable performance gains for audio coding.

5.2 Beyond Neural Waveform Coding: a Semi-Supervised Acoustic Unit

The proposed neural waveform codec not only serves an algorithm to compress speech and audio signals but also gives potential for various acoustic signal processing tasks via a semi-supervised manner [134] including but not limited to speech enhancement [135] and recognition [136].

As indicated in Chapter 1, ubiquitous are end-to-end neural networks for speech enhancement [137, 138]. Usually, these models are trained in a supervised manner, as they consume noisy mixtures in the time domain as the input, and learn to output each source separately for speech separation, or preserve the speech source of interest while suppressing all other interfering signals for speech enhancement. One of the most compelling challenges is generalizability. While they perform well on the training dataset, they cannot be generalized to the test dataset well, since, for example, the noise type can be unseen.

Our neural waveform codec may open a direction to address the generalizability issue. One idea is to firstly train a neural codec on clean speech signals so as to learn a set of salient embeddings of the input. Based on the codec, another network component is employed in the latent space to conduct separation / enhancement on the embeddings. As the neural codec is trained on clean speech only, when the input becomes noisy, the pattern of the embedding can differ noticeably from that of a clean speech signal so as to be easily detected by the separation network; moreover, as this method is not bounded by the noise type, it can be extended to unseen noise types during model inference.

Likewise, the learned latent representation of speech signals can be used in ASR. The potential virtue of using our neural codec as a pre-processor is twofold. First, it can be optimized together with the neural network for ASR, as opposed to taking a deterministic DSP codec from the shelf which can lead to suboptimal performance. Second, during the encoding process the speaker identification is also encrypted. While the encrypted representation can still be recognized to enable communications services, the privacy of the speaker is better protected.

In summary, we believe that the proposed efficient and scalable neural waveform codec in this thesis can serve a semi-supervised acoustic unit. It can learn task-specific representations of the acoustic input signals to facilitate downstream applications towards a more human-like and computationally affordable solution for natural language understanding in the era of Internet of Things (IoT) and Big data.

Bibliography

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

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

[3] 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.

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

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

[6] 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.

[7] M. Schroeder and B. 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.

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

[9] 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.

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

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

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

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

[14] 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.

[15] 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.

[16] 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.

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

[18] 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.

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

[20] 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.

[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] P. Govalkar, J. Fischer, F. Zalkow, and C. Dittmar, "A comparison of recent neural vocoders for speech signal reconstruction," in *Proc. 10th ISCA Speech Synthesis Workshop*, pp. 7–12, 2019.

[24] 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.

[25] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *International Conference on Machine Learning*, pp. 2410–2419, PMLR, 2018.

[26] 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.

[27] 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.

[28] 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.

[29] W. A. Jassim, J. Skoglund, M. Chinen, and A. Hines, "Speech quality factors for traditional and neural-based low bit rate vocoders," in *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, pp. 1–6, IEEE, 2020.

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

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

[38] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[39] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[40] 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.

[41] 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.

[42] 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.

[43] 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.

[44] 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.

[45] D. Rowe, "Codec 2-open source speech coding at 2400 bits/s and below," in *TAPR and ARRL 30th Digital Communications Conference*, pp. 80–84, 2011.

[46] R. L. Goldstone, "Becoming cognitive science," *Topics in cognitive science*, vol. 11, no. 4, pp. 902–913, 2019.

[47] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman, "How to grow a mind: Statistics, structure, and abstraction," *science*, vol. 331, no. 6022, pp. 1279–1285, 2011.

[48] R. Moreno and R. Mayer, "Interactive multimodal learning environments," *Educational psychology review*, vol. 19, no. 3, pp. 309–326, 2007.

[49] Y. P. Huang and R. P. Rao, "Predictive coding," *Wiley Interdisciplinary Reviews: Cognitive Science*, vol. 2, no. 5, pp. 580–593, 2011.

[50] J. M. Kilner, K. J. Friston, and C. D. Frith, "Predictive coding: an account of the mirror neuron system," *Cognitive processing*, vol. 8, no. 3, pp. 159–166, 2007.

[51] W. T. Fitch, "Toward a computational framework for cognitive biology: unifying approaches from cognitive neuroscience and comparative cognition," *Physics of life reviews*, vol. 11, no. 3, pp. 329–364, 2014.

[52] R. Kirov, "Brain oscillations and predictive coding in the context of different conscious states and sleep-wake cycle: Implications for decision making and psychopathology," *Frontiers in psychology*, vol. 7, p. 1768, 2016.

[53] R. Neff and A. Zakhor, "Very low bit-rate video coding based on matching pursuits," *IEEE Transactions on circuits and systems for video technology*, vol. 7, no. 1, pp. 158–171, 1997.

[54] T. Frajka and K. Zeger, "Residual image coding for stereo image compression," *Optical Engineering*, vol. 42, no. 1, pp. 182–189, 2003.

[55] Y. A. Reznik, "Coding of prediction residual in mpeg-4 standard for lossless audio coding (mpeg-4 als)," in *2004 IEEE International conference on acoustics, speech, and signal processing*, vol. 3, pp. iii–1024, IEEE, 2004.

[56] G. Fant, *Acoustic theory of speech production*. No. 2, Walter de Gruyter, 1970.

[57] W. L. Gulick, G. A. Gescheider, and R. D. Frisina, *Hearing: Physiological acoustics, neural coding, and psychoacoustics.* Oxford University Press, 1989.

[58] F. Baumgarte and C. Faller, "Binaural cue coding-part i: Psychoacoustic fundamentals and design principles," *IEEE transactions on speech and audio processing*, vol. 11, no. 6, pp. 509–519, 2003.

[59] H. Yang, K. Zhen, S. Beack, and M. Kim, "Source-aware neural speech coding for noisy speech compression," *arXiv preprint arXiv:2008.12889*, 2020.

[60] 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.

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

[62] 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.

[63] 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.

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

[65] 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.

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

[67] 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.

[68] 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 (CVPR)*, pp. 770–778, 2016.

[69] 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.

[70] 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.

[71] 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.

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

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

[74] 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.

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

[76] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

[77] S. J. Cao, C. Zhang, Z. L. Yao, W. C. Xiao, L. S. Nie, D. C. Zhan, Y. X. Liu, M. Wu, and L. T. Zhang, "Efficient and effective sparse LSTM on FPGA with bank-balanced sparsity," in *Proceedings of the ACM/SIGDA International Symposium on Field-Programmable Gate Arrays (FPGA)*, pp. 63–72, 2019.

[78] S. R. Wang, P. Lin, R. H. Hu, H. Wang, J. He, Q. J. Huang, and S. Chang, "Acceleration of LSTM with structured pruning method on FPGA," *IEEE Access*, vol. 7, pp. 62930–62937, 2019.

[79] M. Zhu and S. Gupta, "To prune, or not to prune: exploring the efficacy of pruning for model compression," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[80] S. Narang, E. Elsen, G. Diamos, and S. Sengupta, "Exploring sparsity in recurrent neural networks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[81] K. Zhen, H. D. Nguyen, F.-J. Chang, A. Mouchtaris, and A. Rastrow, "Sparsification via compressed sensing for automatic speech recognition," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021.

[82] 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.

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

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

[85] 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.

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

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

[88] 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.

[89] 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.

[90] 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.

[91] K. Zhen, M. S. Lee, J. Sung, S. Beack, and K. M, "Efficient and scalable neural residual waveform coding with collaborative quantization," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020.

[92] J. H. McClellen and M. A. Yoder, *DSP first: A multimedia approach.* Prentice Hall PTR, 1997.

[93] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, Mar 1982.

[94] 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.

[95] L. M. Supplee, R. P. Cohn, J. S. Collura, and A. V. McCree, "Melp: the new federal standard at 2400 bps," in *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 1591–1594, IEEE, 1997.

[96] J. Hai and E. M. Joo, "Improved linear predictive coding method for speech recognition," in *fourth international conference on information, communications and signal processing, 2003 and the fourth Pacific rim conference on multimedia. Proceedings of the 2003 joint*, vol. 3, pp. 1614–1618, IEEE, 2003.

[97] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376, 2006.

[98] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[99] A. Chowdhury and A. Ross, "Fusing mfcc and lpc features using 1d triplet cnn for speaker recognition in severely degraded audio signals," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1616–1629, 2019.

[100] 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.

[101] 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.

[102] 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.

[103] 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.

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

[105] 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.

[106] 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

standard 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.

[107] 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.

[108] J. M. Martin-D., 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.

[109] 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.

[110] 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.

[111] 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.

[112] 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.

[113] 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.

[114] 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.

[115] 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.

[116] Z. Y. Zhao, S. Elshamy, and T. Fingscheidt, "A perceptual weighting filter loss for DNN training in speech enhancement," in *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 229–233, IEEE, 2019.

[117] A. Kumar and D. Florencio, "Speech enhancement in multiple-noise conditions using deep neural networks," in *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, (San Francisco, CA, USA), pp. 352–356, 2016.

[118] 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.

[119] 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.

[120] 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.

[121] 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.

[122] K. Zhen, M. S. Lee, J. Sung, S. Beack, and M. Kim, "Psychoacoustic calibration of loss functions for efficient end-to-end neural audio coding," *IEEE Signal Processing Letters*, vol. 27, pp. 2159–2163, 2020.

[123] T. Painter and A. Spanias, "Perceptual coding of digital audio," *Proceedings of the IEEE*, vol. 88, no. 4, pp. 451–515, 2000.

[124] K. Tan, J. Chen, and D. Wang, "Gated residual networks with dilated convolutions for monaural speech enhancement," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, pp. 189–198, 2019.

[125] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1874–1883, 2016.

[126] M. S. Lee, J. Sung, M. Kim, and K. Zhen, "Audio signal encoding method and audio signal decoding method, and encoder and decoder performing the same," Apr. 30 2020. US Patent App. 16/543,095.

[127] 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.

[128] A. Herrera and F. Del Rio, "Frequency bark cepstral coefficients extraction for speech analysis by synthesis.," *The Journal of the Acoustical Society of America*, vol. 128, no. 4, pp. 2290–2290, 2010.

[129] S. S. Stevens and J. Volkmann, "The relation of pitch to frequency: A revised scale," *The American Journal of Psychology*, vol. 53, no. 3, pp. 329–353, 1940.

[130] E. Zwicker, "Subdivision of the audible frequency range into critical bands (frequenzgruppen)," *The Journal of the Acoustical Society of America*, vol. 33, no. 2, pp. 248–248, 1961.

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

[132] 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.

[133] 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.

[134] O. Chapelle, B. Scholkopf, and A. Zien, "Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]," *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 542–542, 2009.

[135] S. Leglaive, L. Girin, and R. Horaud, "Semi-supervised multichannel speech enhancement with variational autoencoders and non-negative matrix factorization," in *ICASSP 2019-*

2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 101–105, IEEE, 2019.

[136] G. Synnaeve, Q. T. Xu, J. Kahn, T. Likhomanenko, E. Grave, V. Pratap, A. Sriram, V. Liptchinsky, and R. Collobert, "End-to-end asr: from supervised to semi-supervised learning with modern architectures," *arXiv preprint arXiv:1911.08460*, 2019.

[137] K. Tan, B. Xu, A. Kumar, E. Nachmani, and Y. Adi, "Sagrnn: Self-attentive gated rnn for binaural speaker separation with interaural cue preservation," *IEEE Signal Processing Letters*, 2020.

[138] K. Zhen, M. S. Lee, and M. Kim, "A dual-staged context aggregation method towards efficient end-to-end speech enhancement," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 366–370, IEEE, 2020.

CURRICULUM VITAE

Kai Zhen
Luddy School of Informatics Computing and Engineering, Cognitive Science Program
Indiana University Bloomington
Contact: kaizhen723@gmail.com, `http://kaizhen.us`

Education

Indiana University, Bloomington, IN, USA — May, 2021

- Ph.D., dual major in Computer Sciences 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

Publications

1. **Kai Zhen**, Hieu Duy Nguyen, Feng-Ju (Claire) Chang, Athanasios Mouchtaris, "Sparsification via Compressed Sensing for Automatic Speech Recognition," *in Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Toronto, ON, Canada, June 6-12, 2021.*

2. Haici Yang, **Kai Zhen**, Seungkwon Beack, Minje Kim, "Source-Aware Neural Speech Coding for Noisy Speech Compression," *in Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Toronto, ON, Canada, June 6-12, 2021.*

3. **Kai Zhen**, Mi Suk Lee, Jongmo Sung, Seungkwon Beack, and Minje Kim, "Psychoacoustic Calibration of Loss Functions for Efficient End-to-End Neural Audio Coding," *IEEE Signal Processing Letters 27 (2020): 2159-2163.*

4. **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.*

5. **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.*

6. **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.*

7. Mi Suk Lee, Jongmo Sung, Minje Kim, and **Kai Zhen**. "Audio signal encoding method and audio signal decoding method, and encoder and decoder performing the same." U.S. Patent Application 16/543,095, filed April 30, 2020.

8. Jongmo Sung, 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.*

9. **Kai Zhen**, Aswin Sivaraman, Jongmo Sung, Minje Kim. "On psychoacoustically weighted cost functions towards resource-efficient deep neural networks for speech denoising." *The 7th Annual Midwest Cognitive Science Conference, Bloomington, IN, May 12-13, 2018.*

Industry Experience

Alexa Edge Machine Learning, Amazon, Inc.

- Applied Scientist Intern, 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