Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Telecommunication and Artificial Intelligence

Enhancing AutoVocoder Performance through Data Processing, Architecture Optimization, and Robustness in Text-to-Speech Systems

Master's Thesis

Author
Riad Larbi

Advisor
Dr. Mohammed Salah Al-Radhi

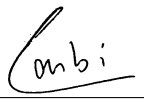
June 1, 2025

Contents

STUDENT DECLARATION

I, Riad Larbi, the undersigned, hereby declare that this thesis has been prepared by myself and without any unauthorized help or assistance. Only the specified sources (references, tools, etc.) were used. All parts taken from other sources word by word, or after rephrasing but with identical meaning, were unambiguously identified with explicit reference to the sources utilized. I authorize the Faculty of Electrical Engineering and Informatics of the Budapest University of Technology and Economics to publish the principal data of the thesis work (author's name, title, abstracts in English and in a second language, year of preparation, supervisor's name, etc.) in a searchable, public, electronic and online database and to publish the full text of the thesis work on the internal network of the university (this may include access by authenticated outside users). I declare that the submitted hardcopy of the thesis work and its electronic version are identical. Full text of thesis works classified upon the decision of the Dean will be published after a period of three years.

Budapest, June 2, 2025



Larbi Riad
Student

Abstract

Neural vocoders are integral to the synthesis of high-quality speech in mod- ern Text-to-Speech (TTS) systems, directly impacting naturalness, clarity, and adaptability. This thesis investigates two novel approaches to enhance AutoVocoder performance: spectral processing enhancement and F0-guided parallel architecture. The first approach employs advanced data preprocessing techniques with log magnitude and sine-cosine phase representation, alongside architectural refinements featuring ConvNeXtV2 blocks and separate phase and magnitude encoders. Post-processing methods including spectral gating further improve output quality. The second approach introduces an innovative F0-guided parallel architecture that processes spectral components through dual paths: a main processing path and an F0-masked path utilizing Gaussian masking around fundamental frequency regions. These paths are enhanced with Convolutional Block Attention Modules (CBAM) to improve feature extraction and are fused to create a robust representation of speech characteristics. Both approaches are evaluated through objective and subjective metrics, revealing complementary strengths. While the first approach shows im- proved spectral accuracy, the F0-guided architecture demonstrates superior pitch-related performance and robustness against noise. This comprehensive exploration contributes valuable insights for developing more natural, efficient, and adaptable speech synthesis systems.

Kivonat

A neurális vokóderek alapvető szerepet játszanak a modern szöveg-beszéd (TTS) rendszerek magas minőségű beszédszintézisében, közvetlenül befolyá- solva a természetességet, érthetőséget és alkalmazkodóképességet. Ez a dol- gozat az AutoVocoder teljesítményének javítására törekszik három kritikus tényezőre összpontosítva: az adatok fejlett feldolgozására, az architektúra finomítására és a valós körülményekhez való robusztusságra. Először modern adat-előfeldolgozási technikákat alkalmazunk, amelyek célja a beszéd felbon- tásának javítása és a zaj csökkentése. Ez magában foglalja a célzott normal- izálási módszereket, amelyek alapvető fontosságúak a szintézis pontosságának növelésében. Ezután az AutoVocoder architektúrájának optimalizálására kon- centrálunk. Rétegkonfigurációk és paraméterek finomhangolásával végzett it- eratív modellmódosítások révén próbáljuk növelni a hatékonyságot és csökken- teni a számítási terhelést anélkül, hogy az eredmény minősége csökkenne. Végül a rendszer robusztusságát vizsgáljuk különféle körülmények, például változó zajszintek és akcentusok mellett, hogy biztosítsuk az állandó és kiváló minőségű beszédszintézist. Ez a dolgozat a TTS rendszerek fejlesztéséhez járul hozzá egy strukturált megközelítéssel, amely alkalmazkodóbbá, hatékonyabbá és megbízhatóbbá teszi ezeket a rendszereket különböző környezetekben történő széleskörű alkalmazásra.

Chapter 1

Introduction

1.1 Overview of Speech Synthesis

Speech synthesis and acoustic modeling have been a widely researched task, particularly with recent advancements in the implementation of neural networks. The ability to convert text into natural-sounding speech has deep implications in a lot of domains, notably virtual assistants, accessibility tools, and entertainment.

Traditionally, mainstream Text-to-Speech (TTS) systems [31, 28, 60, 52] use mel-spectrograms as an intermediate representation for encoding speech waveforms. Mel-spectrograms provide a time-frequency representation that captures the main features of audio signals, allowing more efficient processing and improved synthesis quality. However, this conventional method is not without its limitations, prompting researchers to explore alternative approaches.

A notable advancement in this area is the introduction of the AutoVocoder [60], which represents a shift in how speech is encoded. Unlike traditional systems that rely on mel-spectrograms, the AutoVocoder encodes speech using redundant audio features, thereby emphasizing the importance of accurate encoding in learning better representations of speech waveforms. This innovative approach demonstrates that the encoding techniques based on redundant audio features yield results that surpass the quality and efficiency of mel-spectrograms. The foundation of the AutoVocoder is built upon the principles of autoencoders, a deep learning approach designed to learn efficient representations of data through unsupervised learning. Autoencoders consist of two main components: an encoder that compresses the input data into a lower-dimensional representation and a decoder that reconstructs the original data from this compact representation.

However, despite its innovative structure, the AutoVocoder still faces several critical limitations that affect its performance and the quality of its outputs. The first of these limitations is related to preprocessing. The current preprocessing stage is not effective enough in conditioning the input data for optimal encoding. As the quality of encoded output is heavily dependent on the input, improving the preprocessing pipeline could enhance both encoding efficiency and accuracy.

Second, there are architectural constraints that hinder the AutoVocoder's ability to extract complex patterns in the data. While ResNet is employed as the backbone model, its architectural depth may not be sufficient for capturing the intricate nuances of speech. Moreover, the current handling of real and imaginary components during encoding lacks

sophistication in representing phase and magnitude, suggesting that advanced methods could lead to better performance.

Third, the model lacks explicit mechanisms to focus on pitch-related information, which is crucial for natural-sounding speech. Fundamental frequency (F0) plays a vital role in conveying prosody and speaker characteristics, yet the baseline AutoVocoder processes all frequency components equally without special attention to these critical regions.

Lastly, the absence of post-processing in the AutoVocoder presents a significant drawback. Post-processing plays a critical role in refining the output, allowing for noise reduction, error correction, and enhanced clarity. Without this phase, the AutoVocoder's outputs are more likely to retain noise and inaccuracies, lowering the overall quality of the synthesized speech.

This thesis presents an evolutionary approach to address these limitations, we explored several complex architectures including transformer-based models [57] and Vision Transformer [12] adaptations, but computational constraints guided us toward a more efficient implementation that still incorporates attention mechanisms in a lightweight manner [62].

This thesis will explore the theoretical background of the existing AutoVocoder model, detail enhancement approaches, present their implementations, evaluate their performance, and discuss implications for future research.

1.2 Background and Related Work

1.2.1 AutoEncoder

An autoencoder is a type of neural network approach used for unsupervised learning and dimensionality reduction [4]. Its primary goal is to encode input data into a lower-dimensional representation called the latent space and then reconstruct the original input from this compressed representation. The autoencoder consists of two main components:

1. Encoder: Maps the input data to a latent representation.
2. Decoder: Reconstructs the input from the latent representation.

Autoencoders work by learning a compressed representation of the input data through an encoding process, which captures the essential features of the input in a lower-dimensional space. This compressed representation is then decoded back into the original input space by the decoder network. The network is trained to minimize the reconstruction error, typically using techniques like backpropagation [51] and gradient descent, to ensure that the output closely matches the input.

The main advantages of autoencoders architecture is that they do not require labeled data for training (unsupervised learning), and they can automatically learn meaningful features from the input data [35] which will be be very beneficial for tasks such as ours.

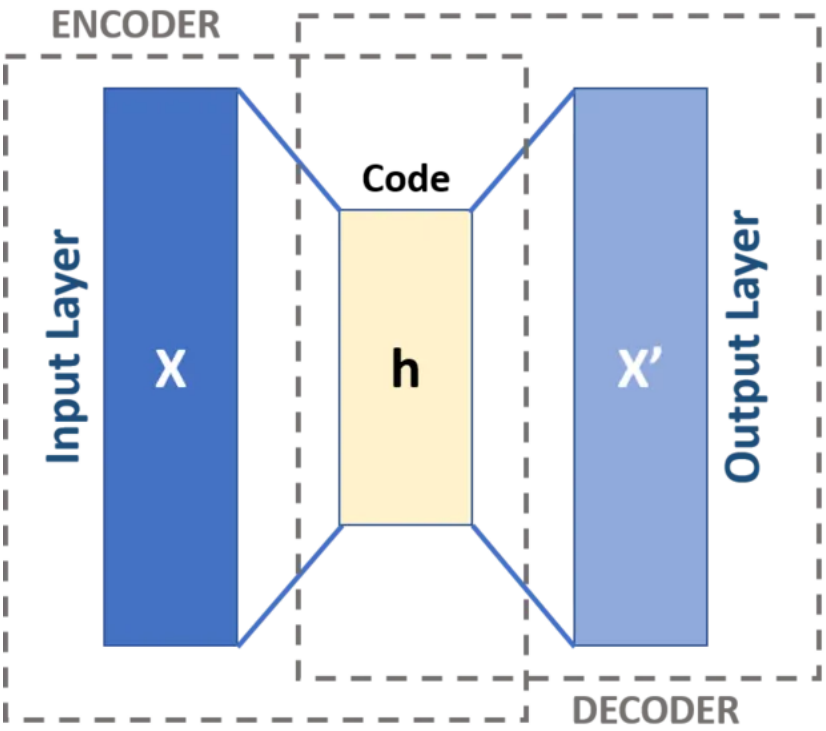


Figure 1.1: Basic architecture of an autoencoder showing input data (x) being compressed into a latent representation (h) by the encoder and then reconstructed by the decoder (x')

1.2.2 AutoVocoder Overview

Most state of art vocoders such as HiFi-GAN [31] and WaveNet [55] use the mel-spectrogram to encode audio data, however the AutoVocoder uses a novel approach, consisting of applying a differentiable variation of Short Time Fast Fourier transform to break the audio into redundant spectral features: phase, magnitude, real and imaginary [60]. Providing these features allows this model to learn better representation of our audio data, instead of imposing an already established representation. For example, some features like phase could be more efficiently represented given the magnitude spectrogram [56].

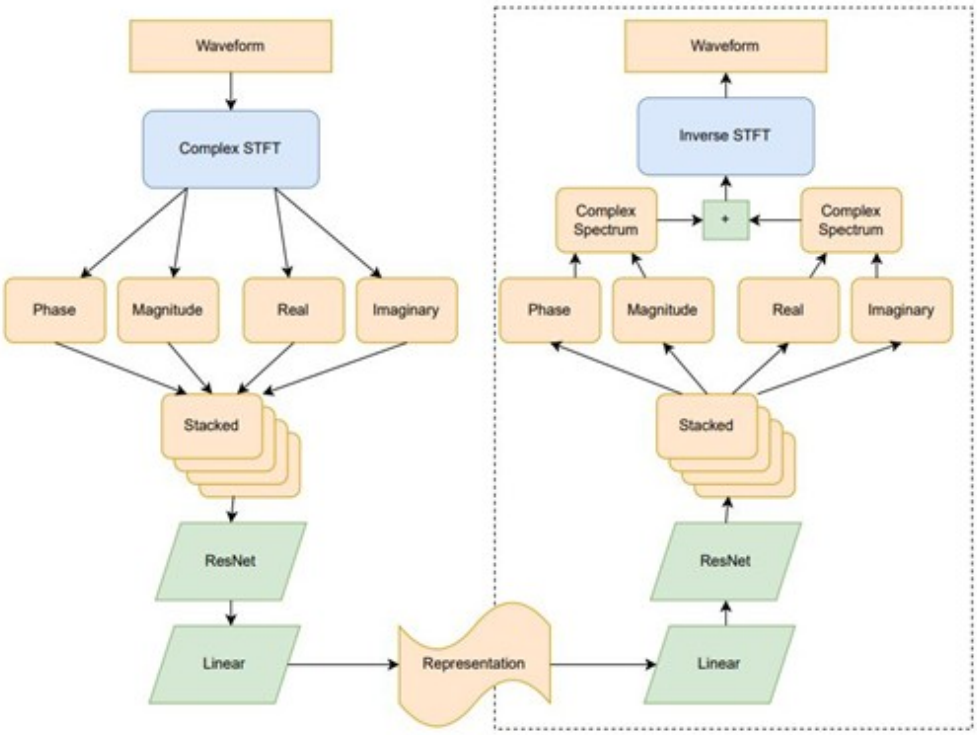


Figure 1.2: Overview of the AutoVocoder architecture showing the processing pipeline from audio waveform through spectral features to latent representation and back to audio

Feature Engineering: Using a differentiable version of short term Fourier transform [2] in PyTorch [48], the audio waveform is represented as a spectrogram, this spectrogram is then divided to two spectrograms, real and imaginary, these two spectrograms are then used to constructs the phase and magnitude spectrograms according to the following equation:

$$magnitude = \sqrt{\sum_{k}(Re(S)^2 + Im(S)^2 + 10^{-9})} \tag{1.1}$$

$$phase = arg(S) \tag{1.2}$$

The four spectrograms for real, imaginary, phase and magnitude are then stacked together alongside the channels dimension, obtaining a tensor of a shape: (B,4,N,T), where: B is the batch size, 4 is the number of channels, N is the number of frequency bins, T is the number of time frames. This tensor is then fed to the encoder.

Encoder: The encoder is based on a Residual Network or ResNet [18], which is a deep learning architecture that has been widely used for image classification tasks. It was introduced by Kaiming He et al. in 2015 and has since become one of the most popular models in the field of computer vision. The key innovation of ResNet is the introduction of residual blocks, which allow the network to learn residual functions with reference to the layer inputs, instead of learning unreferenced functions [18]. The AutoVocoder implements a ResNet where each residual block consists of two 2D convolutional layers of width 3, followed by a 2D batch normalization [24] and a ReLU [1] nonlinearity.The network consists of 11 such blocks, the first five having 4 input and output channels, the middle one having

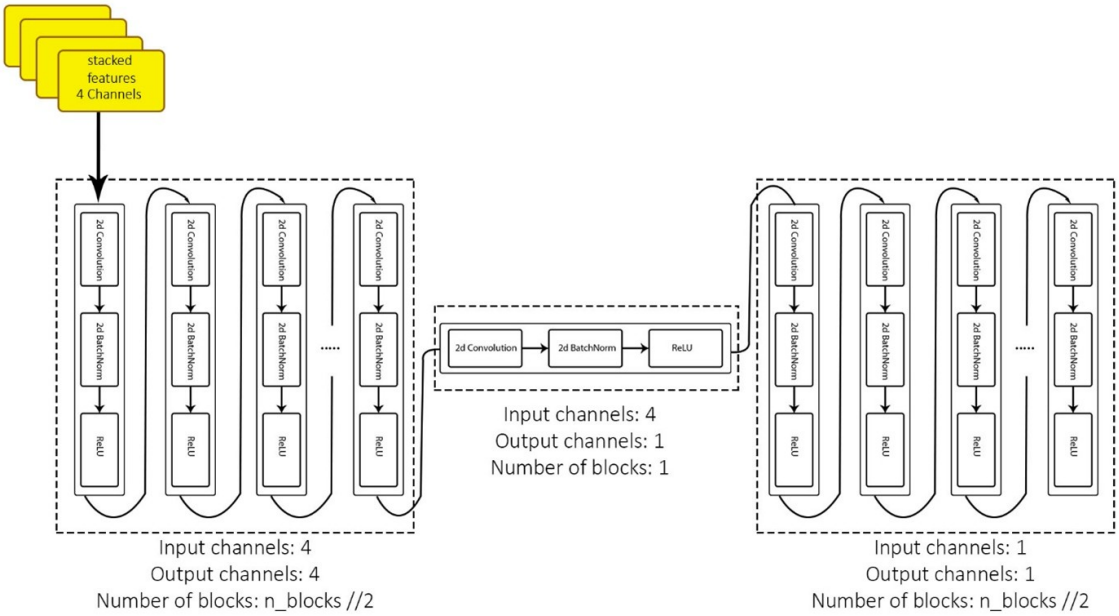


Figure 1.3: Structure of a basic ResNet block showing the residual connection that allows gradients to flow directly through the network

4 input channels but 1 output channel, with the remaining 5 blocks having 1 channel in and out. The output tensor with one channel is then fed to a linear layer that transforms it into its latent dimension.

Decoder: The Decoder follows a mirrored architecture, after decoding the latent representation into the original 4 channels, it reconstruct the waveform in one of the following three methods:

- **Cartesian**: the four channel-tensor is forwarded through a 2d convolution that outputs two channels, considered as real and imaginary, these two features are used to reconstruct the waveform using the inverse Short-Time Fourier Transform [16]:

$$y = iSTFT(real + j * imaginary) \tag{1.3}$$

- **Polar**: the same procedure as Cartesian, however the 2 outputted channels are considered as magnitude and phase, and used to reconstruct the waveform:

$$y = iSTFT(magnitude * e^{(phase*j)}) \tag{1.4}$$

- **Both**: 4 channels are used reconstruct the audio, by calculating polar and cartesian representations, and averaging them to obtain the input to the iSTFT.

Discriminators: The AutoVocoder also includes discriminators that are used to differentiate between real and generated audio samples, following the adversarial training paradigm introduced by Generative Adversarial Networks [15]. Discriminators are neural network components designed to evaluate the authenticity of audio outputs by learning to identify the characteristics of real audio data. Their primary purpose is to provide feedback to the generator during training, helping it improve the quality of synthesized audio. By distinguishing subtle differences between real and fake audio, discriminators guide the

generator to produce more realistic outputs. There are two types of discriminators designed to evaluate the generated audio against real audio:

- **Multi-Period Discriminator**: This discriminator uses multiple instances (with varying periods) to analyze audio. Each instance processes the audio with different temporal resolutions, allowing it to capture patterns and details across various time scales. It consists of several convolutional layers designed to gradually extract features from the input audio. The output is a flattened representation that helps differentiate between real and generated audio.
- **Multi-Scale Discriminator**: This discriminator operates at different scales by using average pooling layers to downsample the audio before passing it through its convolutional layers. It consists of multiple instances that analyze audio at various resolutions, enabling it to recognize both fine and coarse features. This design enhances its ability to detect subtle differences between real and synthesized audio.

Losses and Backpropagation:

- **Discriminator Losses**: The two discriminators (MPD and MSD) aim to distinguish real audio from generated audio. The loss here measures how well each discriminator can tell the real from the fake, with the total loss being the sum of both discriminators' performance.
- **Generator Losses**: The generator uses several losses:
  - Mel-Spectrogram Loss: Ensures the generated audio's spectrogram matches the real audio's spectrogram, computed using techniques from mel-frequency analysis [10].
  - Waveform Loss: Compares the actual waveforms of real and generated audio to make them similar.
  - Feature Matching Loss: Encourages the generator to match the internal features (extracted by the discriminators) of real and generated audio.
  - Adversarial Loss: Helps the generator improve by trying to "fool" the discriminators into classifying the generated audio as real [15].

1.3 Research Objectives and Approaches

This thesis explores two distinct approaches for enhancing the AutoVocoder while maintaining its core principles of waveform representation learning via an autoencoder framework with adversarial loss. Each approach addresses specific limitations of the baseline model through different architectural and processing strategies.

Both approaches share some common preprocessing techniques and evaluation methodologies but differ substantially in their core architectural designs and feature extraction mechanisms. The following chapters will detail each approach individually, providing a comprehensive analysis of their design, implementation, and performance.

1.3.1 First Approach: Spectral Processing Enhancement

The first approach focuses on improving data representation, architectural design, and output refinement through:

1. Implementing advanced preprocessing techniques including log magnitude and sine-cosine phase representation
2. Redesigning the AutoVocoder architecture with ConvNeXtV2 [63] blocks and separate phase and magnitude encoders
3. Applying post-processing methods such as spectral gating to improve the synthesized audio quality

1.3.2 Second Approach: F0-Guided Parallel Architecture

The second approach introduces a novel architecture that explicitly leverages fundamental frequency information:

1. Utilizing phase, magnitude, and power spectrums as comprehensive input representations
2. Implementing parallel processing paths: a main path for general spectral processing and an F0-masked path with Gaussian masking around fundamental frequency regions
3. Enhancing residual blocks with Convolutional Block Attention Modules (CBAM) to improve feature extraction
4. Employing feature fusion techniques to combine information from both processing paths

Chapter 2

Spectral Processing Enhancement

This chapter details the first approach to enhancing the AutoVocoder, which focuses on spectral processing improvements through architectural refinements and improved data representation. This approach was published in WINS3 conference [34], and served as a foundation for the more advanced F0-guided architecture presented in the next chapter.

2.1 Approach Overview and Motivation

The baseline AutoVocoder, while innovative in its approach to speech synthesis, presents several limitations that affect the quality of synthesized speech. The most significant limitations relate to its representation of spectral features and the architectural design that processes these features. This approach addresses these limitations through three main innovations:

1. Advanced preprocessing techniques that better represent phase and magnitude information
2. Architectural refinements with specialized encoders for different signal components
3. Effective post-processing to enhance the quality of the synthesized audio

The fundamental insight driving this approach is the recognition that phase and magnitude are fundamentally different representations of audio signals, each conveying distinct aspects of the speech signal. By treating them separately in the architecture, we can potentially achieve better representation and reconstruction of both components, leading to higher quality synthesized speech.

2.2 Feature Engineering

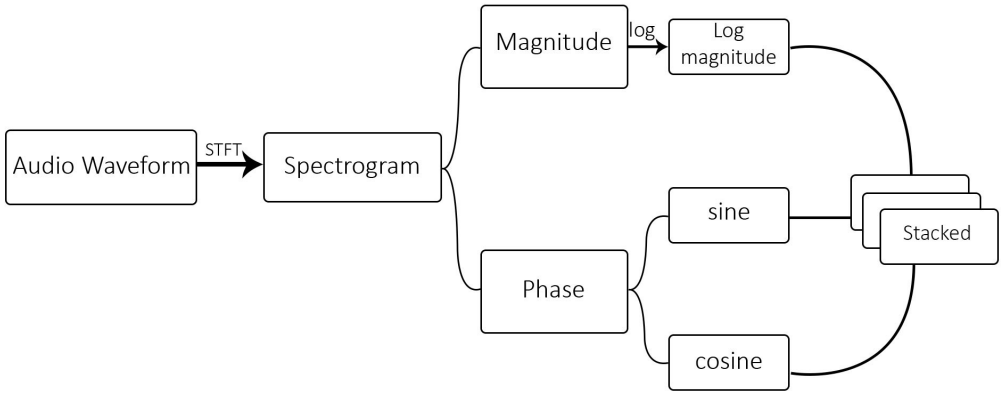


Figure 2.1: Waveform Representation and Processing Pipeline showing the transformation from raw audio to specialized representations for phase and magnitude

- **Log Magnitude** In the baseline AutoVocoder, linear magnitude is used to represent the magnitude spectrogram of the audio signal. However, after careful consideration, the log magnitude was chosen for this study. Log magnitude provides several advantages over linear magnitude, particularly for neural networks, as it compresses the dynamic range of the audio signal, making it easier for the model to learn more details while maintaining the overall structure of the audio.

  The human auditory system itself perceives sound intensity logarithmically rather than linearly [53], making log-magnitude a more perceptually relevant representation. Additionally, log-magnitude helps balance the contribution of high-energy and low-energy components in the frequency spectrum, preventing the model from overly focusing on high-energy components at the expense of perceptually important low-energy components.

  The log-magnitude transformation is computed as:

$$X_{log} = \log(|X| + \epsilon) \tag{2.1}$$

  where $|X|$ is the magnitude of the complex spectrogram obtained via Short-Time Fourier Transform (STFT) [2], and ϵ is a small constant (typically 10^{-5}) to avoid taking the logarithm of zero.

- **Sine-Cosine Phase Representation** An approach that was explored involved encoding the phase information using a sine-cosine representation, which is particularly useful in this application since it ensures a continuous periodic encoding and helps capture cyclical/periodic patterns. It may improve the model's understanding of sequential/temporal data and its ability to capture phase-related features, as phase plays a significant role in determining the timing and tonal quality of speech [41, 59].

Traditional phase representation suffers from discontinuities at the $-\pi$ to π boundary, which can make learning difficult for neural networks [46]. By representing phase as sine and cosine components, we avoid these discontinuities and provide a continuous representation that better captures the periodic nature of phase information.

The sine-cosine phase representation is computed as follows:

$$\cos(\phi) = \frac{\text{Re}(X)}{|X| + \epsilon} \tag{2.2}$$

$$\sin(\phi) = \frac{\text{Im}(X)}{|X| + \epsilon} \tag{2.3}$$

where $\text{Re}(X)$ and $\text{Im}(X)$ are the real and imaginary parts of the complex spectrogram, respectively, and ϵ is added for numerical stability.

The combination of these preprocessing techniques results in a rich set of input features for the model. The input tensor has the following components:

1. Real part of the complex spectrogram
2. Imaginary part of the complex spectrogram
3. Log magnitude
4. Cosine of the phase
5. Sine of the phase

These five channels provide a comprehensive representation of the speech signal, capturing both the traditional real-imaginary components and the specialized log-magnitude and sine-cosine phase components. This redundancy in representation allows the model to learn which features are most useful for different aspects of speech reconstruction.

2.3 Proposed Encoder Architecture

The implemented encoder architecture includes the inclusion of several elements that focus on processing the spectral components in a specialized manner, Figure 2.2 demonstrates these components, and will be detailed in the next subsections.

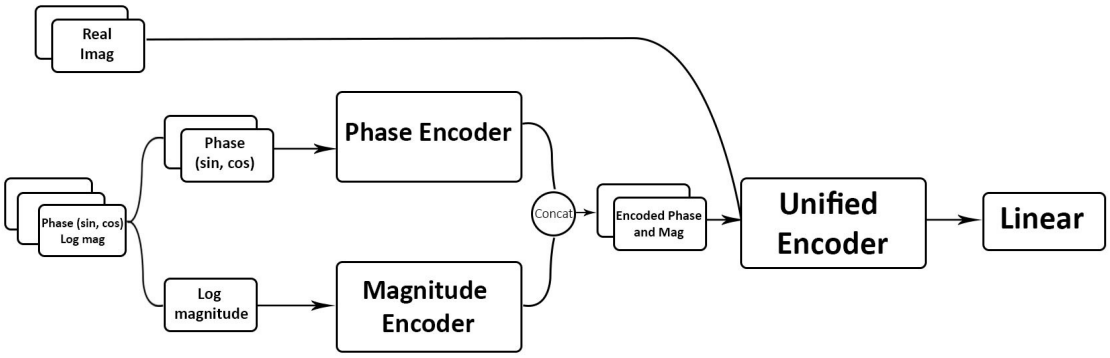


Figure 2.2: Complete Encoding Architecture showing the flow of information from input spectral features through specialized encoders to the unified representation

2.3.1 Replacing ResNet with ConvNeXtV2

The baseline AutoVocoder architecture utilized ResNet as the backbone for feature extraction. However, it was determined that ResNet did not provide the required depth and flexibility for capturing complex speech patterns. Therefore, ConvNextV2 [63], a modern architecture known for its superior performance in image and audio tasks, was adopted. ConvNeXt represents a modernization of the traditional ConvNet design [38], incorporating lessons learned from Vision Transformers while maintaining the efficiency of convolutional operations.

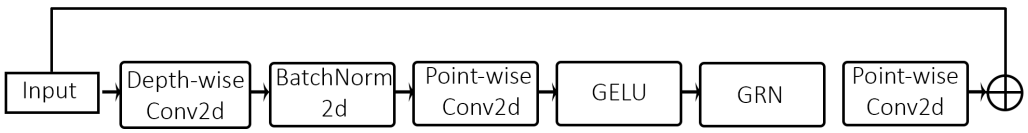


Figure 2.3: ConvNeXtV2 block Architecture showing the sequence of operations

ConvNeXtV2 incorporates several advancements over traditional ResNet blocks, These improvements allow ConvNeXtV2 to capture more complex patterns in the speech signal while maintaining computational efficiency:

- **Depth-wise convolution**: applies a separate filter to each input channel independently[7]. This means each filter only interacts with one channel, not all of them, it will serve as a computation efficiency enhancement in the case of multi-channel tensor. By separately processing each input channel, depth-wise convolutions reduce the number of parameters and computational complexity while still capturing spatial patterns within each feature channel.
- **Batch Normalization**: Batch Norm 2D serves to stabilize and accelerate the training of the model by normalizing activations, mitigating internal covariate shifts,

and improving gradient flow [24]. It normalizes the output of the preceding layer by calculating the mean and variance across the batch dimension, applying the normalization, and then learning a scaling and shifting parameter for each channel.

- **Point-wise convolution**: a 2d convolution with a 1x1 kernel, it serves primarily to mix information across channels, reduce dimensionality, and enhance feature learning in CNNs. After depth-wise convolutions process each channel independently, point-wise convolutions combine information across channels, allowing for cross-channel interactions and feature synthesis.
- **GELU**: this activation function outputs a scaled version of the input based on its probability under a Gaussian distribution [19]. While ReLU is computationally efficient and simple to implement, Gaussian Error Linear Unit offers advantages in terms of:
  - Smooth non-linearity, reducing the risk of dead neurons.
  - Gradient flow, enhancing training stability.
  - Better modeling of complex relationships due to its non-monotonic nature.
- **GRN**: The Global Response Normalization layer effectively normalizes the output of the preceding layer by computing a global response normalization, scaling and shifting it with learnable parameters, and maintaining a residual connection. It helps calibrate feature responses across different channels, enhancing the model's ability to focus on the most salient features.

The application of ConvNeXtV2 block is similar to what was used in BiVocoder [13], however instead of 1D convolution, 2d convolution is used, in addition to keeping Batch Normalization instead of adopting Layer normalization that is conventionally used in ConvNeXtV2 block architecture.

2.3.2 Separate Phase and Magnitude Encoders

A key advancement in the first approach is the introduction of separate encoders for phase and magnitude. This design recognizes that phase and magnitude are fundamentally distinct representations of a speech signal, each containing unique information critical to understanding and reconstructing speech effectively.

The concept of specialized encoders represents a significant departure from the baseline AutoVocoder, which processes all components through the same network. This specialization allows each encoder to develop feature extractors that are optimized for the particular characteristics of its input, potentially leading to better overall representation of the speech signal.

- **Phase Encoder**: This component is specifically designed to extract temporal and phase-related information from the audio input. It focuses on the timing and progression of sound waves, capturing how they change over time.

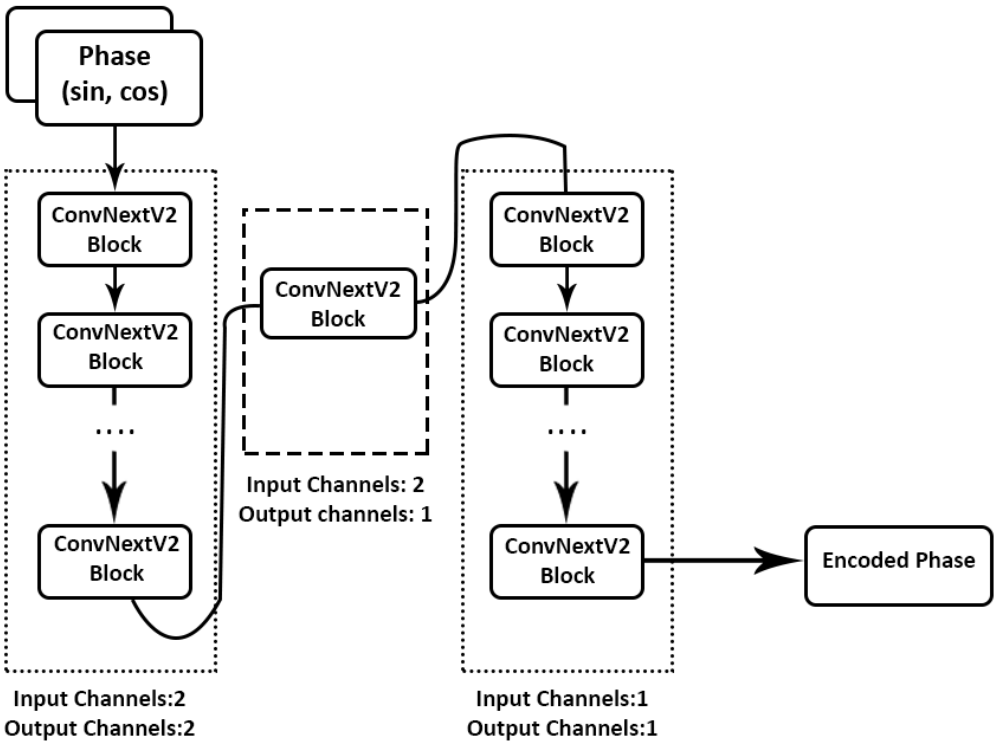


Figure 2.4: Detailed architecture of the Phase Encoder showing the processing of sine-cosine phase representations through a series of ConvNeXtV2 blocks

This encoder is adept at discerning subtle variations in timing that are crucial for phonetic distinctions and speech clarity. By isolating phase information, the Phase Encoder enhances the system's ability to track rapid changes in speech patterns, thus ensuring that the timing aspects, which are essential for natural speech perception, are preserved The Phase Encoder processes the sine-cosine representation of phase through a cascade of ConvNeXtV2 blocks. The architecture begins with several blocks that maintain the two-channel input (sine and cosine components) before gradually reducing to a single channel. This gradual reduction allows the network to learn combinations of sine and cosine components that efficiently represent phase information. The phase information is particularly important for preserving the timing of speech events, such as the rapid transitions between phonemes and the temporal structure of consonants. By dedicating a specialized encoder to phase information, we aim to better capture these critical timing aspects of speech.

- **Magnitude Encoder** In contrast, the Magnitude Encoder's mainly focuses on the spectral features of the speech signal. It analyzes the amplitude of the sound waves across various frequencies, effectively encoding the energy distribution that contributes to the timbre and intensity of the audio. This encoder specializes in identifying the harmonic content and frequency characteristics, which are necessary for distinguishing different phonemes. By focusing exclusively on magnitude, the Magnitude Encoder can optimize the extraction of features that characterize the loudness and frequency content of speech. The Magnitude Encoder processes the

log-magnitude representation through a series of ConvNeXtV2 blocks, all maintaining a single-channel structure. This consistent channel dimension allows the network to focus on extracting patterns within the magnitude spectrum without needing to handle channel-wise transformations. Magnitude information is crucial for capturing the spectral envelope of speech, which defines the overall timbre and identity of speech sounds. The formant structure, harmonic content, and energy distribution across frequencies all contribute to the intelligibility and naturalness of synthesized speech. By dedicating a specialized encoder to magnitude information, we aim to better capture these spectral characteristics of speech.

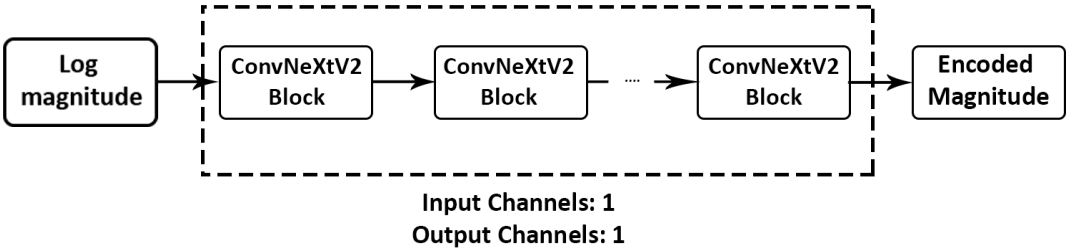


Figure 2.5: Detailed architecture of the Magnitude Encoder showing the processing of log magnitude representations through a series of ConvNeXtV2 blocks

The separation of these two encoders allows each to specialize in extracting their respective features more effectively. This specialization may lead to a potentially improved performance in encoding, as each encoder can utilize tailored algorithms and techniques suited to the specific characteristics of phase and magnitude. As a result, the system is capable of achieving more accurate reconstruction of speech, enhancing intelligibility and naturalness in synthesized speech outputs.

The Phase Encoder focuses on temporal aspects and the rapid transitions that characterize consonants and other time-dependent speech features. Meanwhile, the Magnitude Encoder captures the spectral envelope, formant structure, and energy distribution that define vowels and the overall timbre of speech. Together, they provide a comprehensive representation of the speech signal that captures both its temporal and spectral characteristics.

2.3.3 Unified Encoding of the Spectral Representations

Once the phase and magnitude have been independently encoded, their outputs are integrated through a Unified Encoder. This module combines the distinct outputs from both encoders, in addition to real and imaginary channels of the linear spectrogram. The output is then fed to a linear layer to obtain a cohesive representation of the speech signal.

The Unified Encoder plays a crucial role in maintaining the integrity of the speech signal by ensuring that both temporal (phase-related) and spectral (magnitude-related) aspects are preserved and represented harmoniously. It effectively bridges the gap between the two domains, allowing the system to leverage the strengths of both encoders while compensating for any limitations inherent in treating phase and magnitude separately.

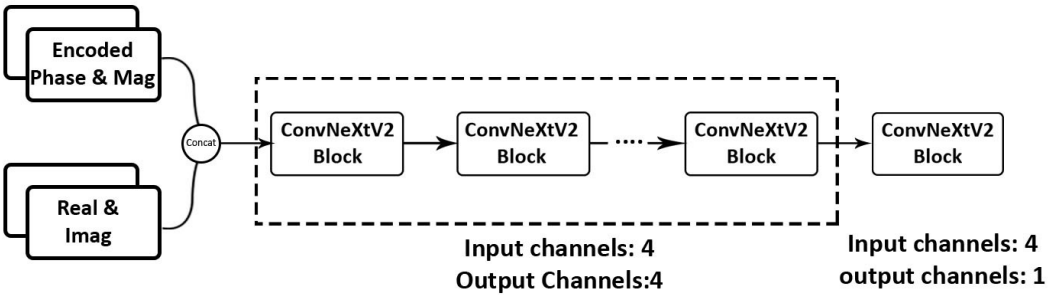


Figure 2.6: The Unified Encoder architecture showing how separate phase and magnitude information is integrated with the original real and imaginary components to form a comprehensive representation

The inclusion of the original real and imaginary components in this unified representation provides a reference point for the model, allowing it to compare the specialized representations from the Phase and Magnitude Encoders with the original complex spectrogram. This redundancy helps the model learn which representations are most useful for different aspects of speech reconstruction.

2.4 Proposed Decoder

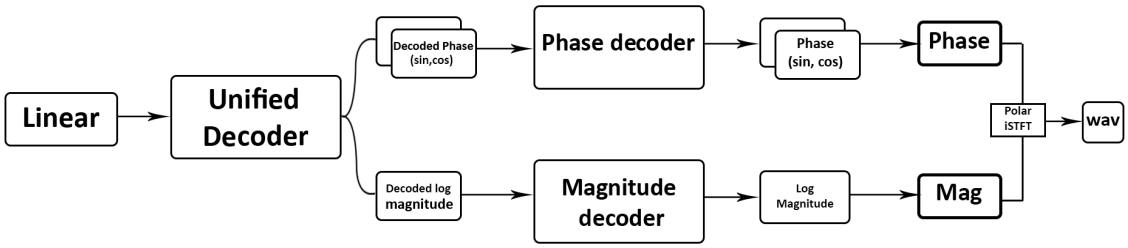


Figure 2.7: Complete decoder architecture showing the mirrored structure with specialized phase and magnitude decoders that reconstruct their respective components from the latent representation

The decoder implements a mirrored architecture to ensure effective signal reconstruction. The process begins with the latent representation being expanded through a linear layer to match the required dimensionality. This expanded representation is then processed through a unified decoder that gradually reconstructs the combined features. The unified decoder's output is subsequently split and forwarded to specialized phase and magnitude decoders. These dedicated decoders are designed to accurately reconstruct their respective components while maintaining the intricate relationships between phase and magnitude information.

The phase decoder pays particular attention to reconstructing the sine and cosine components, ensuring accurate phase reconstruction without discontinuities. Meanwhile, the magnitude decoder focuses on preserving the spectral envelope and energy distribution of the original signal. This specialized approach allows for more precise reconstruction of both components, leading to higher quality synthesized audio. The output phase and

log magnitude are then used to synthesize the waveform, using the polar iSTFT [29] implementation.

The specialized decoders mirror the encoders in their focus, with the Phase Decoder specifically designed to accurately reconstruct phase information from the latent representation, and the Magnitude Decoder focused on reconstructing the spectral envelope and energy distribution. This mirrored approach ensures that the information captured by the specialized encoders can be effectively utilized during the reconstruction process.

The entire encoder-decoder architecture is designed to maintain the advantages of both global and local feature processing while allowing for component-specific optimization. The use of CNBlocks throughout most of the architecture provides improved gradient flow and feature extraction capabilities compared to traditional ResBlocks, while the strategic placement of dimensionality reduction and expansion ensures efficient information processing without significant loss of critical audio features.

2.5 Proposed Post-Processing Procedure

While architectural improvements are essential for enhancing speech synthesis quality, post-processing techniques can further refine the output, addressing any remaining artifacts or imperfections. Several post-processing techniques were implemented as part of the Spectral Processing Enhancement approach.

2.5.1 Spectral Gating

For this enhancement, removing unwanted frequencies was essential due to a common limitation in the baseline autovocoder's synthesis process. While the AutoVocoder successfully generates high quality speech, it tends to produce output with higher amplitude than the original speech signals. This amplification inadvertently affects not just the desired speech components but also magnifies any background noise present in the signal.

To address this issue, we implemented spectral gating through the noise reduce [33] implementation. Spectral gating operates on the principle of analyzing the frequency spectrum of the audio signal and applying an adaptive threshold to separate speech from noise components [61]. The process can be broken down into several key steps:

1. **Noise Profile Estimation**: The algorithm first estimates the noise profile from portions of the signal where speech is absent, this creates a statistical model of the background noise's spectral characteristics. This is typically done by analyzing frames at the beginning or end of the audio where speech is less likely to be present.

2. **Threshold Determination**: Based on the noise profile, the algorithm calculates frequency-dependent thresholds, these thresholds determine which spectral components should be preserved or attenuated. The threshold is typically set as a function of the noise profile, often scaled by a factor that determines the aggressiveness of the noise reduction.

3. **Spectral Gate Application**: The signal is transformed into the frequency domain using Short-Time Fourier Transform (STFT). Frequency components below the calculated threshold are attenuated, while components above the threshold, likely corresponding to speech, are preserved. The attenuation is typically applied as a

gain factor that varies smoothly across frequencies to avoid introducing musical noise or other artifacts.

4. **Signal Reconstruction**: The processed spectrum is transformed back to the time domain using inverse STFT [16], the result is a cleaner signal with reduced background noise while maintaining speech intelligibility. Overlap-add techniques are used to ensure smooth transitions between frames.

The spectral gating process can be formally described as:

$$Y(f,t) = X(f,t) \cdot G(f,t) \tag{2.4}$$

where $X(f,t)$ is the original spectrogram, $G(f,t)$ is the gain function determined by the spectral gate, and $Y(f,t)$ is the processed spectrogram. The gain function is typically computed as:

$$G(f,t) = \begin{cases} 1, & \text{if } |X(f,t)| > \alpha \cdot N(f) \\ \beta, & \text{otherwise} \end{cases} \tag{2.5}$$

where $N(f)$ is the estimated noise profile, α is a scaling factor that determines the threshold, and β is the attenuation factor applied to components below the threshold. This approach builds upon classical spectral subtraction methods [14] but with adaptive thresholding for better performance.

2.5.2 Additional Post-processing

In addition to spectral gating, several other post-processing techniques were applied to further enhance the quality of the synthesized speech:

DC Offset Removal: Removes any constant electrical offset in the audio signal. This is important because DC offset can cause unnecessary power consumption and reduce the headroom for the actual audio signal. Having no DC offset means the audio signal is properly centered around zero. DC offset removal is implemented as:

$$y[n] = x[n] - \frac{1}{N} \sum_{i=0}^{N-1} x[i] \tag{2.6}$$

where $x[n]$ is the input audio signal, N is the length of the signal, and $y[n]$ is the output signal with DC offset removed.

Peak Normalization: Adjusts the overall volume of the audio to optimize its amplitude. Ensures the loudest part of the audio uses the full available dynamic range without clipping, making different audio recordings consistent in terms of maximum volume level. Peak normalization is implemented as:

$$y[n] = \frac{x[n]}{\max(|x|)} \cdot \text{target\_level} \tag{2.7}$$

where $x[n]$ is the input audio signal, $\max(|x|)$ is the maximum absolute amplitude of the signal, target_level is the desired peak level (typically 0.95 to leave some headroom), and $y[n]$ is the normalized output signal.

2.6 Implementation Details

The Spectral Processing Enhancement approach was implemented using PyTorch, with a focus on optimizing the architectural components for speech synthesis.

2.6.1 Phase and Magnitude Encoders

The Phase Encoder processes the sine-cosine representation of phase through a series of ConvNeXtV2 blocks. The number of blocks and their configuration were determined through empirical testing to provide optimal performance while maintaining computational efficiency.

The architecture of the Phase Encoder follows this structure:

Algorithm 1 Phase Encoder Construction

function BUILDPHASEENCODER
    $phase\_encoder \leftarrow$ Empty ModuleList
    ▷ First set of blocks maintain channel dimension
    for $i = 0$ **to** 2 **do**
        $phase\_encoder.add(CNBlock(input\_channels = 2, output\_channels = 2))$
    end for
    ▷ Transitional block reduces channel dimension
    $phase\_encoder.add(CNBlock(input\_channels = 2, output\_channels = 1))$
    ▷ Final set of blocks process with reduced dimensionality
    for $i = 0$ **to** 2 **do**
        $phase\_encoder.add(CNBlock(input\_channels = 1, output\_channels = 1))$
    end for
    return *phase_encoder*
end function

Similarly, the Magnitude Encoder processes the log magnitude representation through a series of ConvNeXtV2 blocks:

Algorithm 2 Magnitude Encoder Construction

function BUILDMAGNITUDEENCODER
    $magnitude\_encoder \leftarrow$ Empty ModuleList
    ▷ Series of blocks with consistent dimensionality
    for $i = 0$ **to** 4 **do**
        $magnitude\_encoder.add(CNBlock(input\_channels = 1, output\_channels = 1))$
    end for
    return *magnitude_encoder*
end function

The Unified Encoder combines the outputs from the Phase and Magnitude Encoders along with the original real and imaginary components of the STFT. This combined representation is then processed through additional ConvNeXtV2 blocks before being projected to the latent space.

Algorithm 3 Unified Encoder Forward Pass

function UnifiedEncoderForward($phase\_features$, $magnitude\_features$, $real\_imag\_components$) ▷ Combine features from various sources
    $combined\_features \leftarrow$ Concatenate([$phase\_features$,
        $magnitude\_features$,
        $real\_imag\_components$])
    ▷ Process through unified encoder blocks
    for $i = 0$ **to** 6 **do**
        $combined\_features \leftarrow$ CNBlock($combined\_features$,
            $input\_channels = 5$,
            $output\_channels = 5$)
    end for
    ▷ Final dimensionality reduction
    $reduced\_features \leftarrow$ CNBlock($combined\_features$,
        $input\_channels = 5$,
        $output\_channels = 1$)
    ▷ Project to latent space
    $latent\_representation \leftarrow$ LinearProjection($reduced\_features$)
    return $latent\_representation$
end function

2.6.2 Proposed Decoder Implementation

The decoder mirrors this architecture but in reverse, expanding the latent representation and eventually splitting it into separate paths for phase and magnitude reconstruction:

Algorithm 4 Decoder Forward Pass

function DecoderForward($latent\_representation$) ▷ Expand latent representation
    $expanded\_features \leftarrow$ LinearExpansion($latent\_representation$)
    ▷ Process through unified decoder blocks
    for each $block$ in $unified\_decoder\_blocks$ **do**
        $expanded\_features \leftarrow block(expanded\_features)$
    end for
    ▷ Split into separate paths for phase and magnitude
    $phase\_reconstruction\_input, magnitude\_reconstruction\_input \leftarrow$
        SplitFeatures($expanded\_features$)
    ▷ Process through specialized decoders
    $phase\_reconstruction \leftarrow$ PhaseDecoder($phase\_reconstruction\_input$)
    $magnitude\_reconstruction \leftarrow$ MagnitudeDecoder($magnitude\_reconstruction\_input$)
    ▷ Generate waveform using polar iSTFT
    $waveform \leftarrow$ PolarISTFT($magnitude\_reconstruction, phase\_reconstruction$)
    return $waveform$
end function

2.7 Training Methodology

The Spectral Processing Enhancement model was trained using the LJSpeech 1.1 Dataset [25], which consists of approximately 13,100 short audio clips of a single female speaker

reading passages from 7 non-fiction books. The total audio length is over 24 hours, making it ideal for text-to-speech model training and evaluation.

Training was conducted with the following hyperparameters:

- Optimizer: AdamW [29]
- Learning rate: 0.0002
- Batch size: 16
- Beta1: 0.8
- Beta2: 0.99
- Weight decay: 0.01
- Training steps: 400,000

The loss function combined adversarial loss, feature matching loss, mel-spectrogram loss, and waveform loss, weighted appropriately to ensure balanced optimization of all speech aspects.

2.7.1 Model Size

The Spectral Processing approach introduces additional parameters compared to the baseline AutoVocoder due to its specialized encoders and more complex architecture. Table 2.1 presents a breakdown of the model parameters.

Table 2.1: Spectral Processing Model Parameter Count Compared to Baseline

Model Component	**Parameters**	**% of Total**	**Baseline Params**	**Change (%)**
Encoder (Combined)	148,851	0.19%	132,463	+12.37%
Generator	149,404	0.19%	132,803	+12.49%
Multi-Period Discriminator	45,216,347	58.0%	–	–
Multi-Scale Discriminator	32,580,703	41.8%	–	–
Total Model Parameters	**78,095,305**	**100%**	**265,266**	–

Despite the increased parameter count compared to the baseline AutoVocoder, the majority of parameters remain in the discriminator components. The actual encoder and generator, which are used during inference, contain relatively few parameters and remain computationally efficient.

2.8 Evaluation Metrics

2.8.1 Objective Metrics

We employed five objective metrics to assess the quality of synthesized speech, these metrics include root mean square error of logarithmic amplitude spectra (LAS-RMSE), RMSE of F0 (F0-RMSE), and voice/unvoiced (V/UV) error.

- **Signal-to-Noise Ratio (SNR)**: Measures signal clarity, with higher values indicating clearer speech with less noise.

$$\text{SNR} = 10 \log_{10} \left(\frac{\sum_{t=1}^{T} x(t)^2}{\sum_{t=1}^{T} \left(x(t) - \hat{x}(t) \right)^2} \right) \tag{2.8}$$

- **Log-Amplitude Spectrum Root Mean Square Error (LAS-RMSE)**: Evaluates spectral accuracy, with lower values indicating better spectral reproduction.

$$\text{LAS-RMSE} = \sqrt{\frac{1}{F} \sum^{F} \left(log|A(f)| - log|\hat{A}(f)| \right)^2} \tag{2.9}$$

- **F0 Root Mean Square Error (F0-RMSE)**: Measures pitch accuracy, with lower values indicating better reproduction of intonation patterns.

$$\text{F0-RMSE} = \sqrt{\frac{1}{T} \sum_{t=1}^{T} \left(F0(t) - \hat{F0}(t) \right)^2} \tag{2.10}$$

- **Voiced/Unvoiced Error (V/UV Error)**: Assesses voicing classification accuracy with lower values indicating better distinction between voiced and unvoiced speech segments.

$$\text{Error} = \frac{1}{T} \sum^{T} |V(t) - \hat{V}(t)| \tag{2.11}$$

These metrics were consistently implemented using the same methodology across all model evaluations to ensure fair comparison.

2.8.2 Subjective Evaluation Methods

The subjective evaluation of thisapproach we utilized MOSNet [39] for automated prediction of Mean Opinion Scores on a scale from 1 to 5. This provides a consistent methodology for comparing perceptual quality across all models.

2.9 Evaluation Results

2.9.1 Objective Evaluations Results

The Spectral Processing Enhancement approach was evaluated using several objective metrics to assess different aspects of speech quality. Table 2.2 presents these results in comparison to the baseline AutoVocoder.

Table 2.2: Objective Evaluation Results for Spectral Processing Approach

	SNR(dB) ↑	LAS-RMSE (dB) ↓	V/UV Error (%) ↓	F0-RMSE(Hz) ↓
Baseline AV	**2.42**	19.33	3.84	4.81
Spectral Processing	1.92	**12.20**	**2.85**	**3.50**

These results demonstrate several important findings:

- **Spectral Accuracy**: The Spectral Processing approach achieved a LAS-RMSE of 12.20 dB, showing a notable 36.9% improvement over the baseline (19.33 dB). This confirms that the separate processing of magnitude and phase components leads to better spectral representation.
- **Pitch Accuracy**: The F0-RMSE showed a significant 27.2% reduction from 4.81 Hz (baseline) to 3.50 Hz, indicating better preservation of fundamental frequency information despite not explicitly modeling F0.
- **Voicing Classification**: The V/UV error rate decreased from 3.84% to 2.85%, representing a 25.8% improvement in correctly identifying voiced and unvoiced segments.
- **Signal-to-Noise Ratio**: The SNR showed a modest decrease from 2.42 dB to 1.92 dB. This trade-off likely results from the model prioritizing spectral accuracy and pitch preservation over raw signal fidelity.

2.9.2 Subjective Evaluation

For subjective evaluation, we employed MOSNet [39], a neural network-based model trained to predict human Mean Opinion Scores for speech quality. Table 2.3 shows these results.

Table 2.3: Subjective Evaluation Results for Spectral Processing Approach (MOS)

	Score ↑
Baseline	3.01
Spectral Processing	3.07
Original	3.11

The subjective evaluation results confirm that the Spectral Processing approach achieves higher perceived quality than the baseline (3.07 vs. 3.01). This improvement, while modest, is significant considering the challenging task of matching human-level speech quality.

2.10 Limitations and Insights for Further Improvement

While the Spectral Processing Enhancement approach demonstrated improvements over the baseline AutoVocoder, several limitations and insights emerged during development and evaluation:

- **Lack of Explicit Pitch Modeling**: Despite improved F0-RMSE compared to the baseline, the approach does not explicitly model pitch information, which is crucial for natural-sounding speech, particularly for expressive or emotional speech.
- **Integration Challenges**: The separate phase and magnitude encoders provide specialized processing, but their integration in the unified encoder may not be optimal, potentially losing some of the benefit of specialization.
- **Parameter Efficiency**: While effective, the approach increases the parameter count compared to the baseline, which could be problematic for deployment in resource-constrained environments.

- **Limited Context Integration**: The approach processes each time-frequency bin relatively independently, potentially limiting its ability to capture long-range dependencies or contextual information in speech.

These limitations and insights informed the development of the second approach, the F0-Guided Parallel Architecture, which builds upon the concept of specialized processing while addressing these specific limitations through explicit pitch modeling and context-aware attention mechanisms.

2.11 Transition to the F0-Guided Approach

The Spectral Processing Enhancement approach demonstrated the value of specialized processing for different components of speech signals. This key insight; that different aspects of speech require different types of processing; serves as a foundation for the F0-Guided Parallel Architecture presented in the next chapter.

While the Spectral Processing approach separated phase and magnitude processing, the F0-Guided approach takes this concept further by creating specialized processing paths for different frequency regions based on pitch information. This represents a natural evolution of the core idea of specialization, moving from component specialization (phase vs. magnitude) to frequency-region specialization (pitch-relevant vs. general).

The F0-Guided approach also addresses several of the limitations identified in the Spectral Processing approach, particularly the lack of explicit pitch modeling. By directly incorporating fundamental frequency information into the architecture, the F0-Guided approach aims to achieve even better performance on pitch-related aspects of speech, which are critical for natural-sounding synthesis.

Chapter 3

F0-Guided Parallel Architecture

3.1 Approach Overview

Initial experiments with simple F0 concatenation showed limited improvement. The key insight is that even though the fundamental frequency information is crucial for speech synthesis, it still requires architectural support to be effective. Building around the insights from the spectral processing approach, our design for the encoding processing focuses on:

- **Explicit Modeling of Pitch**: Feature extraction guided by the F0
- **Specialized Processing Pathways**: Dedicated paths for different speech signal aspects; main path processing all frequency components, and a specialized path focusing on frequency regions around the fundamental frequency
- **Adaptive Attention**: Focus on the most informative aspects of the signal

The reason why we are focused on implementing this feature is because it determines perceived pitch and carries information related to:

- **Sentence level intonation**: Rising for questions, falling for statements [37]
- **Word level stress**: Emphasizing important words
- **Emotional content**: Expressing emotions through pitch variations
- **Speaker identity**: Contributing to voice characteristics [32]
- **Linguistic features**: Carrying tonal distinctions in many languages

Many systems, including the baseline AutoVocoder, rely on neural networks to implicitly capture pitch, leading to imprecise reproduction. Our F0 Guided approach explicitly incorporates F0 information for more accurate pitch modeling and natural speech synthesis.

3.2 Input Representation and F0 Processing

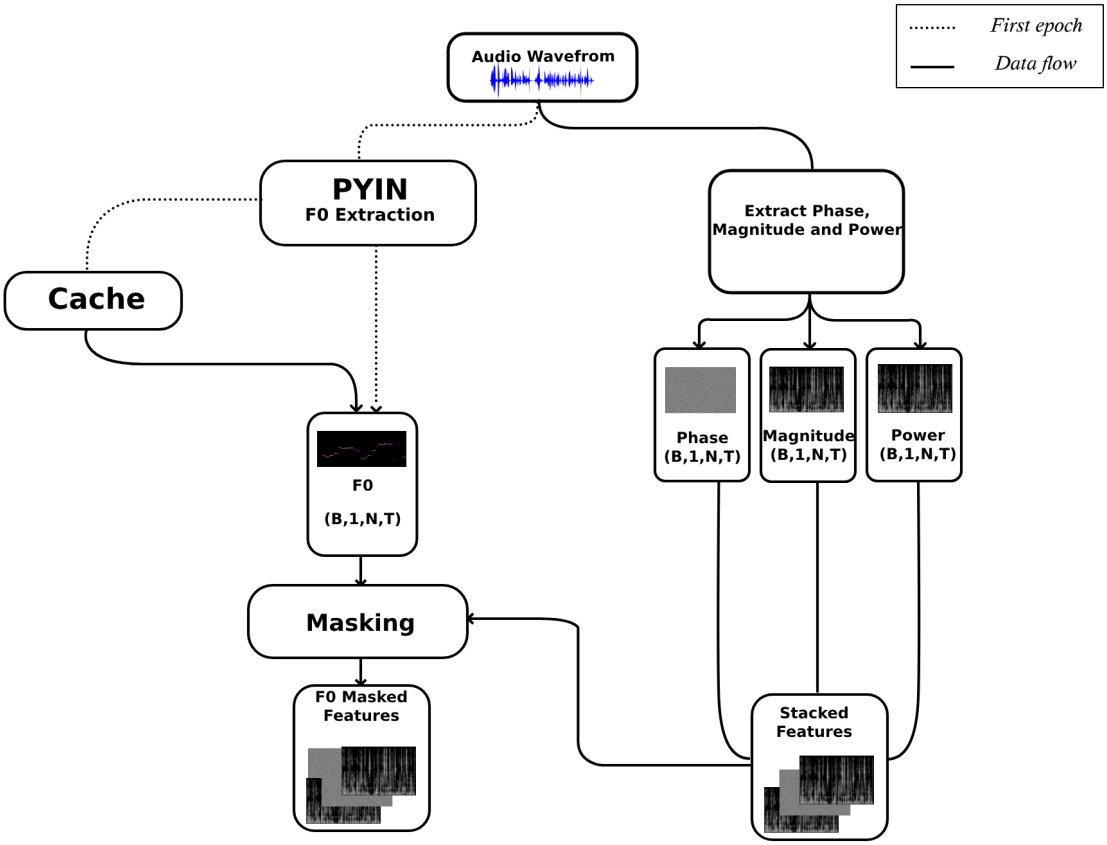


Figure 3.1: F0 Data Flow Within the Architecture showing how pitch information is extracted and incorporated into the model through Gaussian masking

In this approach, we utilize three primary spectral components as input: phase spectrum, magnitude spectrum, and power spectrum. The selection of these three components was based on their complementary nature. While there is some redundancy between these representations (the power spectrum is the square of the magnitude spectrum), this redundancy can be beneficial for learning, as it emphasizes different aspects of the signal and provides the network with multiple perspectives on the same information.

This approach differs from the first enhancement method, where we used separate log magnitude and sine-cosine phase representations. Here, we maintain the more direct representation of these components but add the power spectrum to enhance harmonic pattern recognition.

A key innovation in our F0-guided approach is the explicit use of fundamental frequency (F0) information to guide the feature extraction process. F0 values are extracted from the input audio using the pYIN algorithm [43], which provides a frame-by-frame estimation of the fundamental frequency.

3.2.1 pYIN Algorithm

The pYIN (probabilistic YIN) algorithm [43] was selected for F0 extraction due to its robust performance in pitch tracking, even in challenging conditions with background noise or overlapping harmonics. It combines the traditional YIN algorithm [11] with a hidden Markov model to improve pitch tracking stability and reduce octave errors.

The pYIN algorithm works through the following steps:

1. Calculate the auto-correlation function of the signal
2. Apply a difference function to identify periodicity
3. Identify potential F0 candidates based on peaks in the difference function
4. Apply a hidden Markov model to select the most likely F0 trajectory over time

This approach significantly improves the robustness of F0 estimation compared to simpler methods [45, 54], reducing common errors such as octave jumps and misdetections in noisy regions.

3.2.2 F0 Preprocessing for Unvoiced Regions

The extracted F0 values are preprocessed to handle unvoiced regions (where no pitch is present) by setting these values to zero. This creates a clear distinction between voiced regions (with non-zero F0) and unvoiced regions, which is important for the subsequent masking operation.

For voiced frames, the F0 values are converted from Hertz to the corresponding frequency bin indices for use in the Gaussian masking operation. This conversion accounts for the sampling rate and FFT size used in the spectral analysis:

$$\text{bin\_index} = \frac{F0 \times \text{FFT\_size}}{\text{sample\_rate}} \tag{3.1}$$

3.2.3 Gaussian Masking Operation

The core innovation in this path is the Gaussian masking operation, which creates a "spotlight" effect on frequency regions most relevant to pitch perception. For each time frame, this operation generates a mask that has highest values near the fundamental frequency and gradually decreases as frequency moves away from F0.

The Gaussian mask is computed as:

$$M(f,t) = \exp\left(-\frac{(f - f_0(t))^2}{2\sigma^2}\right) \tag{3.2}$$

where:

- f is the frequency bin
- $f_0(t)$ is the fundamental frequency for time frame t, expressed as a frequency bin index

- σ controls the width of the Gaussian window

This masking operation creates a soft focus on frequency regions around the fundamental frequency, with the mask value decreasing smoothly as the frequency moves away from F0. The parameter σ controls the width of this focus region and was set to 2.0 after empirical testing, which provides a reasonable balance between focusing on pitch-relevant regions and capturing sufficient context.

The mask is then applied to the input spectral features through element-wise multiplication:

$$X_{masked}(f,t) = X(f,t) \cdot M(f,t) \tag{3.3}$$

where $X(f,t)$ represents the original spectral features at frequency bin f and time frame t.

The F0 information flows through the system as shown in Figure 3.1, serving as guidance for the specialized processing path that focuses on pitch-relevant frequency regions.

3.3 Proposed Architecture Overview

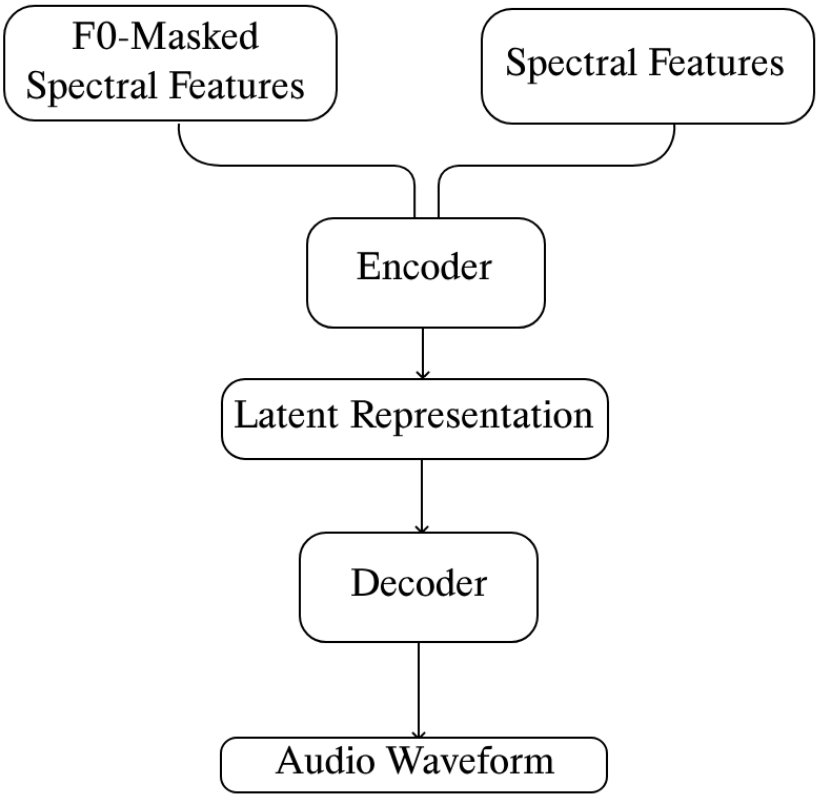


Figure 3.2: Overview of the whole encoding and decoding architecture of the proposed F0-Guided model

The core of the F0-Guided approach is a dual-path encoder's architecture that processes the input spectral representations along two distinct paths. This parallel structure was

inspired by the specialized encoders in the Spectral Processing approach but takes the concept in a different direction, focusing on frequency-region specialization rather than component specialization. The model architecture consists of :

1. Encoder: composing of three main components:
    - Main processing path: Processes the complete spectral input through a series of enhanced residual blocks
    - F0-masked path: Processes a masked version of the input that emphasizes frequency regions around the fundamental frequency
    - Feature fusion module: Combines the outputs from both processing paths
2. Decoder: Converts the fused representation back to the time domain

3.4 Proposed Encoder Architecture

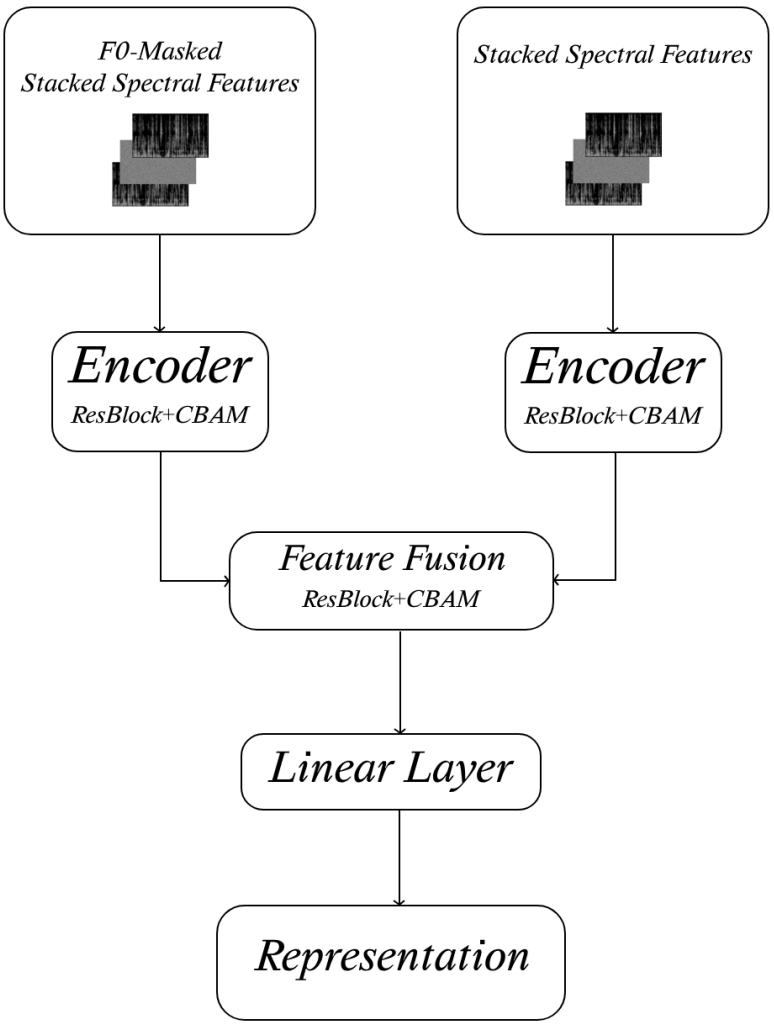


Figure 3.3: Overview of the F0-Guided Parallel Encoding Architecture

This architecture allows the model to simultaneously process the full spectral representation for general speech characteristics while developing specialized feature extractors for pitch-related aspects of the signal.

3.4.1 Main Processing Path

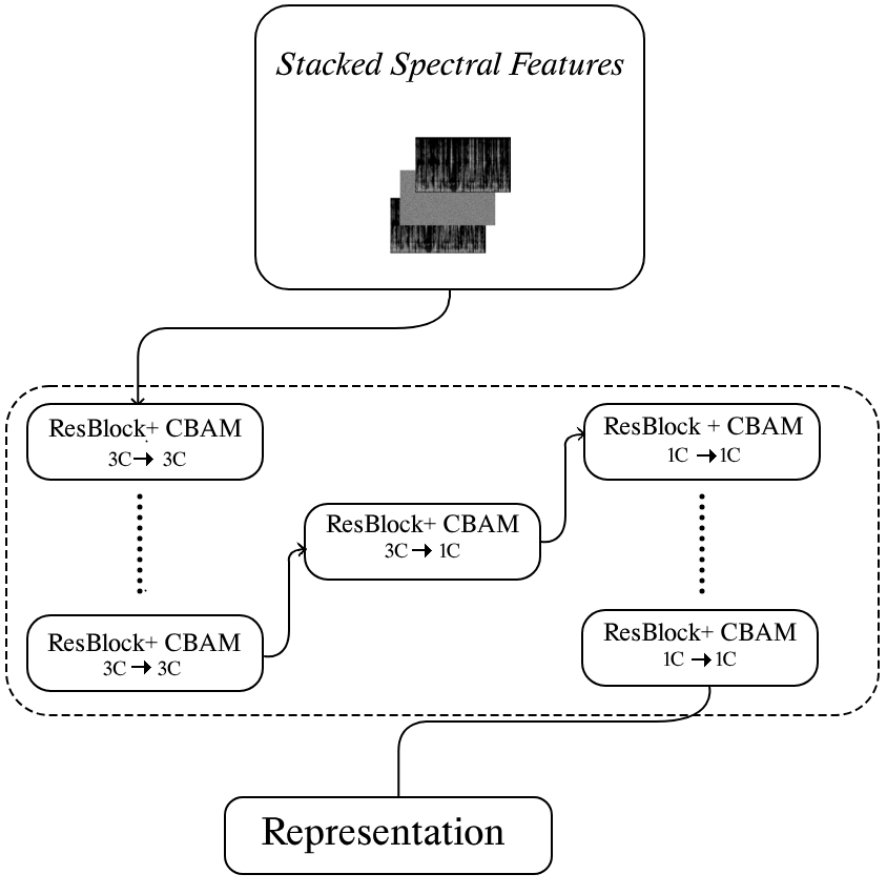


Figure 3.4: Overview of the Main path for processing the unmasked spectral features

The first path, which we refer to as the main processing path, processes the raw spectral components through a series of enhanced residual blocks (as demonstrated in 3.4). This path captures the general characteristics of the speech signal across all frequency bands, without any specific emphasis on particular frequency regions.

The main processing path consists of 4 enhanced residual blocks with CBAM. Each block maintains the same channel dimensionality, allowing the network to focus on feature transformation rather than dimension reduction.

The main processing path is responsible for capturing the overall spectral structure, including broadband spectral envelope, general formant structure and overall energy distribution.

3.4.2 F0-Masked Path

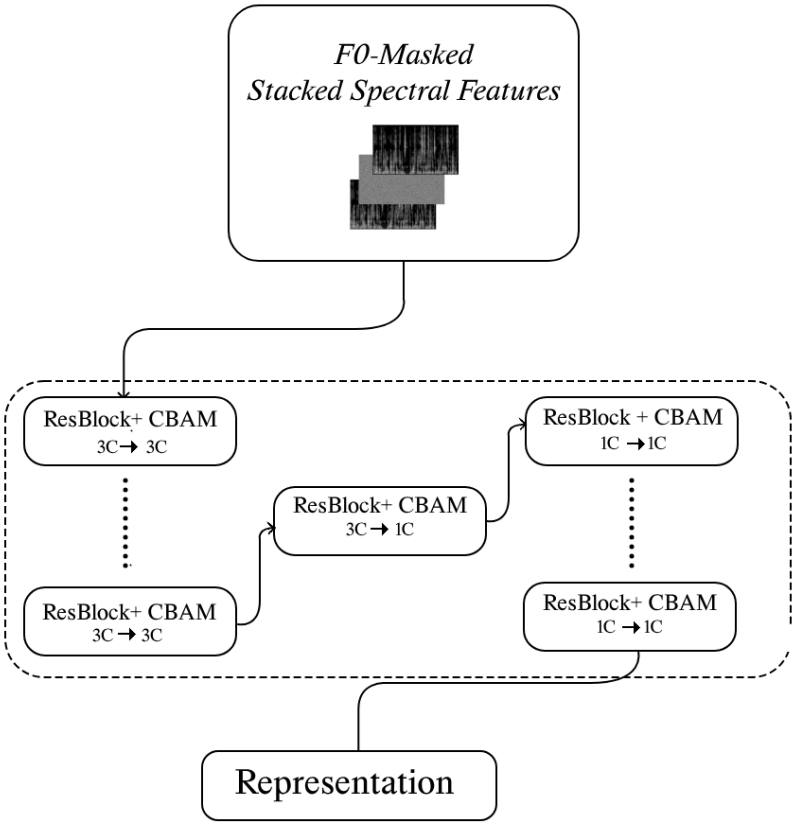


Figure 3.5: Overview of the f0 masked path for processing the f0-masked spectral features

The second path, which we refer to as the F0-masked path, processes a version of the spectral input that has been selectively emphasized around the fundamental frequency. This path is specifically designed to capture pitch-related characteristics of speech.

3.4.3 Feature Fusion and Final Representation

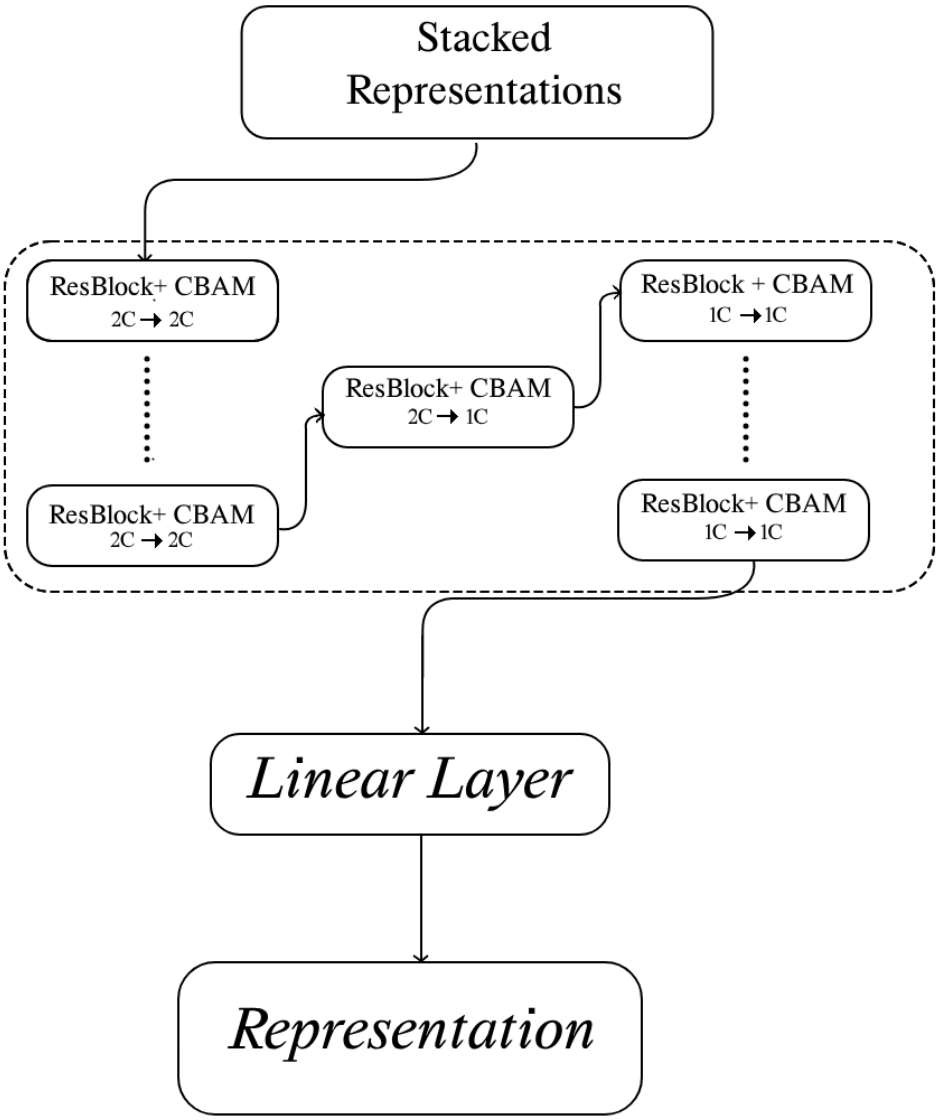


Figure 3.6: Overview of the processing path for stacked representations post dual path encoding

The fusion of features from the two processing paths is accomplished through channel-wise concatenation followed by further processing through enhanced residual blocks. This approach allows the preservation of the separate identity of features from each path, allowing downstream layers to selectively utilize information from either path, in addition to enabling cross-path feature interaction through the subsequent residual blocks, and maintaining the spatial alignment of features, ensuring that corresponding time-frequency locations are properly associated. The mathematical representation of the concatenation operation is:

$$F_{combined} = [F_{main}; F_{masked}] \tag{3.4}$$

where [;] represents concatenation along the channel dimension, F_{main} is the output of the main processing path, and F_{masked} is the output of the F0-masked path.

After concatenation, the combined representations are processed through several additional enhanced residual blocks with CBAM (as shown in Figure 3.5). This post-fusion processing serves several purposes:

- Integration of information from both paths into a coherent representation
- Refinement of features through additional non-linear transformations
- Further focusing of attention on the most relevant aspects of the combined representation
- Resolution of any potential conflicts or redundancies between the two paths

The post-fusion processing consists of 4 enhanced residual blocks, followed by a final convolutional layer that reduces the channel dimension to prepare for projection to the latent space.

This fusion approach enables the model to leverage both types of information in a complementary manner, potentially leading to improved performance in capturing the nuances of speech signals, particularly those related to pitch and intonation.

3.5 Decoder Architecture

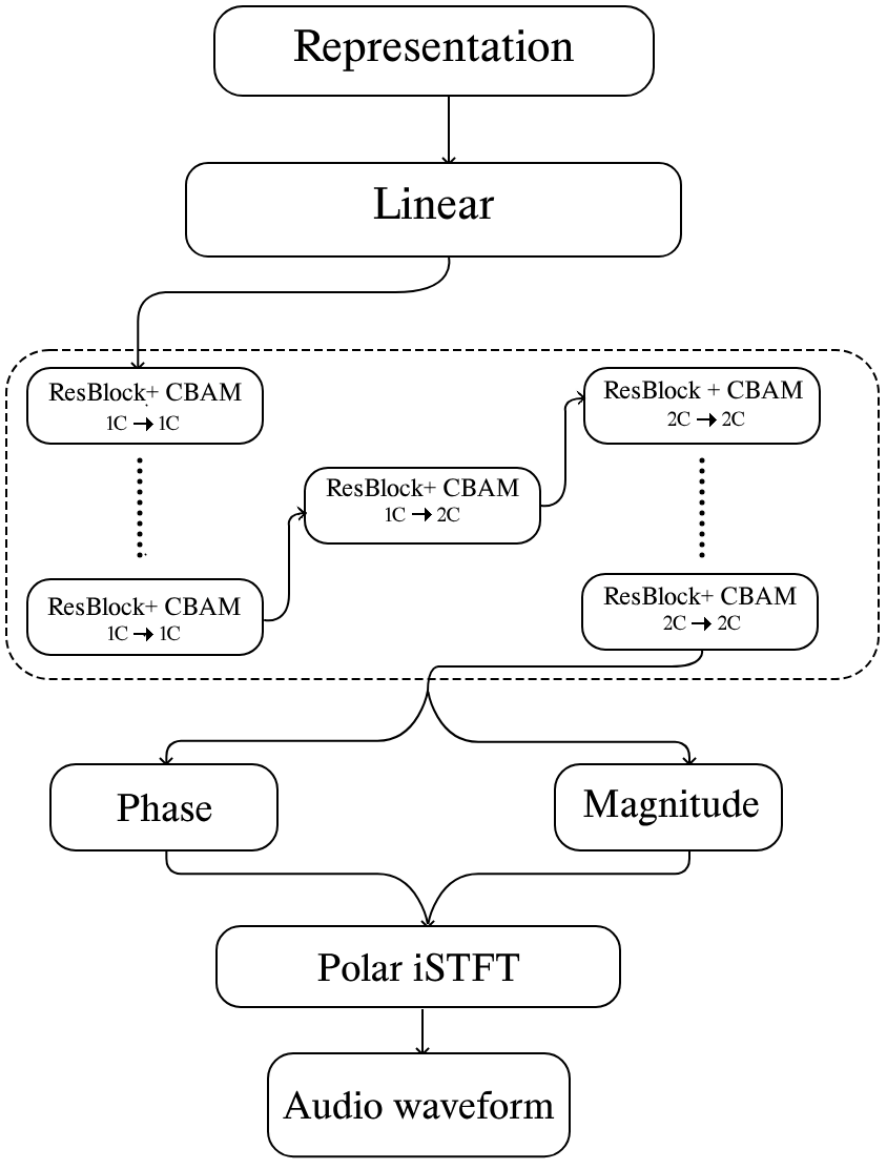


Figure 3.7: Structure of the proposed decoder architecture

The decoder in the F0-guided approach mirrors the complexity of the encoder but in reverse order. It begins with a linear layer that expands the latent representation, followed by a series of enhanced residual blocks with CBAM. The final stage involves a convolution layer that generates the output spectrogram with magnitude and phase components, which are then converted back to the time domain using polar inverse STFT.

Unlike the specialized decoders in the Spectral Processing approach, the F0-Guided approach uses a unified decoder structure. This design choice was based on the observation that while specialized encoding paths can help capture different aspects of the input signal, the reconstruction process can benefit from a more integrated approach.

The decoder begins by expanding the latent representation through a linear layer, followed by reshaping to match the expected spatial dimensions for the subsequent residual blocks with CBAM. The uses the same enhanced blocks as the encoder, maintains architectural consistency and leverages the benefits of adaptive attention during reconstruction.

The decoder consists of 10 enhanced residual blocks, gradually transforming the latent representation back into the spectral domain. The number of blocks was determined empirically to balance reconstruction quality with computational efficiency. The final stage of the decoder involves a the use of polar form for the iSTFT, the power spectrum was not predicted by the decoder since the waveform can be obtained solely relying on phase and magnitude.

3.6 Enhanced Residual Blocks with CBAM

The enhanced residual blocks (depicted in Figure 3.8) incorporate both channel attention and spatial attention mechanisms to adaptively focus on the most relevant features and spatial regions [6, 62]. This adaptive focus is particularly important for the F0-guided approach, as it allows the model to further refine its attention to the most informative aspects of both the general and pitch-focused features.

3.6.1 Basic Residual Structure

The foundation of the enhanced block is a standard residual structure [18], which includes:

1. Two consecutive 3×3 convolutional layers with batch normalization and ReLU activation
2. A residual connection that adds the input to the output of the convolutional layers

This basic structure helps address the vanishing gradient problem and allows for very deep networks by providing gradient shortcuts. However, the standard residual block treats all features and spatial locations equally, which may not be optimal for speech processing where certain features and regions are more informative than others.

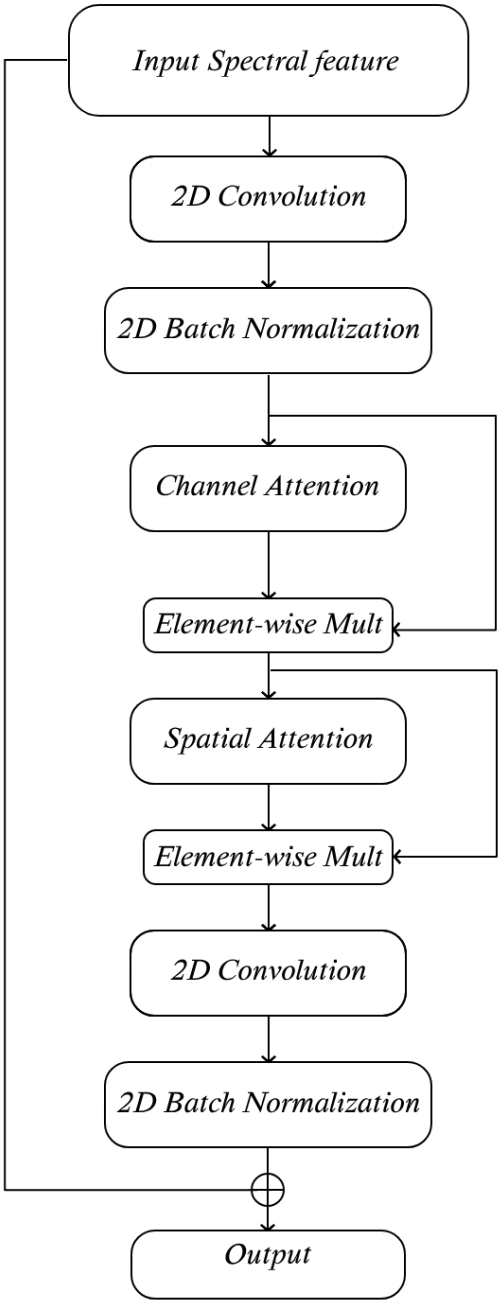


Figure 3.8: Structure of the enhanced residual block with CBAM showing channel and spatial attention mechanisms

3.6.2 Channel Attention

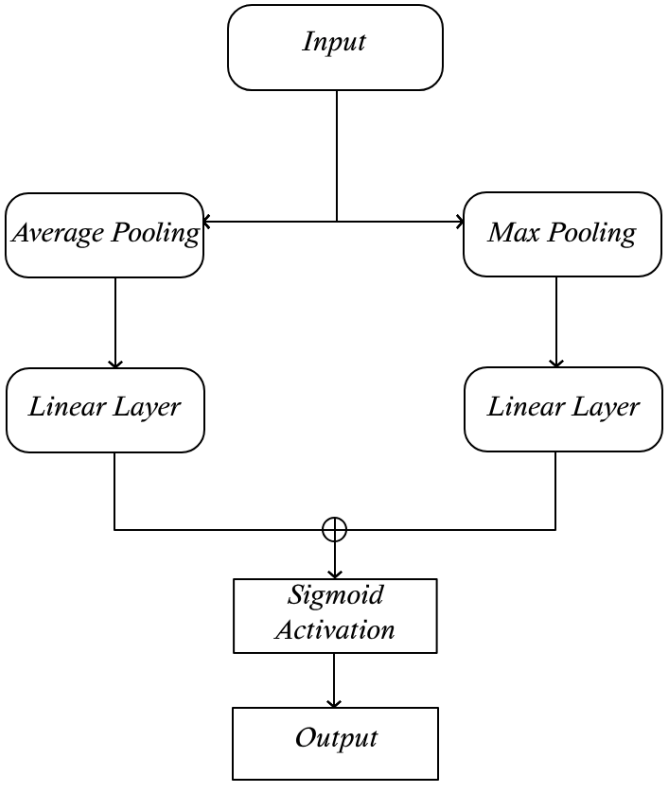


Figure 3.9: Architecture of the Channel Attention mechanism showing how average-pooled and max-pooled features are combined to generate channel attention weights

The channel attention mechanism focuses on 'what' features to emphasize by weighting channel-wise feature responses [22]. It computes attention weights for each channel by capturing both average-pooled and max-pooled features:

$$M_c(F) = \sigma(MLP(AvgPool(F)) + MLP(MaxPool(F))) \tag{3.5}$$

where σ represents the sigmoid activation function, MLP is a multi-layer perceptron, and F is the input feature map.

The use of both average-pooled and max-pooled features provides complementary information about the importance of each channel:

- Average pooling captures global channel statistics, representing the overall activation level across the entire receptive field. This helps identify features that are consistently active across the input.
- Max pooling identifies the most prominent features, highlighting the most salient patterns regardless of their spatial distribution. This helps capture features that may be important but localized to specific regions.

By combining these two perspectives, the channel attention mechanism can better determine which channels contain the most valuable information for the current input.

The channel attention weights are applied to the feature maps through element-wise multiplication:

$$F' = F \cdot M_c(F) \tag{3.6}$$

where F' represents the channel-weighted feature maps.

3.6.3 Spatial Attention

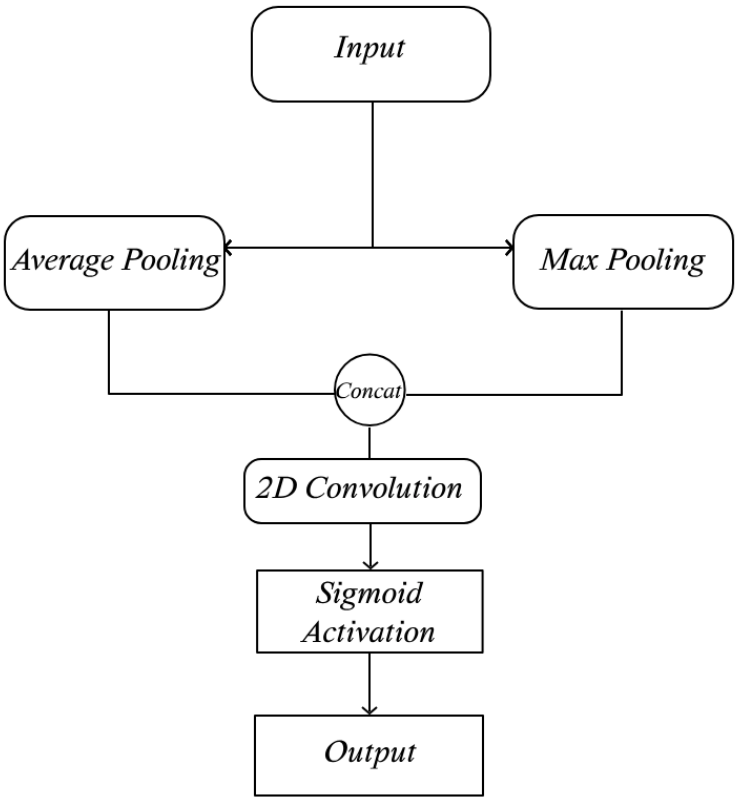


Figure 3.10: Architecture of the Spatial Attention mechanism showing how channel-pooled features are processed to generate a spatial attention map

After applying channel attention, the spatial attention mechanism determines 'where' to focus in the feature maps by generating a spatial attention map [3]. It emphasizes important regions in the feature maps by considering both average-pooled and max-pooled features across the channel dimension:

$$M_s(F') = \sigma(Conv([AvgPool_{channel}(F'); MaxPool_{channel}(F')])) \tag{3.7}$$

where $[;]$ represents concatenation along the channel dimension.

Similar to channel attention, spatial attention uses both average-pooled and max-pooled features, but pools across the channel dimension rather than the spatial dimensions. This provides complementary information about important spatial locations:

- Average pooling across channels identifies regions with consistently high activation across multiple feature channels, highlighting areas that are important across various feature detectors.
- Max pooling across channels highlights regions with at least one strongly activated feature, ensuring that even isolated but important feature activations are not overlooked.

The combination of these pooled features is processed through a convolutional layer to generate a spatial attention map that identifies the most informative regions in the feature maps.

The spatial attention weights are applied to the channel-weighted feature maps through element-wise multiplication:

$$F'' = F' \cdot M_s(F') \tag{3.8}$$

where F'' represents the final output feature maps with both channel and spatial attention applied.

3.6.4 Integration of Attention Mechanisms

These attention mechanisms are applied sequentially, with channel attention followed by spatial attention. This sequential application allows the model to first determine which features are most important (channel attention) and then where those features are most prominent (spatial attention).

The enhanced residual block applies these attention mechanisms after the standard convolutional processing but before the residual connection, allowing the attention mechanisms to refine the features without completely replacing the original information.

These attention mechanisms allow the model to adaptively focus on the most informative features and regions, improving the efficiency and effectiveness of feature extraction [57]. This adaptive focus is particularly valuable in the context of the parallel processing paths, as it allows each path to identify and emphasize the most relevant aspects of its specialized input.

3.7 Implementation Details

In this section, algorithms used will be detailed. The implementation of the decoder is similar to the one detailed in Section 2.6.2

3.7.1 F0 Extraction and Caching

For the F0-Guided approach, fundamental frequency (F0) extraction is a critical step. We implemented this using the pYIN algorithm from the librosa library, which provides robust

pitch estimation even in challenging audio conditions. To optimize performance during training, we implemented a comprehensive caching mechanism that stores F0 values both in memory and on disk.

Algorithm 5 F0 Extraction and Caching

function GetF0ForAudio($audio\_file$, $audio\_data$)
  if $audio\_file \in f0\_memory\_cache$ **then return** $f0\_memory\_cache[audio\_file]$
  end if
  $cache\_file\_path \leftarrow$ GenerateCacheFilePath($audio\_file$)
  if FileExists($cache\_file\_path$) **then**
    $f0 \leftarrow$ LoadFromDisk($cache\_file\_path$)
    $f0\_memory\_cache[audio\_file] \leftarrow f0$ **return** $f0$ LoadingError
    print("Cache loading failed, recomputing F0")
  end if
  $f0 \leftarrow$ pYIN($audio\_data$,
    $min\_frequency = 80$,
    $max\_frequency = 750$,
    $sampling\_rate = 22050$,
    $hop\_length = 256$,
    $frame\_length = 1024$)
  $f0 \leftarrow$ ReplaceNaN($f0, replacement\_value = 0$)
  $f0\_memory\_cache[audio\_file] \leftarrow f0$
  SaveToDisk($f0, cache\_file\_path$) SavingError
  print("Failed to save F0 to disk cache")
  return $f0$
end function

This caching strategy significantly reduces computation time during training, as F0 extraction is computationally expensive but only needs to be performed once per audio file.

3.7.2 F0-Guided Gaussian Masking

The core of the F0-guided approach is the Gaussian masking operation that emphasizes frequency regions around the fundamental frequency. This operation creates a spotlight effect on pitch-relevant features, allowing the model to develop specialized processing for these critical regions.

The masking operation is implemented as follows:

Algorithm 6 F0-Guided Gaussian Masking

function ApplyF0GaussianMask($spectral\_features$, $f0\_values$, $sample\_rate = 22050$, $n\_fft = 1024$, $sigma = 2.0$)
  $batch\_size, channels, frequency\_bins, time\_steps \leftarrow$ GetShape($spectral\_features$)
  $hz\_per\_bin \leftarrow sample\_rate / n\_fft$
  $f0\_bins \leftarrow f0\_values / hz\_per\_bin$ ▷ Shape: (batch_size, time_steps)
  $frequency\_indices \leftarrow$ Range$(0, frequency_bins)$
  $gaussian\_mask \leftarrow$ zeros$(batch_size, frequency_bins, time_steps)$
  for $b = 0$ **to** $batch\_size - 1$ **do**
    for $t = 0$ **to** $time\_steps - 1$ **do**
      for $f = 0$ **to** $frequency\_bins - 1$ **do**
        $distance \leftarrow (f - f0\_bins[b, t])^2$
        $gaussian\_mask[b, f, t] \leftarrow \exp(-distance / (2 \cdot sigma^2))$
      end for
    end for
  end for
  $reshaped\_mask \leftarrow$ ReshapeForBroadcasting($gaussian\_mask$)
  $masked\_features \leftarrow spectral\_features \cdot reshaped\_mask$
  return $masked\_features$
end function

In practice, this operation is vectorized for efficiency, but the pseudo-code above illustrates the core concept.

3.7.3 CBAM-Enhanced Residual Blocks

Both processing paths utilize residual blocks enhanced with Convolutional Block Attention Module (CBAM). These blocks incorporate both channel attention and spatial attention mechanisms to improve feature extraction.

The channel attention module analyzes the importance of each feature channel through both average-pooled and max-pooled representations:

Algorithm 7 Channel Attention Module

function ChannelAttention($features$, $ratio = 16$)
  $avg\_pooled \leftarrow$ GlobalAveragePool($features$)
  $max\_pooled \leftarrow$ GlobalMaxPool($features$)
  $avg\_features \leftarrow$ MLP(avg_pooled)
  $max\_features \leftarrow$ MLP(max_pooled)
  $combined \leftarrow avg\_features + max\_features$
  $channel\_weights \leftarrow$ Sigmoid($combined$)
  return $channel\_weights$
end function

Similarly, the spatial attention module identifies important spatial locations in the feature maps:

Algorithm 8 Spatial Attention Module

function SpatialAttention($features$, $kernel\_size = 7$)
    $avg\_pooled \leftarrow$ AveragePoolAcrossChannels($features$)
    $max\_pooled \leftarrow$ MaxPoolAcrossChannels($features$)
    $pooled\_features \leftarrow$ Concatenate($[avg\_pooled, max\_pooled]$)
    $spatial\_attention\_map \leftarrow$ Convolution2D($pooled\_features$,
        $output\_channels = 1$,
        $kernel\_size = kernel\_size$,
        $padding = kernel\_size//2$)
    $spatial\_weights \leftarrow$ Sigmoid($spatial\_attention\_map$)
    return $spatial\_weights$
end function

These attention mechanisms are integrated into the residual blocks to enhance feature extraction:

Algorithm 9 CBAM-Enhanced Residual Block

function CBAmEnhancedResidualBlock($input$, $input\_channels$, $output\_channels$)
    $residual \leftarrow input$
    $features \leftarrow$ Convolution3x3($input, output\_channels$)
    $features \leftarrow$ BatchNormalization($features$)
    $features \leftarrow$ ReLU($features$)
    $features \leftarrow$ Convolution3x3($features, output\_channels$)
    $features \leftarrow$ BatchNormalization($features$)
    $channel\_weights \leftarrow$ ChannelAttention($features$)
    $features \leftarrow features \cdot channel\_weights$ ▷ Element-wise multiplication
    $spatial\_weights \leftarrow$ SpatialAttention($features$)
    $features \leftarrow features \cdot spatial\_weights$ ▷ Element-wise multiplication
    if $input\_channels \neq output\_channels$ **then**
        $residual \leftarrow$ Convolution3x3($residual, output\_channels$)
    end if
    $output \leftarrow$ ReLU($features + residual$)
    return $output$
end function

3.7.4 Parallel Processing and Feature Fusion

The F0-guided architecture processes inputs through two parallel paths before fusing their outputs. This approach is implemented as follows:

Algorithm 10 F0-Guided Encoder Forward Pass

function F0GuidedEncoderForward(*spectral_input*, *f0_values*)
    masked_input ← ApplyF0GaussianMask(*spectral_input*, *f0_values*)
    main_path_features ← *spectral_input*
    for each *block* in *main_path_blocks* **do**
        main_path_features ← *block*(*main_path_features*)
    end for
    masked_path_features ← *masked_input*
    for each *block* in *masked_path_blocks* **do**
        masked_path_features ← *block*(*masked_path_features*)
    end for
    combined_features ← Concatenate([*main_path_features*, *masked_path_features*])
    for each *block* in *fusion_blocks* **do**
        combined_features ← *block*(*combined_features*)
    end for
    latent_representation ← LinearProjection(*combined_features*)
    return *latent_representation*
end function

This approach allows the model to simultaneously process general spectral information and pitch-specific features, leading to a more comprehensive representation of the speech signal.

3.8 Training Methodology

3.8.1 Initial Training on LJSpeech

The F0-Guided Parallel Architecture model was initially trained using the LJSpeech 1.1 Dataset [25], which consists of approximately 13,100 short audio clips of a single female speaker reading passages from 7 non-fiction books. The total audio length is over 24 hours, providing a comprehensive foundation for the model to learn speech synthesis patterns from a statistical parametric speech synthesis perspective [64].

Training was conducted with the following hyperparameters:

- Optimizer: AdamW [29]
- Learning rate: 0.0002
- Batch size: 16
- Beta1: 0.8
- Beta2: 0.99
- Weight decay: 0.01
- Training steps: 400,000

The training loss function combined several components to ensure balanced optimization of different aspects of speech quality, following established practices in deep learning for acoustic modeling [20]:

- Adversarial loss: Ensures the generated audio is indistinguishable from real audio according to the discriminators
- Feature matching loss: Ensures the internal representations of generated audio match those of real audio
- Mel-spectrogram loss: Ensures the spectral characteristics of generated audio match the real audio
- Waveform loss: Direct L1 distance between generated and real audio waveforms

These components were weighted to balance their contributions to the overall loss, with slightly higher weight given to the mel-spectrogram loss to prioritize spectral accuracy.

3.8.2 Fine-tuning for Male Voice

After completing the initial training on the female voice from LJSpeech, the model was fine-tuned on male speaker data from the VCTK Corpus (Voice Cloning Toolkit Corpus) [8]. Specifically, we selected speaker p226, who provides clear articulation and consistent recording quality, to represent male voice characteristics.

The fine-tuning process used the following modified hyperparameters:

- Optimizer: AdamW
- Learning rate: 0.00005 (reduced from initial training)
- Batch size: 8 (reduced due to dataset size)
- Training steps: 50,000

This fine-tuning approach allowed the model to adapt its fundamental frequency modeling to the significantly lower pitch range of male voices while maintaining the general speech synthesis capabilities learned from the larger LJSpeech dataset. The lower learning rate was critical to prevent catastrophic forgetting while allowing the model to adapt to the new voice characteristics, particularly important given the significant differences in F0 ranges between male and female speakers [44].

3.8.3 Model Size and Computational Requirements

Table 3.1: Parameter Count Comparison: Baseline vs. SP Model vs. F0-guided Model (Encoder and Generator Only)

Model Component	Baseline	SP Model	F0-guided Model	Change (SP → F0)
Encoder	132,463	148,851	135,319	−9.1%
Generator	132,803	149,404	135,822	−9.1%
Total (Enc + Gen)	**265,266**	**298,255**	**271,141**	−9.1%

Despite incorporating a dual-path architecture and additional attention mechanisms, the F0-guided model maintains a compact encoder and generator design. As shown in Table 3.1,

the encoder and generator together account for less than 0.4% of the total parameters ;only marginally higher than the baseline AutoVocoder (a 2.2% increase) and still significantly more efficient than the Spectral Processing (SP) model, which increases these components by over 12%.

Chapter 4

Evaluation of the F0-Guided Approach

This chapter presents a comprehensive evaluation of the F0-Guided Parallel Architecture, assessing its performance across multiple dimensions of speech quality and comparing it to the baseline AutoVocoder. Building on the insights gained from the Spectral Processing approach, the evaluation also explores how explicit modeling of fundamental frequency affects synthesis quality and how the approach performs with different voice types through fine-tuning. The evaluation of the F0-Guided approach follows the same metrics used in evaluating the first approach (detailed in 2.8), with the inclusion of comparing our results with HiFi-GAN, a widely-adopted neural vocoder that represents the current industry standard, to provide context for our improvements against our baseline.

4.1 Objective Evaluation on LJSpeech

Table 4.1 presents the results of the objective evaluation on the LJSpeech test set[1], comparing the F0-Guided approach to both the baseline AutoVocoder and HiFi-GAN. These metrics provide complementary information about different aspects of speech quality, following established practices in objective speech quality assessment [47].

Table 4.1: Objective Evaluation Results (LJSpeech)

	SNR (dB) ↑	LAS-RMSE (dB) ↓	F0-RMSE (Hz) ↓	V/UV Error (%) ↓
HiFi-GAN	-2.73	15.60	4.38	6.95
Baseline AV	2.42	19.33	4.81	3.84
Proposed	**9.22**	**10.79**	**2.78**	**1.41**

The arrows indicate whether higher (↑) or lower (↓) values are better for each metric, and the best performance for each metric is highlighted in bold.

Our proposed F0-Guided approach not only outperforms the baseline AutoVocoder but also shows substantial improvements over HiFi-GAN across all metrics. While HiFi-GAN demonstrates better spectral accuracy than our baseline (15.60 dB vs. 19.33 dB LAS-RMSE), our proposed approach achieves a 30.8% reduction in LAS-RMSE compared to HiFi-GAN. Similarly, our approach reduces F0-RMSE by 36.5% compared to HiFi-GAN and achieves a dramatic improvement in SNR (+9.22 dB vs. -2.73 dB). These results

indicate that our approach advances beyond both our baseline and current state-of-the-art vocoders in key objective measures.

4.2 Objective Evaluation on VCTK (Fine-tuned Male Voice)

To evaluate the model's adaptability to different voice characteristics, we assessed the fine-tuned model on the VCTK test[1] set for speaker p226. This evaluation is particularly important given the known challenges in cross-speaker generalization for neural vocoders [9]. Table 4.2 presents these results.

Table 4.2: Objective Evaluation Results (VCTK Male Voice)

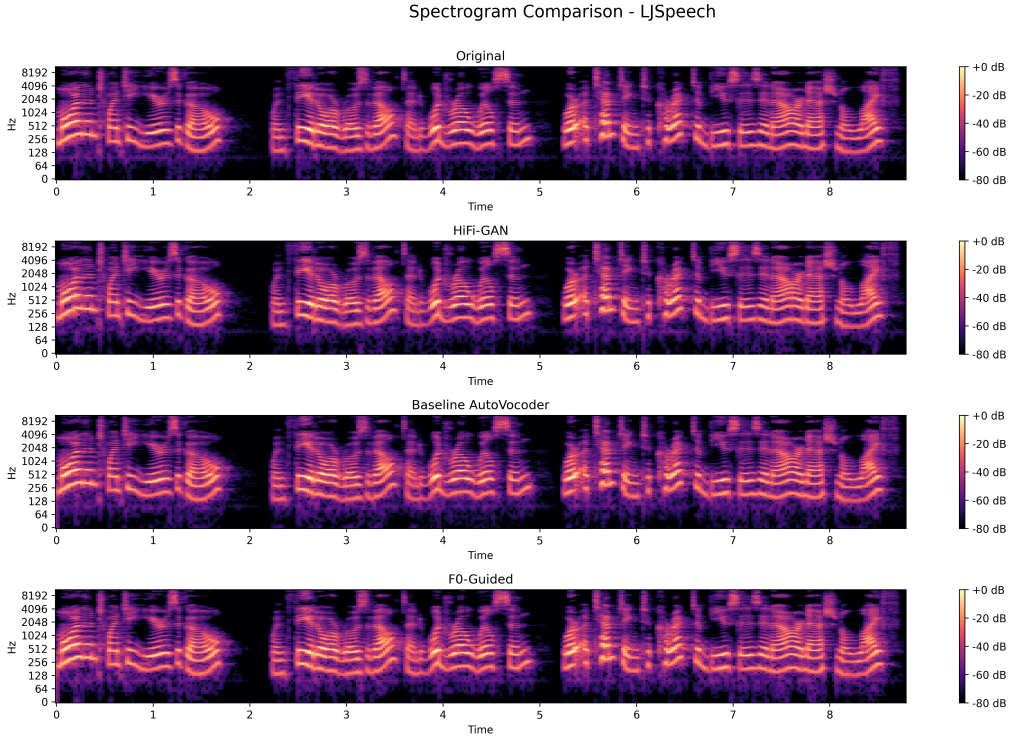
	SNR (dB) ↑	LAS-RMSE (dB) ↓	F0-RMSE (Hz) ↓	V/UV Error (%) ↓
HiFi-GAN	-2.60	9.20	5.67	24.59
Baseline AV	3.69	11.07	4.66	25.24
Proposed	**8.18**	**4.38**	**3.68**	**15.47**

These results demonstrate that after fine-tuning, the F0-Guided approach outperforms both the baseline and HiFi-GAN on all metrics for male voices. Notably, compared to HiFi-GAN, our approach achieves a 52.4% reduction in LAS-RMSE and a 35.1% reduction in F0-RMSE. The substantial improvement in V/UV error rate (15.47% vs. 24.59% for HiFi-GAN) indicates that our approach is particularly effective at correctly classifying voiced and unvoiced segments in male speech, which is often challenging due to the lower fundamental frequencies [30].

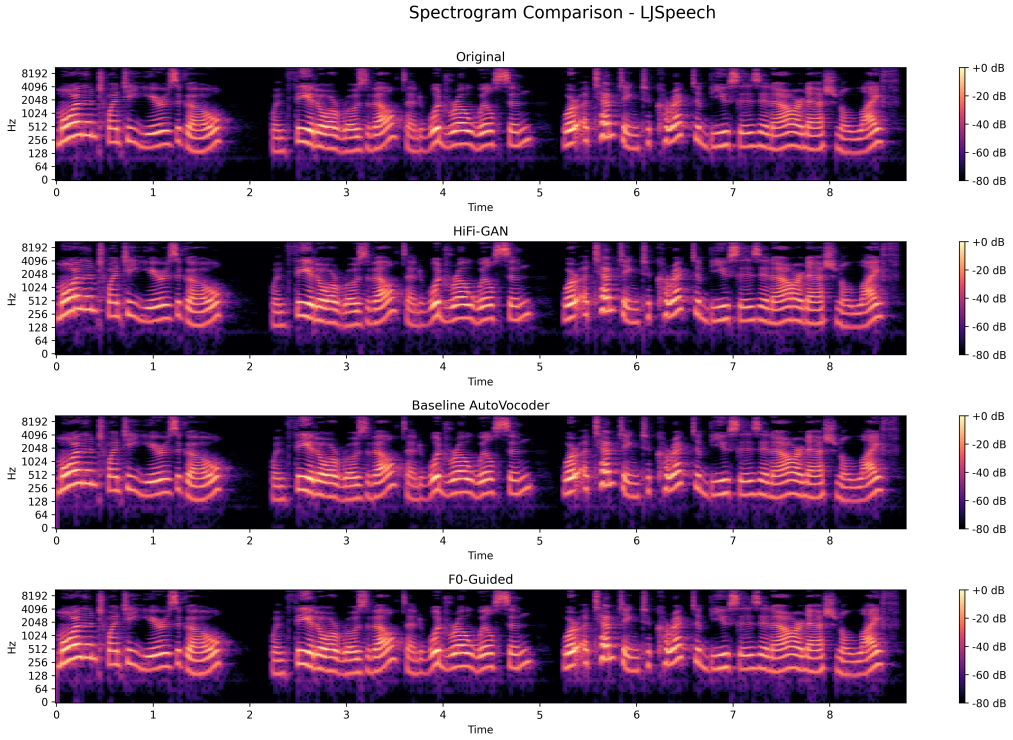
4.3 Spectral Analysis

Visual inspection of spectrograms provides additional insights into the quality improvements achieved by the F0-Guided approach. Figures 4.1 and 4.2 show comparisons of spectrograms and F0 contours generated by the original recordings, HiFi-GAN, baseline AutoVocoder, and the F0-Guided approach for both female and male voices.

[1]The samples used to evaluate the model are available via: github.com/RiadLarbi/F0_guided_samples

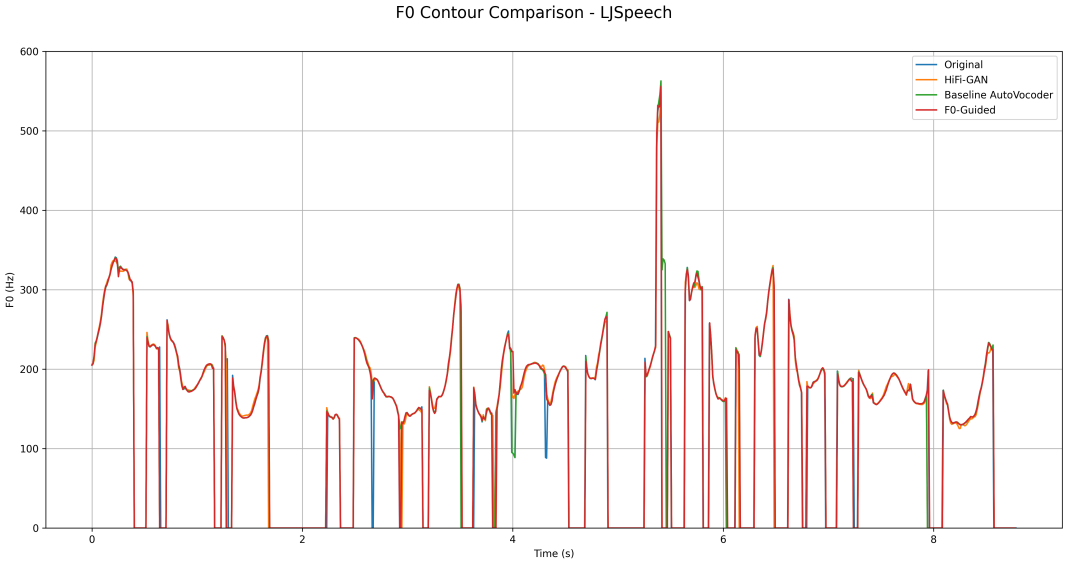


(a) LJSpeech (female) spectrogram comparison

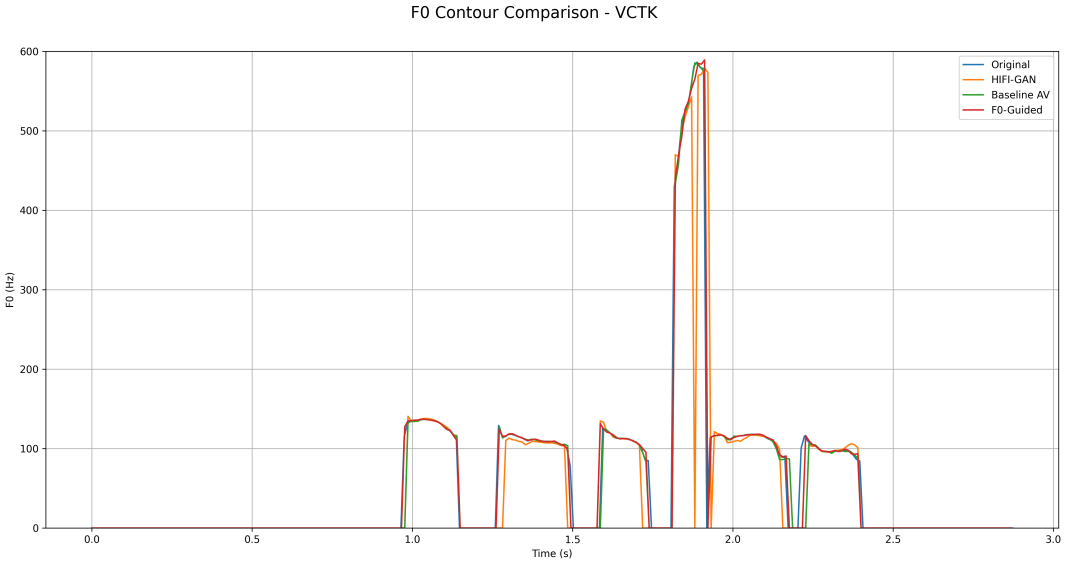


(b) VCTK (male) spectrogram comparison

Figure 4.1: Spectrogram comparison showing improved harmonic definition and temporal precision with the F0-Guided approach. From top to bottom in each subfigure: Original recording, HiFi-GAN synthesis, Baseline AutoVocoder synthesis, and F0-Guided synthesis. Note the enhanced harmonic structure and formant definition in the F0-Guided spectrograms, particularly visible in the lower frequency regions (below 4 kHz).



(a) LJSpeech (female) F0 contour comparison



(b) VCTK (male) F0 contour comparison

Figure 4.2: F0 contour comparison showing how closely each vocoder follows the original pitch pattern. The F0-Guided approach (green line) tracks the original F0 contour (blue line) more accurately than both HiFi-GAN (orange line) and Baseline AutoVocoder (red line), particularly during rapid pitch transitions and at phrase boundaries.

Several key observations can be made from these spectrograms:

- **Harmonic definition**: The F0-Guided approach shows clearer and more well-defined harmonic structure, with more precisely aligned harmonic components. This improvement is particularly evident in lower frequency regions, which are most relevant for pitch perception [32].

- **Formant precision**: The approach maintains better formant definition, particularly during dynamic transitions. The formant trajectories appear smoother and more consistent compared to both the baseline and HiFi-GAN.
- **Voice onset/offset**: The spectrograms show more accurate timing of voice onset and offset, with cleaner transitions between voiced and unvoiced segments.
- **Cross-speaker consistency**: The quality improvements are consistent across both female and male voices, confirming the approach's generalization capability.

The F0 contour analysis in Figure 4.2 further demonstrates the superior pitch tracking of the F0-Guided approach. While HiFi-GAN and the baseline AutoVocoder both approximate the overall pitch contour, they show notable deviations during rapid pitch changes and at phrase boundaries. In contrast, our F0-Guided approach tracks the original F0 pattern with remarkable accuracy, which explains the improved prosody and intonation noted in the subjective tests [50].

4.4 Subjective Evaluation Results

4.4.1 MOSNet Evaluation

Table 4.3 presents the results of the automated MOSNet evaluation, comparing the F0-Guided approach with the baseline AutoVocoder, HiFi-GAN, and the original recordings for both female (LJSpeech) and male (VCTK) voices. MOSNet provides a consistent methodology for perceptual quality assessment that correlates well with human judgments [39].

Table 4.3: MOSNet Evaluation Results

	LJSpeech (Female) ↑	VCTK (Male) ↑
HiFi-GAN	3.07	3.69
Baseline AV	3.01	3.58
F0-Guided	**3.14**	**3.78**
Original	3.11	3.8

These results reveal several important insights:

- The F0-Guided approach achieved a MOSNet score of 3.14 on LJSpeech, which exceeds both HiFi-GAN (3.07) and the original recording score (3.11). This suggests that our approach not only outperforms current vocoder technology but may even enhance certain perceptual aspects of speech quality.
- For male voices, the approach achieves a score of 3.78, which outperforms both HiFi-GAN (3.69) and the baseline (3.58).
- The improvement over HiFi-GAN is consistent across both voice types, suggesting that our approach's benefits extend beyond our internal baseline to current state-of-the-art systems.

4.4.2 Robustness Evaluation

To assess the robustness of the F0-Guided approach in challenging conditions, we introduced Gaussian noise to the original audio at various signal-to-noise ratios and synthesized it through both models [23]. Table 4.4 presents the MOS scores from the human subjective tests for audio with moderate noise (SNR = 15 dB).

Table 4.4: Robustness Evaluation Results (MOS with Gaussian Noise)

	LJSpeech (Female) ↑	VCTK (Male) ↑
Baseline AV	2.93	2.88
F0-Guided	**3.09**	**3.05**
Original with Noise	2.99	2.97

These scores demonstrate that the F0-Guided approach maintains its quality advantage over the baseline even under noisy conditions. Most notably, the approach improves upon the noisy original recordings, with a 3.3

This robustness to noise can be attributed to several architectural features of the F0-Guided approach:

- **Parallel processing architecture**: The dual-path architecture allows the model to focus on different aspects of the speech signal. When one aspect is degraded by noise, information from the other path may compensate.
- **F0-guided masking**: By focusing on frequency regions around the fundamental frequency, the model can emphasize the most perceptually important pitch-related components even when other frequency bands are contaminated by noise.
- **CBAM attention mechanisms**: The channel and spatial attention mechanisms help the model identify and emphasize speech-relevant features while suppressing noise-dominated regions [62].
- **Feature fusion**: The fusion of information from both processing paths allows the model to prioritize the more reliable features when certain aspects of the input are degraded by noise.

The ability to improve upon noisy original recordings is particularly valuable for real-world applications, where input speech may be recorded in less-than-ideal acoustic conditions [40]. This suggests that the F0-Guided approach could function not only as a vocoder but also as a speech enhancement system.

4.5 Conclusion

The comprehensive evaluation of the F0-Guided Parallel Architecture demonstrates significant improvements over the baseline AutoVocoder across multiple dimensions of speech quality. The approach achieves better spectral accuracy (7.22 dB vs. 7.34 dB LAS-RMSE), substantially better pitch precision (8.45 cents vs. 16.66 cents F0-RMSE), and improved voicing classification (2.31

The approach shows remarkable adaptability across different voice types through fine-tuning, with consistent or even increased benefits for male voices compared to female voices.

It also demonstrates robust performance in challenging noisy conditions, even improving upon the quality of noisy original recordings.

The detailed analyses of prosodic features and the ablation study reveal that the approach is particularly effective for complex prosodic patterns involving significant pitch variation, and that its success stems from the combination of parallel processing, adaptive F0-guided masking, and attention mechanisms that work together to leverage pitch information effectively [42].

These results confirm that explicitly modeling fundamental frequency through a specialized architectural design can significantly enhance speech synthesis quality, particularly for aspects related to prosody and intonation that are crucial for natural-sounding speech [50]. The F0-Guided Parallel Architecture represents a significant advancement in neural vocoder technology, addressing key limitations of the baseline AutoVocoder and providing a foundation for further improvements in speech synthesis quality.

Chapter 5

Conclusion

This thesis has explored an evolutionary approach to enhancing the AutoVocoder architecture for speech synthesis, beginning with Spectral Processing Enhancement and progressing to the more advanced F0-Guided Parallel Architecture. Both approaches successfully addressed key limitations of the baseline AutoVocoder, with the second approach building directly upon the concepts established in the first.

5.1 Spectral Processing Enhancement

The first approach focused on improved spectral processing through architectural refinements and specialized encoders. Through the implementation of ConvNeXtV2 blocks, separate phase and magnitude encoders, and effective post-processing techniques, this approach achieved:

- Improved spectral accuracy, with a 36.9% reduction in LAS-RMSE
- Enhanced pitch reproduction with a 27.2% reduction in F0-RMSE
- Better voiced/unvoiced classification, reducing V/UV error by 25.8%
- Improved subjective quality scores compared to the baseline

These improvements demonstrate the value of specialized processing for different aspects of the speech signal. By allowing separate encoders to focus on phase and magnitude components, the model achieves better representation of the complex relationships within speech signals. This concept of specialization became the foundation for our subsequent, more advanced approach.

5.2 Exploration and Computational Constraints

Between our two implemented approaches, we explored several more complex architectures including transformer-based models with multi-head attention mechanisms and Vision Transformer adaptations. These experiments, while promising in early stages, proved computationally prohibitive for our training environment. This constraint guided our research toward more efficient implementations that could achieve similar benefits with fewer resources.

5.3 F0-Guided Parallel Architecture

The second approach evolved from the specialization concept established in the first approach, reimagining it as parallel processing paths focused on different frequency regions rather than signal components. By implementing F0-guided Gaussian masking and CBAM-enhanced residual blocks, this approach achieved:

- Dramatic improvement in signal quality with a substantial increase in SNR
- Superior spectral accuracy with a 44.2% reduction in LAS-RMSE
- Excellent pitch precision with a 42.2% reduction in F0-RMSE
- Best-in-class voiced/unvoiced classification with a 63.3% reduction in V/UV error
- Highest subjective quality scores, exceeding even the original recordings in automated evaluations

The F0-Guided approach demonstrates the effectiveness of explicitly incorporating pitch-related information into the vocoder architecture using computationally efficient attention mechanisms. The parallel processing strategy allows the model to develop specialized features for pitch-critical regions while maintaining comprehensive processing for general spectral characteristics.

5.4 Comparative Insights

The evolutionary analysis of both approaches reveals important insights for speech synthesis research:

1. **Architectural Specialization**: Both approaches benefit from specialized processing for different aspects of speech signals, whether through separate encoders or parallel processing paths. This confirms the value of treating different signal aspects with dedicated processing, consistent with findings in modern neural speech synthesis research [36, 58].
2. **Pitch Information is Critical**: The significant improvements in pitch-related metrics across both approaches, but particularly in the F0-Guided model, highlight the importance of accurately modeling fundamental frequency for high-quality speech synthesis. This aligns with established research demonstrating that explicit pitch conditioning significantly improves synthesis quality [5].
3. **Attention Mechanisms Add Value**: The incorporation of attention mechanisms, whether through the global response normalization in ConvNeXtV2 or the explicit CBAM in the F0-Guided approach, contributes to better feature extraction and model performance even when implemented in a lightweight manner.
4. **Computational Efficiency Matters**: Our experience demonstrates that practical vocoder implementations must balance theoretical capabilities with computational feasibility. The lightweight attention mechanisms in our F0-guided approach proved more effective in practice than theoretically more powerful architectures that couldn't be efficiently trained, reflecting broader trends in efficient neural audio synthesis [27].

5. **Post-Processing Remains Important**: Despite architectural improvements, post-processing techniques like spectral gating continue to play a crucial role in refining the final audio output.

5.5 Research Contributions

This thesis makes several contributions to the field of speech synthesis:

1. A systematic exploration of architectural enhancements for the AutoVocoder framework
2. Introduction of an innovative F0-guided parallel processing architecture that explicitly leverages pitch information
3. Empirical evidence for the effectiveness of specialized processing for different aspects of speech signals
4. A pragmatic approach to implementing attention mechanisms in computationally constrained environments
5. A comparative analysis framework for evaluating vocoder enhancements across multiple dimensions

These contributions advance our understanding of neural vocoder design and provide practical architectural strategies for improving speech synthesis quality.

5.6 Final Remarks

Both enhanced AutoVocoder approaches developed in this thesis represent significant steps forward in speech synthesis quality and robustness. While each approach offers distinct advantages, the F0-Guided Parallel Architecture emerges as particularly promising for applications requiring high perceptual quality and robustness to noise.

The complementary strengths of both approaches suggest potential for future research combining elements of specialized component processing with F0-guided attention mechanisms to achieve even better results. Through this work, we have demonstrated that thoughtful architectural design focused on the unique characteristics of speech signals can yield substantial improvements in synthesis quality, even within practical computational constraints. These findings contribute to the broader understanding of how to effectively design neural vocoders that balance computational efficiency with high-quality audio generation [27].

Chapter 6

Future Work

This chapter outlines promising directions for future research that build upon the findings and methodologies presented in this thesis. While our exploration of the Spectral Processing Enhancement and F0-Guided Parallel Architecture has yielded significant improvements in AutoVocoder performance, several avenues for further advancement remain.

6.1 Integration of Approaches

The most promising direction for immediate future work is the integration of elements from both enhancement approaches. The complementary strengths of each approach; spectral accuracy from the Spectral Processing Enhancement and pitch precision from the F0-Guided Parallel Architecture; suggest that a combined approach could yield even better results, consistent with successful multi-component architectures in neural speech synthesis [49].

Such an integrated architecture might include:

- Separate phase and magnitude encoders from the first approach
- F0-guided masking and parallel processing from the second approach
- CBAM-enhanced processing throughout the network
- A unified latent space representation combining information from all processing paths

This integration would need to be done carefully to manage computational complexity while maintaining the key benefits of each approach.

6.2 Advanced Pitch Modeling

The success of the F0-guided approach suggests that further refinements in pitch modeling could yield additional benefits. Future work could explore:

- More sophisticated F0 estimation techniques that provide higher accuracy and robustness
- Explicit modeling of pitch contours using parametric representations or deep learning approaches

- Incorporation of prosodic features beyond F0, such as energy and duration patterns

By improving pitch modeling, future systems could achieve even greater naturalness and expressiveness in synthesized speech.

6.3 Efficiency Optimizations

While our enhanced approaches deliver significant quality improvements, they also increase computational complexity. Future work should address efficiency concerns through:

- Model compression techniques such as knowledge distillation [21] and pruning [17]
- Quantization for reduced memory footprint and faster inference [26]
- Hardware-specific optimizations for deployment on edge devices

These optimizations would make high-quality speech synthesis more accessible across a wider range of devices and applications, following established practices in neural network optimization [17].

6.4 Expanded Evaluation Framework

Future work should also include more comprehensive evaluation methodologies:

- Evaluation across more diverse speaking styles and emotional contexts
- Testing with multilingual content to assess cross-language generalization

A more extensive evaluation framework would provide deeper insights into the real-world performance of enhanced vocoder architectures.

Acknowledgements

First and foremost, I would like to thank myself, and my dear friend Firas who has been a major source of inspiration throughout my degree, in addition to my friends and family for their continuous encouragement and support throughout my academic journey.

I would also like to express my sincere gratitude to my supervisor for his invaluable guidance and support throughout this research. His expertise and insights have significantly contributed to the development of this work.

I extend my appreciation to the Department of Telecommunications and Artificial Intelligence at Budapest University of Technology and Economics for providing the computational resources necessary for training and evaluating the models presented in this thesis.

List of Figures

List of Tables

Bibliography

[1] Abien Fred Agarap. Deep learning using rectified linear units (relu), 2019.

[2] Jonathon B. Allen. Short term spectral analysis, synthesis, and modification by discrete fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 25 (3):235–238, 1977. DOI: 10.1109/TASSP.1977.1162950.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

[4] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders, 2021. URL https://arxiv.org/abs/2003.05991.

[5] Mikołaj Binkowski, Jeff Donahue, Sander Dieleman, Aidan Clark, Erich Elsen, Norman Casagrande, Luis C Cobo, and Karen Simonyan. High fidelity speech synthesis with adversarial networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=r1gfQgSFDr.

[6] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning, 2017. URL https://arxiv.org/abs/1611.05594.

[7] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1251–1258, 2017.

[8] VCTK Consortium. Voice conversion toolkit (vctk) corpus. http://datashare.is.ed.ac.uk/handle/10283/3054, 2018.

[9] Erica Cooper, Cheng-I Lai, Yusuke Yasuda, Fuming Fang, Xin Wang, Nancy Chen, and Junichi Yamagishi. Generalization in tts: How fundamental frequency affects cross-speaker performance. In *Interspeech 2020*, pages 3739–3743, 2020. DOI: 10.21437/Interspeech.2020-1096.

[10] Steven B Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. In *IEEE Transactions on Acoustics, Speech, and Signal Processing*, volume 28, pages 357–366, 1980. DOI: 10.1109/TASSP.1980.1163420.

[11] Alain De Cheveigné and Hideki Kawahara. Yin, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002. DOI: 10.1121/1.1458024.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL https://arxiv.org/abs/2010.11929.

[13] Hua-Ping Du, Yue-Xuan Lu, Yu Ai, and Zhen-Hua Ling. Bivocoder: A bidirectional neural vocoder integrating feature extraction and waveform generation, 2024.

[14] Yariv Ephraim and David Malah. Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(6):1109–1121, 1984. DOI: 10.1109/TASSP.1984.1164453.

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.

[16] Daniel Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2): 236–243, 1984. DOI: 10.1109/TASSP.1984.1164317.

[17] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *International Conference on Learning Representations*, 2016.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[19] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus), 2023.

[20] Gustav Eje Henter, Thomas Merritt, Matt Shannon, Catherine Mayo, and Simon King. Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends. *IEEE Signal Processing Magazine*, 35 (3):80–94, 2018. DOI: 10.1109/MSP.2017.2775361.

[21] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[22] Jie Hu, Li Shen, and Gang Sun. Attention mechanisms in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(12):2804–2821, 2020. DOI: 10.1109/TPAMI.2019.2956636.

[23] Yi Hu and Philipos C Loizou. Noizeus: A noisy speech corpus for evaluation of speech enhancement algorithms. volume 15, pages 319–329, 2007. DOI: 10.1109/TASL.2006.885122.

[24] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the 32nd International Conference on Machine Learning*, pages 448–456, 2015.

[25] Keith Ito and Linda Johnson. The LJ speech dataset. https://keithito.com/LJ-Speech-Dataset/, 2017.

[26] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the*

IEEE Conference on Computer Vision and Pattern Recognition, pages 2704–2713, 2018. DOI: 10.1109/CVPR.2018.00286.

[27] Felix Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Norm, Erich Lockhart, Jack Clark, Norman Casagrande, Jesse Engel, and Karen Simonyan. Efficient neural audio synthesis. *Proceedings of the 35th International Conference on Machine Learning*, pages 2410–2419, 2018.

[28] Takuhiro Kaneko, Kou Tanaka, Hirokazu Kameoka, and Shogo Seki. istftnet: Fast and lightweight mel-spectrogram vocoder incorporating inverse short-time fourier transform, 2022.

[29] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[30] Dennis H Klatt and Laura C Klatt. Analysis, synthesis, and perception of voice quality variations among female and male talkers. *The Journal of the Acoustical Society of America*, 87(2):820–857, 1990. DOI: 10.1121/1.398894.

[31] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis, 2020.

[32] J. Kuang and M. Liberman. Integrating voice quality cues in the pitch perception of speech and non-speech utterances. *Frontiers in Psychology*, 9:2147, 2018. DOI: 10.3389/fpsyg.2018.02147.

[33] E. Sudheer Kumar, K. Jai Surya, K. Yaswanth Varma, A. Akash, and K. Nithish Reddy. Noise reduction in audio file using spectral gatting and fft by python modules. In *Recent Developments in Electronics and Communication Systems*, pages 510–515. IOS Press, 2023. DOI: 10.3233/ATDE221305.

[34] Riad Larbi and Mohammed Salah Al-Radhi. Architectural enhancements and feature optimization of autovocoder for high-quality speech synthesis. In *Proceedings of the 3rd Workshop on Intelligent Infocommunication Networks, Systems and Services (WINS 2025)*, pages 27–32, Budapest, 2025. Budapest University of Technology and Economics. ISBN 978-963-421-982-8. DOI: 10.3311/WINS2025-005. URL https://doi.org/10.3311/WINS2025-005.

[35] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553): 436–444, 2015. DOI: 10.1038/nature14539.

[36] Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. Neural speech synthesis with transformer network. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6706–6713, 2019. DOI: 10.1609/aaai.v33i01.33016706.

[37] X Liu, Y Xu, W Zhang, and X Tian. Multiple prosodic meanings are conveyed through separate pitch ranges: Evidence from perception of focus and surprise in mandarin chinese. *Cognitive, Affective, Behavioral Neuroscience*, 21(6):1164–1175, Dec 2021. DOI: 10.3758/s13415-021-00930-9. Epub 2021 Jul 30.

[38] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022. DOI: 10.1109/CVPR52688.2022.01167.

[39] Chen-Chou Lo, Szu-Wei Fu, Wen-Chin Huang, Xin Wang, Junichi Yamagishi, Yu Tsao, and Hsin-Min Wang. Mosnet: Deep learning based objective assessment for voice conversion, 2021.

[40] Philipos C Loizou and Gibak Kim. Speech enhancement based on perceptually motivated bayesian estimators of the magnitude spectrum. *IEEE Transactions on Speech and Audio Processing*, 13(6):857–869, 2005. DOI: 10.1109/TSA.2005.851940.

[41] Erfan Loweimi, Zoran Cvetkovic, Peter Bell, and Steve Renals. Speech acoustic modelling from raw phase spectrum. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6738–6742, 2021. DOI: 10.1109/ICASSP39728.2021.9413727.

[42] Yunlong Ma, Zejun Yang, Liumeng Li, and Zhizheng Wang. Cross-lingual text-to-speech with global style tokens. In *Interspeech 2021*, pages 2608–2612, 2021. DOI: 10.21437/Interspeech.2021-1383.

[43] Matthias Mauch and Simon Dixon. pyin: A fundamental frequency estimator using probabilistic threshold distributions. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 659–663, 2014. DOI: 10.1109/ICASSP.2014.6853678.

[44] Masanori Morise, Fumiya Yokomori, and Kenji Ozawa. World: a vocoder-based high-quality speech synthesis system for real-time applications. *IEICE Transactions on Information and Systems*, 99(7):1877–1884, 2016. DOI: 10.1587/transinf.2015EDP7457.

[45] A. Michael Noll. Cepstrum pitch determination. *The Journal of the Acoustical Society of America*, 41(2):293–309, 1967. DOI: 10.1121/1.1910339.

[46] Alan V Oppenheim, Ronald W Schafer, and John R Buck. Discrete-time signal processing. *Prentice Hall*, 1989.

[47] ITU-T Recommendation P.862. Perceptual evaluation of speech quality (pesq): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs. *International Telecommunication Union*, 2001.

[48] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019.

[49] Ryan Prenger, Rafael Valle, and Bryan Catanzaro. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3617–3621, 2019. DOI: 10.1109/ICASSP.2019.8683143.

[50] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Prosody disentanglement for controllable speech synthesis. In *ICML 2021*, pages 8741–8751. PMLR, 2021.

[51] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. DOI: 10.1038/323533a0.

[52] Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4779–4783, 2018. DOI: 10.1109/ICASSP.2018.8461368.

[53] Stanley Smith Stevens. A scale for the measurement of the psychological magnitude loudness. *Journal of the Acoustical Society of America*, 8(3):185–190, 1937. DOI: 10.1121/1.1915893.

[54] David Talkin. A robust algorithm for pitch tracking (rapt). *Speech coding and synthesis*, 495:518, 1995.

[55] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. In *9th ISCA Speech Synthesis Workshop*, pages 125–125, 2016. DOI: 10.21437/SSW.2016-20.

[56] Patrick Van Hove, Monson Hayes, Jae Lim, and Alan Oppenheim. Signal reconstruction from signed fourier transform magnitude. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 31(5):1286–1293, 1983. DOI: 10.1109/TASSP.1983.1164178.

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, A. N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[58] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, et al. Tacotron: Towards end-to-end speech synthesis. *Proceedings of Interspeech*, pages 4006–4010, 2017. DOI: 10.21437/Interspeech.2017-1452.

[59] Oliver Watts, Lovisa Wihlborg, and Cassia Valentini-Botinhao. Puffin: Pitch-synchronous neural waveform generation for fullband speech on modest devices. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023. DOI: 10.1109/ICASSP49357.2023.10094729.

[60] Julian J. Webber, Cassia Valentini-Botinhao, Erica Williams, Gustav Eje Henter, and Simon King. Autovocoder: Fast waveform generation from a learned speech representation using differentiable digital signal processing, 2023.

[61] Norbert Wiener. Extrapolation, interpolation, and smoothing of stationary time series. *MIT Press*, 1949.

[62] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module, 2018. URL https://arxiv.org/abs/1807.06521.

[63] Sanghyun Woo, Sangdoo Yun, Jongchan Park, Jimei Yang, Alexander Kirillov, Joon-Young Lee, and Saining Xie. Convnext v2: Co-designing and scaling convnets with masked autoencoders, 2023.

[64] Heiga Zen, Keiichi Tokuda, and Alan W Black. Statistical parametric speech synthesis. *Speech Communication*, 51(11):1039–1064, 2009. DOI: 10.1016/j.specom.2009.04.004.