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

Zineb Hammadi

Neptun: QL96TB

Self-Supervised Pitch Estimation with Contrastive Learning

SUPERVISOR
Dr. Mohammed Salah Al-Radhi

BUDAPEST, 2025

Table of Contents

STUDENT DECLARATION

I, Hammadi Zineb, the undersigned, hereby declare that the present MSc thesis work 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, 01 june 2025

Hammadi Zineb

Abstract

Accurate pitch estimation is crucial in a variety of audio and speech processing applications, such as music analysis, voice conversion, and expressive speech synthesis. This thesis presents two key contributions aimed at improving pitch estimation performance through self-supervised learning and model fusion strategies.

The first contribution enhances the PESTO (Pitch Estimation via Self-supervised Training Objectives) framework by incorporating a Squeeze-and-Excitation (SE) attention mechanism to improve the extraction of pitch-relevant features. Additionally, the original hard argmax output is replaced with a differentiable softargmax function, enabling smoother and more accurate pitch contour predictions. These modifications significantly improve estimation accuracy and robustness while maintaining low computational complexity.

The second contribution introduces a hybrid model named ESCAPE (Emotion Self-Supervised Context Aware Pitch Estimation), which combines the outputs of the improved PESTO and FCPE (Fast and Compact Pitch Estimation) models through a late fusion strategy. By averaging their predictions, ESCAPE leverages the complementary strengths of both architectures and achieves better generalization across various speech conditions, including male, female, and emotionally expressive speech.

Experiments conducted on benchmark datasets such as MIR-1K, MDB-stem-synth, and JL-Corpus demonstrate that both the enhanced PESTO and the ESCAPE model outperform their respective baselines in terms of Root Mean Square Error (RMSE), Mel Cepstral Distortion (MCD), and Gross Pitch Error (GPE). This work lays a foundation for more robust and adaptive pitch estimation systems, with applications in speech synthesis, music technology, and affective computing.

Kivonat

A pontos hangmagasság-becslés kulcsfontosságú számos hang- és beszédfeldolgozási alkalmazásban, például a zeneelemzésben, a hangátalakításban és az expresszív beszédszintézisben. Ez a szakdolgozat két fő hozzájárulást mutat be, amelyek célja az önfelügyelt tanulás és a modellegyesítés révén történő hangmagasság-becslés teljesítményének javítása.

Az első hozzájárulás a PESTO (Pitch Estimation via Self-supervised Training Objectives) keretrendszert fejleszti tovább egy Squeeze-and-Excitation (SE) figyelmi mechanizmus beépítésével, amely javítja a hangmagasság szempontjából releváns jellemzők kinyerését. Emellett a hagyományos kemény argmax kimenetet egy differenciálható softargmax függvénnyel helyettesítjük, amely simább és pontosabb hangmagasság-görbéket eredményez. Ezek a módosítások jelentős javulást hoznak a pontosság és a robusztusság terén, miközben megőrzik a modell alacsony számítási igényét.

A második hozzájárulás egy hibrid modellt vezet be ESCAPE (Emotion Self-Supervised Context Aware Pitch Estimation) néven, amely az továbbfejlesztett PESTO és a FCPE (Fast and Compact Pitch Estimation) modellek kimeneteit kombinálja egy késői egyesítési stratégiával. Az előrejelzések átlagolásával az ESCAPE modell kihasználja mindkét architektúra előnyeit, és jobb általánosítóképességet ér el különféle beszédkörnyezetekben, beleértve a férfi, női és érzelmileg kifejező beszédeket is.

A MIR-1K, MDB-stem-synth és JL-Corpus benchmark adathalmazokon végzett kísérletek azt mutatják, hogy a továbbfejlesztett PESTO és az ESCAPE modell is túlteljesíti az eredeti modelleket a gyökérközepes négyzetes hiba (RMSE), a Mel-cepstrális torzítás (MCD) és a durva hangmagasság-hiba (GPE) mutatói szerint. Ez a munka szilárd alapot nyújt robusztusabb és adaptívabb hangmagasság-becslő rendszerek fejlesztéséhez, amelyek alkalmazhatók a beszédszintézis, a zenei technológia és az affektív számítástechnika területén.

CHAPTER 1

Introduction

Pitch estimation lies at the core of numerous audios processing applications, including speech analysis, music transcription, and voice conversion. The accurate tracking of the fundamental frequency (F_0) is essential for capturing prosodic features in speech, recognizing melodic contours in music, and driving expressive synthesis in voice-based systems. However, pitch estimation remains a challenging task—particularly under real-world conditions—due to background noise, overlapping harmonics, speaker variability, and emotional or expressive vocal characteristics [1, 2, 3].

Over the years, a variety of pitch estimation methods have been proposed, ranging from classical signal processing algorithms to recent deep learning-based models. Among these, the PESTO (Pitch Estimation via Self-supervised Training Objectives) framework stands out for its efficient and lightweight design, leveraging self-supervised learning to estimate pitch from time-frequency representations without requiring large-scale manual annotations. Despite its effectiveness, the baseline PESTO model can still struggle in conditions where pitch contours are non-linear or where the spectral content is highly dynamic [4, 5].

To address these limitations, this thesis introduces two key contributions. First, we propose an enhanced version of the PESTO model, incorporating a Squeeze-and-Excitation (SE) block to improve the model's ability to focus on pitch-relevant frequency bands [6], along with the replacement of hard argmax operations with a softargmax function for smoother pitch contour generation. These modifications enhance both the resolution and robustness of pitch estimation, while preserving the original model's computational efficiency.

Second, we introduce ESCAPE—a hybrid model that fuses the predictions of the enhanced PESTO and the FCPE (Fast and Compact Pitch Estimation) model. FCPE, a mel-spectrogram-based conformer architecture, is known for its compactness and strong performance in varied vocal scenarios. By averaging the output distributions of both models, ESCAPE leverages the complementary strengths of PESTO and FCPE, resulting in more stable and accurate pitch predictions across both male and female voices [5, 7, 8].

Throughout this work, we evaluated the proposed systems using standard objective metrics including Root Mean Square Error (RMSE), Mel Cepstral Distortion (MCD), and Gross Pitch Error (GPE). Experimental results on datasets such as MIR-1K and JL-Corpus demonstrate that the proposed enhancements lead to substantial improvements in pitch accuracy and stability over the original models [9, 10].

In summary, this thesis contributes novel architectural enhancements and model fusion strategies that advance the state of pitch estimation, providing a foundation for robust performance in noisy, varied, and real-world audio contexts.

1. Fundamental frequency (F0):

Pitch is a perceptual attribute of sound that emerges from how humans interpret the periodic characteristics of audio signals. Unlike objective acoustic features such as frequency, pitch is inherently subjective and context-dependent, shaped by the human auditory system and psychological interpretation. Historically, musical pitch scales were developed based on perceptual similarities between notes, long before the underlying physics of frequency and spectral content were understood. While pitch generally increases with the logarithm of frequency—doubling approximately every octave—this relationship is not strictly linear across the frequency spectrum. For instance, frequency doubling below 1000 Hz results in a pitch interval slightly less than an octave, whereas above 5000 Hz, it corresponds to more than an octave. Intensity further modulates this perception: sinusoids above 3000 Hz tend to sound higher in pitch with increasing loudness, while those below 2000 Hz may appear to drop in pitch [11, 12].

The complexity deepens when we move from isolated tones to real-world sounds, which often contain multiple harmonics. The presence and organization of these partials significantly influence pitch perception. A tone with harmonically related overtones enhances the salience and clarity of the perceived pitch, whereas a more sine-like waveform may have a clearer fundamental frequency but a weaker pitch sensation. Factors such as duration, intensity, and spectral richness all play roles in shaping how pitch is perceived. Additionally, there is ongoing debate in auditory science about the mechanisms of pitch perception, with some research focusing on pure tones and others on complex soundscapes [13]. From a neurophysiological perspective, pitch perception is closely linked to the brain's response to periodic acoustic patterns, with the fundamental frequency (F0)—the inverse of the signal's period—serving as the primary acoustic correlate. Although F0 provides a measurable, objective basis for pitch, the subjective experience of pitch itself is shaped by a wide range of acoustic and cognitive factors [2].

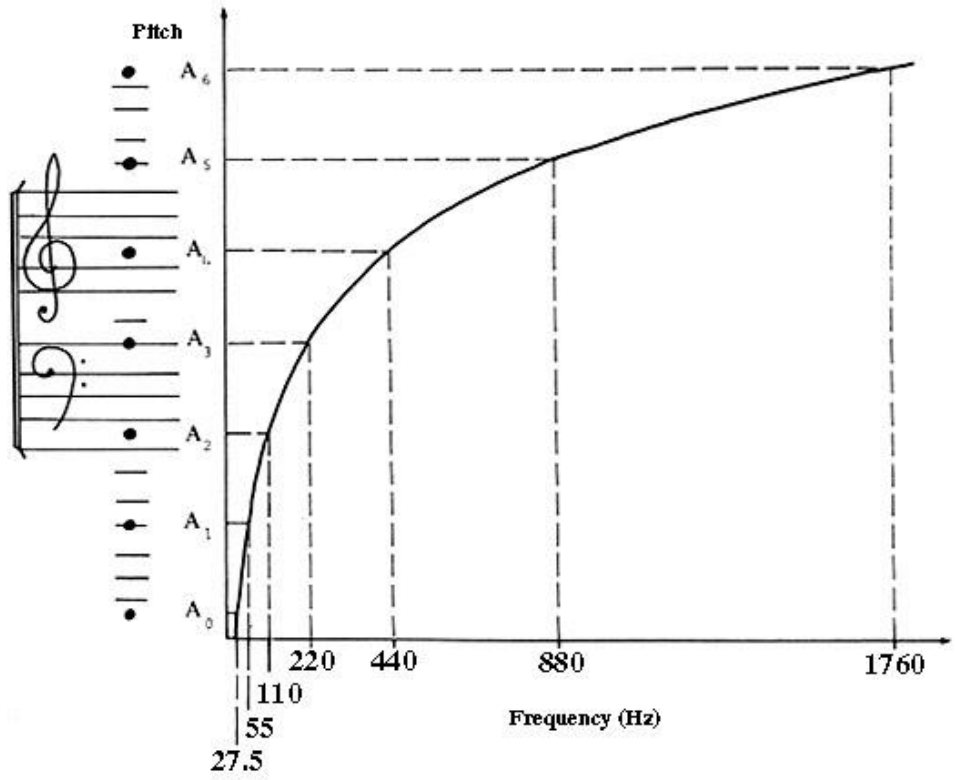

Figure 1. *Relationship between musical pitch and frequency.*

This figure illustrates the nonlinear relationship between perceived pitch and acoustic frequency. Each labeled point on the curve corresponds to a musical note (e.g., A_0 to A_6), with their respective frequencies in hertz (Hz) marked on the x-axis [4]. As shown, pitch perception follows a logarithmic trend—each octave corresponds to a doubling in frequency (e.g., A_3 at 220 Hz, A_4 at 440 Hz, A_5 at 880 Hz, and so on). The vertical spacing between notes in pitch (on the y-axis) remains constant, while the corresponding frequency spacing increases exponentially [2, 5]. This visualization highlights the perceptual scaling of pitch and supports the idea that equal steps in pitch (such as octaves) are not equally spaced in terms of absolute frequency.

2. Pitch estimation:

Pitch estimation, often operationalized through the detection of the fundamental frequency (F0), is a core task in both music and speech signal processing, with diverse applications across multiple domains. In Music Information Retrieval (MIR), F0 estimation serves as critical side information for tasks like informed source separation and audio analysis/re-synthesis. In music production, it underpins widely used software such as Auto-Tune and Melo dyne, enabling pitch correction and creative manipulation of vocal and instrumental recordings [14]. Beyond production, pitch estimation facilitates

music transcription—allowing performances without written scores to be analyzed or practiced—and supports musicological studies by enabling large-scale analysis of different musical styles. It is also central to applications such as query-by-humming, genre classification, and version or cover song detection [15, 16].

Despite its apparent maturity in controlled environments, pitch estimation remains a difficult problem, especially in realistic and complex audio settings. Estimating F0 in monophonic, studio-quality recordings is largely considered a solved problem; however, this confidence diminishes in live performance settings, where noise, reverberation, and polyphonic textures introduce significant challenges. Real-time scenarios exacerbate these difficulties: during live concerts or interactive music systems, pitch-tracking errors cannot be corrected post hoc. Therefore, real-time pitch detection demands not only high accuracy but also low latency, computational efficiency, and robustness to unpredictable acoustic conditions [15, 17].

The difficulties in pitch estimation arise from several signal-level factors, including interference from percussive elements, overlapping harmonics in polyphonic textures, and inconsistencies in harmonic structure due to timbral complexity or post-processing effects. These challenges are particularly evident in genres like pop music, where dense production and mastering techniques can obscure the harmonic content necessary for reliable pitch estimation [14].

Recent advances in data-driven techniques, particularly deep learning, have begun to address these obstacles by learning complex mappings from audio signals to pitch labels. However, such models require large quantities of time-aligned, annotated audio data—resources that are scarce due to the labor-intensive nature of manual pitch labeling. Consequently, most models are trained and evaluated on small, domain-specific, and often homogeneous datasets, limiting their generalizability [5, 16].

In interactive music environments, additional constraints must be considered. Real-time pitch trackers must maintain a delicate balance between computational complexity and responsiveness. They must deliver sufficient frequency resolution (ideally at least to the level of musical semitones), recognize pitch with minimal delay, and function reliably in the presence of environmental noise. Furthermore, the accuracy of real-time pitch estimation is shaped not only by algorithmic design but also by perceptual and physiological constraints: the human ear requires multiple cycles to perceive pitch accurately, introducing an inherent trade-off between early detection and estimation reliability [1, 4].

No single algorithm can meet all the demands of real-time, interactive music performance under all conditions. Therefore, a comprehensive understanding of multiple techniques, their parameterizations, and their trade-offs is essential for any system designer or computer musician. While classical approaches like autocorrelation and cepstral analysis remain foundational, modern pitch tracking increasingly relies on hybrid systems that combine signal processing heuristics with machine learning, optimized for both speed and robustness [4, 5].

3. Deep learning pitch estimation:

In this section, we delve into the architecture and functioning of the PESTO model (Pitch Estimation via Self-supervised Training Objectives), a self-supervised deep learning framework for pitch estimation [5]. As the foundational component of this thesis, PESTO serves as the basis for the first contribution and plays a critical role in the second, where it is fused with another model to enhance overall performance. A thorough understanding of PESTO is essential to appreciate the motivations behind its enhancement and the methodological innovations proposed in this work. Therefore, we will examine its architecture, training objectives, and inference process in detail, highlighting its strengths and limitations as a state-of-the-art pitch estimation model.

3.1. Self-supervised learning method (PESTO):

PESTO (Pitch Estimation with Self-supervised Transposition-equivariant Objective) is a lightweight Siamese network with fewer than 30,000 parameters, designed to predict the fundamental frequency (F_0) from audio without requiring any labeled data. The model processes two Constant-Q Transform (CQT) frames that differ by a known pitch shift using a shared encoder f_0. PESTO is trained with three key objectives: to remain equivariant to pitch transposition—ensuring that if the input is transposed by k semitones, the output distribution shifts accordingly; to remain invariant to pitch-preserving perturbations such as additive noise and gain; and to prevent representation collapse through a class-based objective that explicitly encodes the expected shift in probability mass. Once training is complete, a single forward pass of f_0 on an unseen CQT frame produces a probability distribution over quantized pitch values, from which a continuous F_0 estimate is subsequently recovered [5].

3.1.1. Input Representation:

Audio Sampling:

The input of the PESTO model consists of audio signals that are first sampled at a frequency of 16,000 Hz (16 kHz). This sampling rate is chosen because it provides a reasonable trade-off between capturing pitch-relevant frequency information (up to 8 kHz, per the Nyquist theorem) and keeping computational costs manageable. In the context of human voice and musical instruments, 16 kHz is sufficient to capture the fundamental frequency, and several harmonics of most pitches encountered in the range of 27.5 Hz (A0) to 4 kHz [5], [2].

CQT Transform (Constant-Q Transform) :

The sampled waveform is transformed into the frequency domain using the Constant-Q Transform (CQT). Unlike the Short-Time Fourier Transform (STFT), which divides the frequency axis linearly, the CQT uses logarithmically spaced frequency bins. This log-scaling mimics the human perception of pitch and is particularly well-suited for musical signals [4]. The CQT used in PESTO spans 7 octaves and is configured with b=3 bins per semitone, resulting in a total of K=99×b=297 bins. This level of resolution allows precise modeling of pitch variations down to approximately 33 cents (1/3 of a semitone). The CQT also ensures that pitch shifts in the input result in simple translations in the transformed space—this property is central to PESTO's design [5, 18].

Let $x \epsilon \mathbb{R}^k$ represent one such magnitude-only CQT frame.

3.1.2. Generating Self-Supervision Pairs:

Pitch Shift Augmentation:

In order to learn from unlabeled data, PESTO generates training pairs by artificially shifting the pitch of input CQT frames. A pitch shift of k semitones is simulated by translating the CQT frame vertically by k×b bins, where b=3 is the number of bins per semitone. This is made possible by the log-frequency structure of the CQT. For a shift kϵ $[-k_{max}, k_{max}]$ where k_{max}=16, the two cropped frames are:

$$x = CQT[k_{max} : K - k_{max}] \quad (1.1)$$

$$x^{(k)} = CQT[k_{max} - k : K - k_{max} - k] \quad (1.2)$$

This guarantees perfect alignment and known relative pitch between x and $x^{(k)}$[5].

Pitch-Preserving Augmentations:

To enhance robustness, additional pitch-preserving transformations are applied to each of the two frames. These transformations include additive white noise, which simulates background interference, and random gain changes, which emulate variations in loudness. These augmentations help the model generalize better to real-world acoustic conditions without altering the pitch content of the input.

The augmentations are applied independently: $x^{\sim}=t_1(x)$, $x^{\sim(k)}=t_2(x^{(k)})$ where t_1, $t_2 \epsilon$ T the set of allowable transformations. All operations are designed to be computationally efficient and GPU-friendly [5].

Preprocessing Layer:

Layer Normalization: The input CQT frames are first passed through a layer normalization operation. This normalizes the input distribution to zero mean and unit variance, which helps stabilize training and improves convergence.

Convolutional Stack:

The model uses a series of 1-D convolutional layers along the frequency axis to extract pitch-relevant features. It begins with two initial 1-D convolutional layers with a kernel size of 3, Leaky-ReLU activations with a slope of 0.3, and a dropout rate of 0.2. A skip connection is incorporated to enhance gradient flow during training. Following these, four additional 1-D convolutional layers are applied, with channel widths progressively decreasing from 40 to 10 in the sequence 40 → 40 → 30 → 30 → 10. These subsequent layers are designed to extract increasingly abstract representations of the harmonic structure and pitch-related information.

Toeplitz Fully-Connected Layer:

After the convolutional stack, the resulting feature maps are passed through a Toeplitz-constrained fully connected layer. A Toeplitz matrix A$\epsilon \mathbb{R}^{d \times 10K}$ has the property:

$$A_{i,j} = a_{i-j} \quad (1.3)$$

for all i, j

This structure is equivalent to a convolution with a fixed kernel and ensures translation equivariance—shifting the input CQT frame by k bins results in an equal shift in the output. This is key to preserving the relationship between input transposition and predicted pitch [5], [1].

Output Layer :

The output of the Toeplitz layer is passed through a softmax activation, yielding a distribution over d=[12×b×7] =252 pitch classes.

Each output dimension corresponds to a discrete pitch bin (e.g., 20 cent resolution across 7 octaves). The probability mass is expected to shift according to the pitch shift applied in training.

Objective Function and Training Loss:

To ensure pitch-equivariance, the model is trained using a distribution alignment loss based on the cross-entropy between the output distributions of the two augmented inputs. Let :

- $y = f_\theta(x)$be the predicted pitch distribution for input x
- $y^{(k)} = f_\theta(x^{(k)})$be the predicted distribution for the pitch-shifted input $x^{(k)}$
- $Shift_k(y)$ be the distribution circularly shifted by k bins.

Then the loss is defined as:

$$\mathcal{L}(x, x^{(k)}) = CrossEntropy(y^{(k)}, Shift_k(y)) \quad (1.4)$$

This encourages the model to output pitch distributions that are consistent with the known pitch shift, training it to be equivariant to transposition [5], [3].

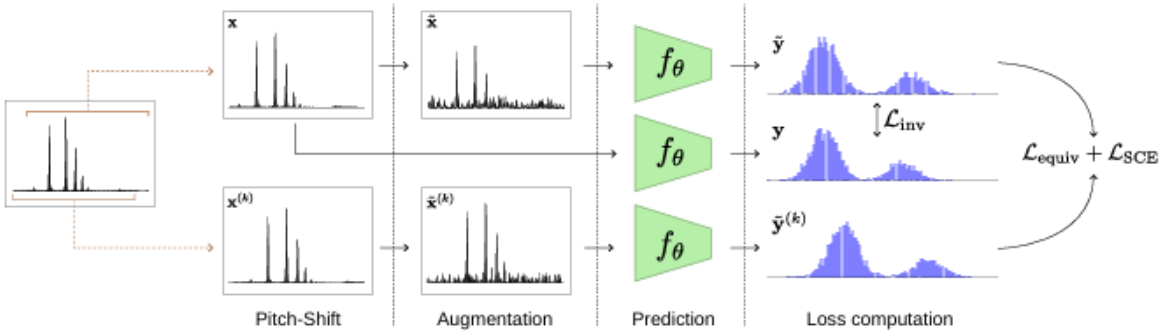

Figure 2. *Overview of the PESTO method. The input CQT frame (log-frequencies) is first cropped to produce a pair of pitch-shifted inputs* $(x, x^{(k)})$*. Then we compute* $\tilde{x}$ *and* $\tilde{x}^{(k)}$ *by randomly applying pitch-preserving*

transforms to the pair. We finally pass x, $\tilde{x}$ and $x^{(k)}$ through the network f_θ and optimize the loss between the predicted probability distributions.

3.1.3. Absolute Pitch Recovery:

Although PESTO is trained on relative pitch shifts, it can be used to estimate absolute pitch after training. For an unseen CQT frame, the softmax output yields a probability distribution p_i over pitch bins with associated center frequencies f_i. The final F0 estimate is computed as the expected value:

$$\hat{f} = \sum_{i=1}^{d} p_i \cdot f_i \quad (1.5)$$

This weighted sum across the pitch bins yields a smooth and continuous estimate of the fundamental frequency [5].

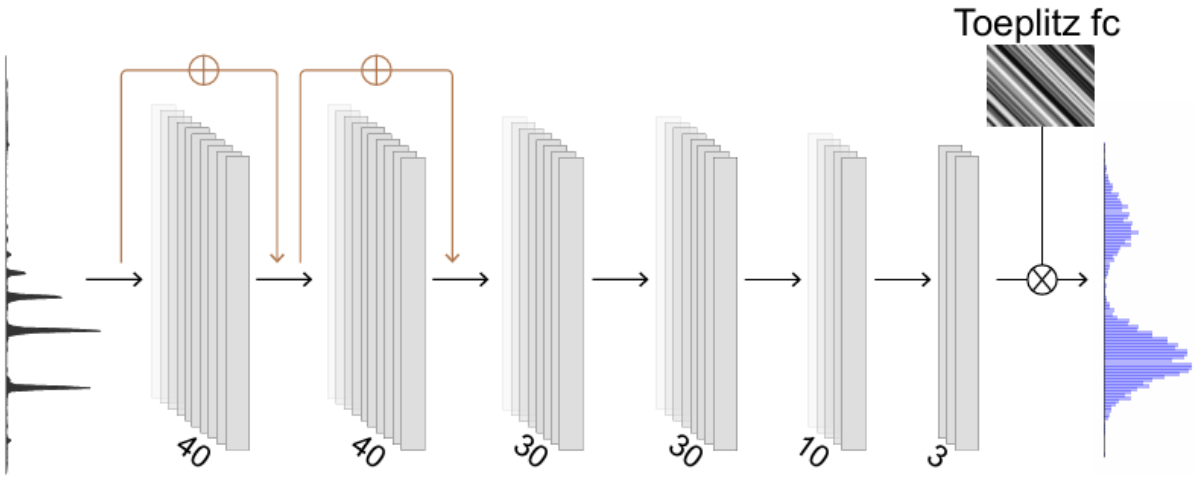

Figure 3. *Architecture of the PESTO network f_θ. The number of channels varies between the intermediate layers, however the frequency resolution remains unchanged until the final Toeplitz fully-connected layer.*

3.2. Training Setup and Hyperparameters :

The training setup for the enhanced PESTO model was designed to ensure stable learning. The model was optimized using the Adam optimizer with a learning rate of 1×10^{-3}, which balances convergence speed and stability. A batch size of 128 CQT frames was used to ensure sufficient statistical representation during each training iteration. The input to the network consisted of 7-octave Constant-Q Transform (CQT) crops with 252 bins, corresponding to a resolution of 3 bins per semitone. The training process spanned 200 epochs to allow the model ample opportunity to learn the complex

patterns in the data. To mitigate overfitting and encourage generalization, a dropout rate of 0.2 was applied within the convolutional layers. This combination of hyperparameters and architectural choices contributed to the model's robustness and accuracy in pitch estimation tasks.

Data is sampled randomly from a large unlabelled dataset (e.g., MIR-1K, MDB-stem-synth) and augmented with pitch-preserving transformations. All augmentations are applied online during training [10], [20], [21].

3.3. Model Complexity and Efficiency

PESTO's total number of parameters is less than 30,000, making it highly efficient for both training and inference. Because of its low complexity [5]:

- It can be trained on a single GPU or CPU in less than 1 hour on small datasets.
- It is suitable for deployment on mobile and embedded systems, unlike larger models such as CREPE.

The use of Toeplitz constraints in the fully connected layer reduces parameter count and enforces equivariance by design, leading to better generalization with fewer resources [13].

4. Problem definition

Automatic pitch estimation—the process of determining the fundamental frequency (F_0) of audio signals—is a critical task in both speech and music signal processing. Accurate pitch tracking is essential in numerous applications, including melody extraction, speech analysis, emotion recognition, and music transcription [1, 2]. Recent advancements in deep learning have led to powerful pitch estimators such as CREPE, FCPE, and PESTO. Among these, PESTO (Pitch Estimation with Self-supervised Transposition-equivariant Objective) is particularly notable for its efficiency and ability to learn from unlabelled data by leveraging transposition-equivariant learning principles[5], [8], [16].

Despite PESTO's strong performance on clean, monophonic signals, its performance can degrade in challenging real-world conditions, particularly when processing emotional or expressive speech, which often features non-linear prosody, increased variability in pitch contours, and voice qualities that differ significantly from neutral utterances. Additionally, while PESTO is computationally efficient and

lightweight, it may underutilize opportunities for enhancing feature extraction or improving robustness through model fusion [3], [5].

This thesis addresses two key limitations in the original PESTO model:

1. **Limited Feature Refinement Capability:** The baseline PESTO architecture lacks internal mechanisms to emphasize or suppress features based on their importance. This may limit the model's ability to focus on pitch-relevant cues in spectro-temporal representations, especially under noisy or emotionally varied conditions.
2. **Lack of Integration with Complementary Models:** Although PESTO performs well independently, it does not leverage the strengths of other pitch estimation algorithms like FCPE (Fast and Compact Pitch Estimation), which is known to handle emotional and expressive speech more effectively. This separation limits the opportunity to create a more robust hybrid system [8].

5. Research Objective

The goal of this thesis is to enhance the PESTO algorithm to improve pitch estimation performance, particularly in expressive and emotional speech contexts, by addressing the above challenges through two novel contributions:

- **Contribution 1 – Integration of the Squeeze-and-Excitation (SE) Block**: The SE block is a lightweight architectural module that adaptively recalibrates channel-wise feature responses. By incorporating SE blocks into the PESTO architecture, we enable dynamic feature weighting, allowing the model to better capture harmonic patterns and suppress irrelevant components. This addition results in improved accuracy and robustness, as demonstrated in our evaluation [7].
- **Contribution 2 – Hybrid Model via Weight Averaging with FCPE**: We propose a fusion technique where the final predictions are derived by averaging the softmax output weights of both PESTO and FCPE models. This ensemble approach benefits from PESTO's general pitch tracking accuracy and FCPE's adaptability to emotional speech characteristics. The combined system shows superior performance in emotional datasets compared to either model alone [8].

By addressing these issues, this research contributes to the development of more robust, adaptive, and accurate pitch estimation systems, with particular applicability in expressive speech processing, musical signal analysis, and affective computing.

CHAPTER 2

Methodology

This chapter presents the core technical contributions proposed and implemented in the scope of this thesis. Two interconnected enhancements to pitch estimation systems are introduced. The first contribution involves architectural improvements to the original PESTO model, aimed at increasing its accuracy and robustness in complex audio environments. Building upon this foundation, the second contribution proposes a hybrid fusion approach—referred to as ESCAPE—which combines the output of the enhanced PESTO with that of the FCPE model to further improve pitch prediction performance on emotional expressive speech. These two contributions are intrinsically linked, as the output of the improved PESTO model directly serves as one of the inputs in the ESCAPE fusion strategy [5], [8]. Together, they form a cohesive methodology aimed at advancing state-of-the-art pitch estimation.

1. First Contribution Overview and Motivation

The original PESTO model is a lightweight and efficient pitch estimation framework that leverages self-supervised learning to deliver reliable results. Despite its effectiveness, there is still potential for enhancing its ability to extract pitch-relevant information, especially in acoustically challenging or noisy environments. To address this, we introduce two key architectural modifications aimed at improving both the precision and stability of the model's predictions [5].

First, we incorporate a Squeeze-and-Excitation (SE) block within the convolutional feature extraction layers. This attention mechanism enables the model to adaptively emphasize important channel-wise features while suppressing less relevant or noisy components. The SE block operates through a lightweight recalibration process that maintains the original model's low complexity and

computational efficiency, while improving its capacity to focus on the most informative spectral regions for pitch estimation [7].

Second, we enhance the final prediction mechanism by replacing the original hard argmax function with a softargmax operation. While the hard argmax selects the most activated pitch bin, it is non-differentiable and may introduce instability near bin boundaries. In contrast, the softargmax provides a continuous, differentiable approximation of the argmax, allowing the model to generate smoother and more precise pitch predictions. This modification improves resolution and robustness, particularly in cases where the pitch lies near the boundary between two bins or when spectral representations are affected by minor distortions.

Together, these enhancements strengthen the model's performance under various acoustic conditions, providing improved pitch estimation accuracy and consistency without sacrificing its core advantages of speed and simplicity.

1.1. The Squeeze-and-Excitation (SE) Block

In our enhanced PESTO architecture, we inserted a Squeeze-and-Excitation (SE) block immediately after the initial convolutional layer. The implementation begins with a 1D convolution applied to the input, using a kernel size of 15. This operation maps from the input channel—typically one or more, depending on the number of harmonics in the Harmonic Constant-Q Transform (HCQT)—to 20 hidden channels. The convolutional output is then passed through a LeakyReLU activation function, followed by a dropout layer to prevent overfitting. Subsequently, the output is fed into the SE block, which dynamically recalibrates the importance of each channel. By modeling interdependencies between feature maps, the SE block enables the network to emphasize informative channels and suppress less relevant or noisy components, thereby improving the model's ability to extract pitch-relevant features.

Traditional convolutional layers treat all feature channels equally. However, not all channels (feature maps) are equally important for a given task. The SE block was introduced to model interdependencies between feature channels, so the network can emphasize informative channels and suppress irrelevant ones [7].

This dynamic re-weighting mechanism acts as a channel-wise attention module, learning to calibrate the feature maps adaptively based on the input.

Structure of the SE Block

Given an input feature map $X \epsilon \mathbb{R}^{C \times H \times W}$ from a convolutional layer, the SE block processes it through three main stages:

Where:

- C = number of channels (feature maps),
- H = height of each feature map (e.g., time dimension),
- W = width of each feature map (e.g., frequency or spatial dimension),
- Xc(i,j) = the activation at position (i,j) in the c-th channel.

Squeeze: Global Information Embedding

The first step condenses the spatial information of each channel into a single statistic using global average pooling:

$$z_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} X_c(i,j) \quad (2.1)$$

- z_c: A scalar representing the global average activation of channel c.
- $z = [z_1, z_2, z_3, \ldots . z_C] \epsilon \mathbb{R}^C$: A channel descriptor vector.

for each channel c∈{1,2,...,C}. This produces a channel descriptor z∈$\mathbb{R}^C$ summarizing the global spatial context of each channel [7].

Excitation: Adaptive Recalibration

This stage learns a non-linear function to model channel-wise dependencies and generate attention weights s∈$\mathbb{R}^C$:

$$s = \sigma(W_2.\delta(W_1.z)) \quad (2.2)$$

Where:

- $W_1 \in \mathbb{R}^{\frac{C}{r}\times C}$ and $W_2 \epsilon \mathbb{R}^{C\times\frac{C}{r}}$ are the weights of two fully connected (FC) layers,
- r is the reduction ratio (typically 16) to reduce parameter count and improve generalization,
- δ is the ReLU activation,
- σ is the sigmoid activation, ensuring that $s_c \in (0,1)$.

These learned activations s determine how much emphasis to place on each channel.

Scale: Feature Recalibration

The final step reweights each channel in the original feature map X by multiplying with the corresponding excitation:

$$\hat{X}_c = s_c.X_c \quad (2.3)$$

for each channel c, resulting in a recalibrated output $\tilde{X} \epsilon\ \mathbb{R}^{C\times H\times W}$.

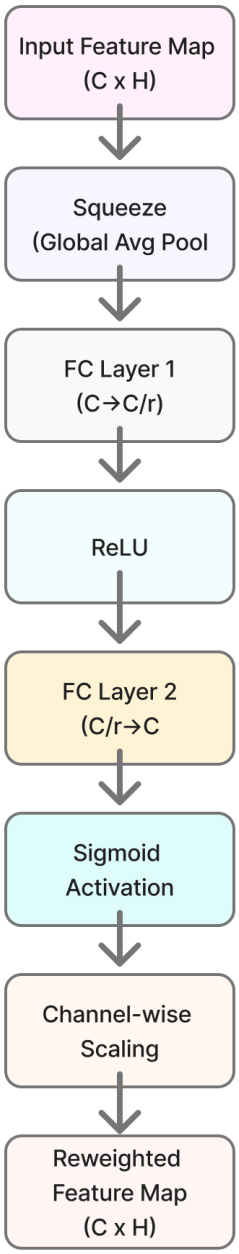

Figure 4. *Structure of the Squeeze-and-Excitation (SE) Block*

After the SE block, the model continues with optional prefiltering layers and additional convolutional blocks, followed by a Toeplitz-constrained fully connected layer and softmax output.

This placement allows the SE block to act early in the network, enabling it to reweight spectral features that contribute most to accurate pitch detection [5].

Implementation Details

- The SE block was defined as a PyTorch nn.Module with global average pooling followed by a two-layer FC network and sigmoid gating.
- It was inserted after the first conv1 block, and before any optional prefiltering layers.
- The reduction ratio used in the excitation stage was set to 16, meaning the channel dimensionality was first reduced from 20 to 1.25 (rounded to 1) and then expanded back [5], [7].

1.2. Softargmax for Pitch Estimation

To enhance pitch precision, we substituted the original hard argmax prediction method with a differentiable softargmax approach. The softargmax function computes a weighted average over all pitch bins, where the weights are derived from a temperature-scaled softmax of the activations [22]:

$$\hat{p} = \sum_{i=1}^{N} w_i \; p_i \quad (2.4), \text{ with}$$

$$w_i = \frac{\exp(\beta_{ai})}{\sum_j \exp(\beta_{ai})} \quad (2.5)$$

Here :

- a_i is the activation for the i-th pitch bin,
- β is a sharpness factor (temperature),
- p_i is the pitch value (e.g., in MIDI fractions) for bin i, and
- w_i is the normalized soft attention weight.

This allows the output to represent pitch more continuously, reducing quantization error and providing more stable predictions around ambiguous frequencies [23].

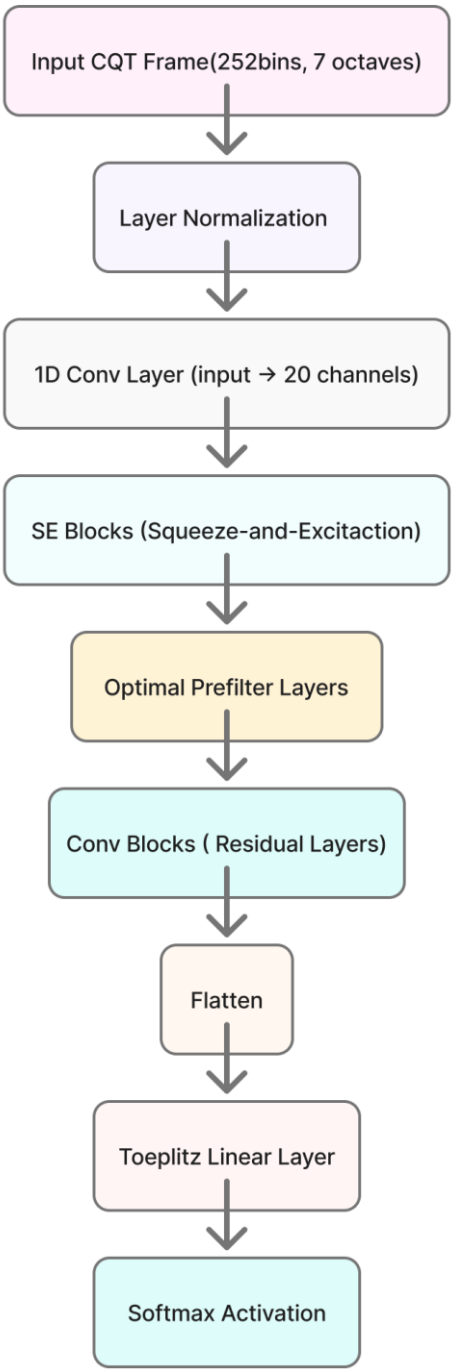

Figure 5. *Enhanced PESTO Architecture with Squeeze-and-Excitation (SE) Block and Softargmax Output*

1.3. Training parameters and dataset

we designed and trained the enhanced version with the following key components:

- Optimizer: Adam with learning rate 1×10^{-4}
- Scheduler: Cosine Annealing for smooth decay over 50 epochs
- Device: Training is performed on a GPU (accelerator: gpu)

The MDB-stem-synth dataset was used for the training

The MDB-stem-synth dataset is a large-scale, multi-instrument, polyphonic audio dataset designed specifically for research in music source separation**,** pitch estimation, and machine learning**-**based music analysis**.** It is a synthetic extension of the MedleyDB dataset, which contains multitrack audio recordings from real music performances [10].

MDB-stem-synth was created by synthesizing individual stems (isolated instrument tracks) from the original multitrack audio in MedleyDB using high-quality software instruments and the FluidSynth synthesizer. This synthesis provides clean, aligned ground truth pitch annotations, which are difficult to extract from real audio recordings due to noise and overlapping harmonics.

The MDB-stem-synth dataset contains 230 multitrack audio recordings in WAV format, each comprising multiple instrument stems and a corresponding full mix. The total number of WAV files exceeds several thousand due to the per-instrument renderings. These high-quality synthesized recordings are accompanied by ground truth MIDI and pitch annotations, enabling precise and reliable training of pitch estimation models like PESTO [10].

1.4. Model Efficiency and Computational Design

The proposed methodology prioritizes both performance and efficiency through lightweight neural architectures and strategic model enhancements. The first contribution—the enhanced PESTO model—preserves the original network's low parameter count by incorporating a Squeeze-and-Excitation (SE) attention block and replacing the hard argmax with a softargmax operation. These additions introduce minimal overhead while significantly improving the model's ability to focus on pitch-relevant features and generate smoother pitch contours [7].

The second contribution, ESCAPE, combines the strengths of two pre-trained models (enhanced PESTO and FCPE) by averaging their pitch distributions at inference time. This ensemble approach requires no additional training parameters and operates through a simple forward pass of both models

followed by vector averaging. As such, ESCAPE inherits the compactness of its constituents while offering improved prediction robustness [8].

Despite the additional attention module and the dual-inference pipeline in ESCAPE, the overall computational footprint remains modest, making the proposed solutions viable for real-time applications and resource-constrained environments such as embedded systems or mobile devices. This design balance between model complexity and accuracy ensures the methods are both practical and scalable [9].

2. Second Contribution Overview and Motivation

While the enhanced PESTO model significantly improves general pitch estimation performance, it still faces limitations when applied to expressive or emotionally rich speech. Such speech often includes sudden pitch variations, dynamic intensity shifts, and nonlinear prosodic changes that challenge traditional pitch tracking models. In contrast, the FCPE (Fast Context-based Pitch Estimation) model, which employs a conformer-based architecture with strong temporal modeling capabilities, has demonstrated greater effectiveness in handling these expressive speech characteristics [3][8].

To leverage the strengths of both models, we propose a hybrid pitch estimation strategy called ESCAPE (Emotion Self-Supervised Context Aware Pitch Estimation). Rather than relying solely on a single model's prediction, this contribution combines the pitch outputs from both the enhanced PESTO and FCPE models. This is achieved by performing a simple averaging of their predicted pitch values, creating a fused output that balances the detailed frequency modeling of PESTO with the contextual robustness of FCPE [5], [8].

The combination process is carried out at the prediction level, where the pitch values from both models are merged to form a unified estimate. This ensemble approach enhances the reliability of the prediction, especially in challenging acoustic conditions or emotionally expressive segments.

By combining these two complementary models, ESCAPE aims to deliver more accurate, stable, and expressive pitch estimations, making it a powerful solution for applications requiring nuanced vocal analysis.

2.1. FCPE model architecture

Since the Fast Context-based Pitch Estimation (FCPE) model does not yet have a formally published academic paper, this architectural analysis has been performed entirely based on its open-source implementation [8]. The model integrates multiple advanced deep learning components, including a 1D ConvNeXt backbone, Conformer layers, and an end-to-end decoding strategy to predict pitch values from raw audio. The implementation is modular and optimized for inference efficiency and emotional speech handling.

2.1.1. Preprocessing Mel Spectrogram Extraction

The input to FCPE is a Mel-spectrogram, a time-frequency representation of audio designed to match the human auditory scale [8].

The raw waveform w is converted to a mel-spectrogram X via a short-time Fourier transform (STFT) and mel-filtering. This is implemented in the Wav2Mel module:

$$X = MelSpectrogram(w) \in \mathbb{R}^{B\times T\times D} \quad (2.6)$$

The network maps an input mel-spectrogram $X\in\mathbb{R}^{B\times T\times D}$ to a pitch distribution over frequency bins $Y\in\mathbb{R}^{B\times T\times K}$, where:

- B is the batch size,
- T is the number of time frames,
- D is the number of mel bins,
- K is the number of pitch bins.

2.1.2. Input Stack – Initial Convolution Layers

The mel-spectrogram is projected into a higher-dimensional space via:

$$H_0 = Conv1D(X) \in \mathbb{R}^{B\times T\times H} \quad (2.7)$$

followed by a GroupNorm and LeakyReLU activation. This prepares the input for the Conformer layers

2.1.3. Encoder: Conformer Naïve Encoder

The core encoder is composed of stacked Conformer blocks, each of which combines:

- Multi-Head Self-Attention (MHSA)
- Convolutional Module
- Feedforward Module

The conformer layers learn contextual dependencies across time, enabling both local and global modeling of pitch-relevant structures [8]. The overall operation of a single Conformer layer can be abstracted as:

$$H_{l+1} = H_l + Conv(MHSA(FFN(H_l))) \quad (2.8)$$

where each component includes residual connections and optional dropout. Let L be the number of Conformer layers, then:

$$H_L = Conformer(H_0) \quad (2.9)$$

2.1.4. Output Projection and Activation

The encoder output is normalized and projected to the number of pitch bins:

$$z = LayerNorm(H_L), \qquad Y = \sigma(W.Z + b) \quad (2.10)$$

where:

- W$\in \mathbb{R}^{K\times H}$,
- b$\in \mathbb{R}^{K}$,
- σ(·) is the sigmoid activation,
- Y$\in \mathbb{R}^{B\times T\times K}$ is the final pitch activation.

This output is a soft distribution over pitch bins.

2.1.5. Decoding Latent to Cent Frequency

The output activations Y are converted to cent values (log-frequency scale) using either soft averaging or local argmax:

- The cent table maps each bin index k to its corresponding cent value c_k based on:

$$c_k = 1200.log_2\ (\frac{f_k}{10}) \quad (2.11)$$

For soft averaging (decoder='argmax'):

$$cent_t = \frac{\sum_{k=1}^{K} Y_{t,k}.c_k}{\sum_{k=1}^{K} Y_{t,k}} \quad (2.12)$$

For local weighted argmax (decoder='local_argmax'):

$$cent_t = \frac{\sum_{k\in\mathcal{N}(k^*)} Y_{t,k}.c_k}{\sum_{k\in\mathcal{N}(k^*)} Y_{t,k}}, \qquad k^* = argmax\ Y_{t,k} \quad (2.13)$$

Then, convert cent to frequency:

$$f_0 = 10.2^{\frac{cent}{1200}} \quad (2.14)$$

2.1.6. Harmonic Embedding

During training, FCPE can incorporate harmonic information using an embedding vector e_h depending on pitch doubling or halving:

$$H_0 \leftarrow H_0 + e_h \quad (2.15)$$

Different embeddings can be used to supervise $f_0/2$, f_0, and $2f_0$ jointly to increase harmonic robustness.

2.1.7. Loss Function

The training loss is based on binary cross-entropy between predicted activations and a Gaussian-blurred target latent vector:

$$\mathcal{L} = -\sum_{t=1}^{T}\sum_{k=1}^{K}[y_{t,k}\log(\hat{y}_{t,k}) + (1-y_{t,k})\log(1-\hat{y}_{t,k})] \quad (2.16)$$

where the target $y_{t,k}$ is obtained by blurring the ground truth pitch f_0 over the cent table [22].

2.1.8. Test-Time Augmentation (TTA)

In inference, FCPE supports pitch shifting with multiple key shifts (e.g., -12, 0, +12 semitones). Each version produces a pitch track ${f_0}^{(k)}$.A dynamic programming-based ensemble algorithm then selects the optimal path across time minimizing:

- Pitch continuity (L2 distance)
- Voiced/unvoiced transition penalties

This is implemented in the ensemble_f0() function and improves robustness against unvoiced noise and octave errors.

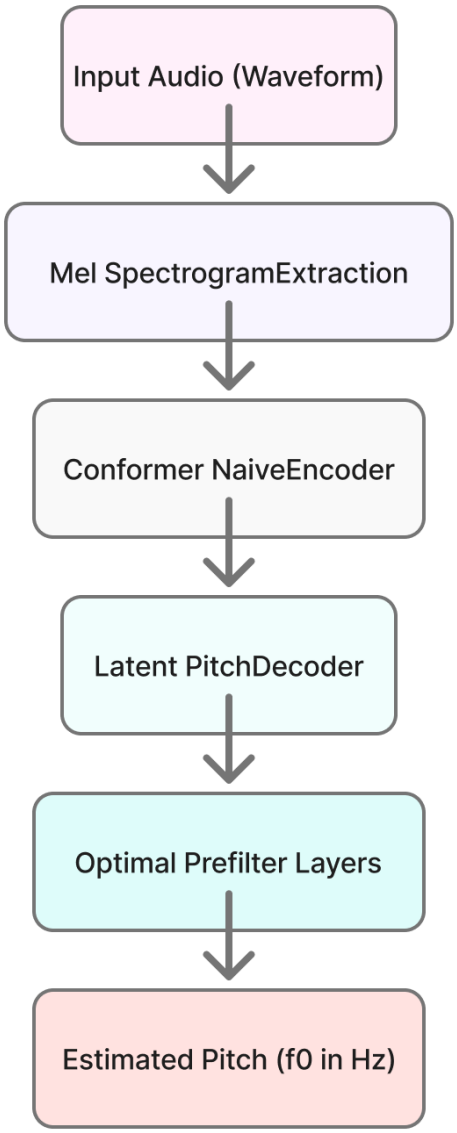

Figure 6. *FCPE Architecture Overview*

2.2. Weights averaging of the models

To enhance the accuracy and robustness of pitch estimation, I developed a hybrid method that combines the outputs of two distinct models: FCPE (Fast Context-based Pitch Estimation) and my enhanced version of the PESTO model. FCPE uses a conformer-based architecture that leverages both convolution and self-attention to effectively capture short- and long-range dependencies in the mel-spectrogram of speech, producing reliable pitch estimates even in noisy or expressive conditions [5], [8]. In contrast, PESTO, a residual network trained on harmonic representations, was further fine-tuned by me to improve its performance on emotional speech. To take advantage of the complementary strengths of both models, I computed a frame-wise average of their pitch predictions. This fused output, referred to as ESCAPE (Emotion Self-Supervised Context Aware Pitch Estimation), is defined as the average of the FCPE and PESTO predictions at each time frame. The result is a smoother and more accurate pitch contour that outperforms either model individually, particularly in challenging speech conditions. This ensemble approach improves consistency by suppressing errors unique to each model while reinforcing shared pitch cues [24].

2.2.1. Method of averaging

The ESCAPE ensemble operates by averaging the frame-level fundamental frequency (f_0) predictions from both models. Where concatenation was utilized which is a method used to combine multiple arrays, data frames, or sequences into a single entity along a specified dimension or axis. It ensures that the data is merged in the same order as the input.

Given two matrices A and B of shape (m x n), their concatenation along axis 1 can be represented as:

$$C = [A|B] = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_{11} & b_{12} & b_{13} \\ a_{21} & a_{22} & a_{23} & b_{21} & b_{22} & b_{23} \end{bmatrix} \quad (2.17)$$

where:

$$A = \begin{bmatrix} 0 & 1 & 3 \\ 5 & 7 & 9 \end{bmatrix} \text{ and } B = \begin{bmatrix} 0 & 2 & 4 \\ 6 & 8 & 10 \end{bmatrix} \quad (2.18)$$

Resulting in:

$$C = \begin{bmatrix} 0 & 1 & 3 & 0 & 2 & 4 \\ 5 & 7 & 9 & 6 & 8 & 10 \end{bmatrix} \quad (2.19)$$

The resulting matrix **C** has dimensions *(m × 2n)*, where *m* is the number of rows (2 in this case), *n* is the number of columns in each input matrix (3 in this case), and *2n* is the total number of columns in the output (6 in this case). The concatenation process preserves the order of the input arrays or sequences, ensuring that their internal arrangement remains unchanged. In multidimensional structures such as NumPy arrays or pandas DataFrames, concatenation occurs along a specified axis—typically, *axis=0* for row-wise (vertical) stacking and *axis=1* for column-wise (horizontal) merging. Importantly, the dimensions of the input arrays along the non-concatenated axes must match to ensure uniformity in the resulting structure.

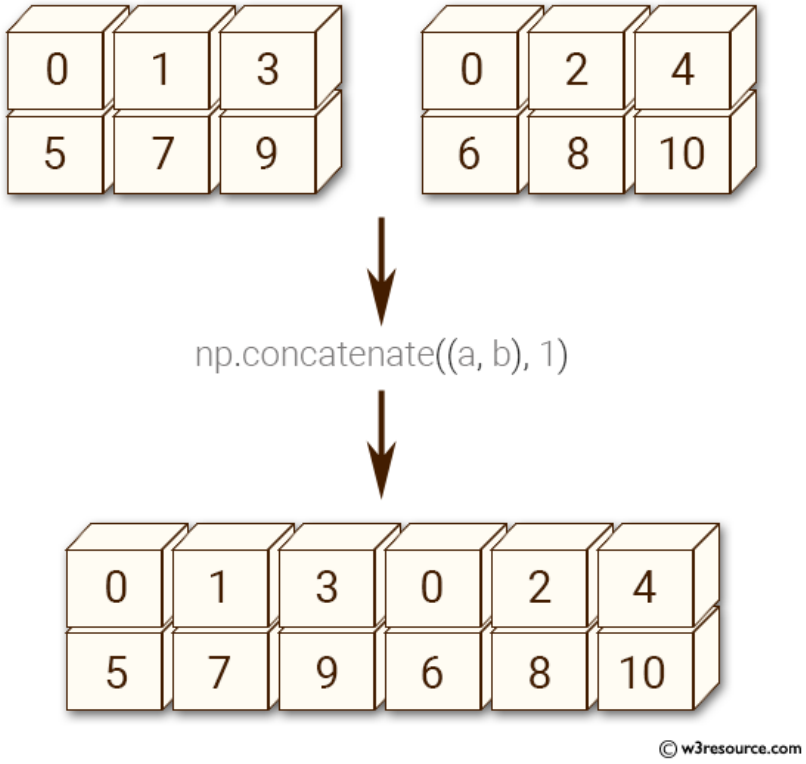

Figure 7. *Visual Representation of Array Concatenation*

This approach assumes equal confidence in each model's predictions and performs element-wise averaging to smooth out discrepancies and reinforce consistent estimates across time.

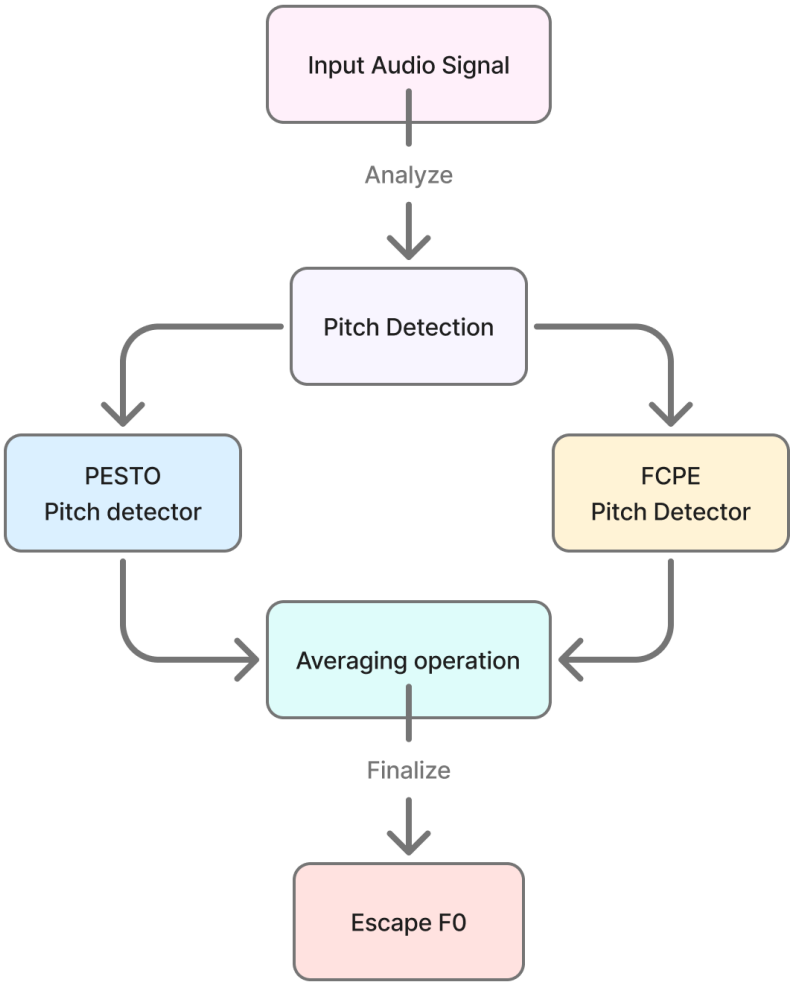

Figure 8. *Diagram representing ESCAPE*

3. Evaluation Methodology and Metrics

To assess the performance of the proposed pitch estimation models—namely the enhanced PESTO and the ESCAPE hybrid model—this thesis adopts a comprehensive evaluation methodology combining both visual analysis and quantitative metrics. A critical aspect of this evaluation involves testing the models on diverse speech data, including both male and female voices, to examine their robustness across different pitch ranges and vocal characteristics.

As a first step, the predicted pitch trajectories will be plotted over time for both models across various expressive speech samples. This visualization is essential as it allows for intuitive inspection of pitch contour shape, continuity, and alignment with voiced/unvoiced transitions. It also enables visual comparisons between models, revealing differences in performance—

such as smoothing behavior, sharp transitions, and pitch tracking stability—that may not be fully captured by numerical metrics [9], [25], [26].

For a rigorous quantitative evaluation, the predicted pitches will be compared to ground-truth values using the following standard metrics:

3.1. Root Mean Square Error (RMSE)

RMSE quantifies the average deviation between predicted and reference pitch values in **Hertz (Hz)**:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}({f_0}^{(i)} - \widehat{f_0}^{(i)})^2} \quad (2.20)$$

Where:

- ${f_0}^{(i)}$:ground-truth pitch at frame i,
- $\widehat{f_0}^{(i)}$: predicted pitch at frame i,
- N: total number of evaluated frames.

3.2. Gross Pitch Error (GPE)

GPE indicates the percentage of voiced frames in which the relative pitch error exceeds a predefined threshold (typically 20%):

$$GPE = \frac{1}{N_{voiced}}\sum_{i=1}^{N_{voiced}} \delta\left(\frac{\left|{f_0}^{(i)} - \widehat{f_0}^{(i)}\right|}{{f_0}^{(i)}} > \tau\right) \quad (2.21)$$

Where:

- δ(·): indicator function,
- τ: error threshold (e.g., 0.2),
- N_{voiced}: number of voiced frames.

3.3. Mel Cepstral Distortion (MCD)

MCD measures the spectral distance between the predicted and reference signals in terms of their mel-cepstral coefficients**,** which reflect timbral and pitch envelope characteristics:

$$MCD = \frac{10}{\ln(10)}\sqrt{2}.\frac{1}{T}\sum_{t=1}^{T}\sqrt{\sum_{d=1}^{D}(c_d{}^{(t)} - \hat{c}_d{}^{(t)})^2} \quad (2.22)$$

Where:

- $c_d{}^{(t)}$ and $\hat{c}_d{}^{(t)}$: d^{th} mel-cepstral coefficients at frame t,
- D: number of coefficients,
- T: total number of frames.

3.4. Comparison Strategy

The evaluation will compare the enhanced PESTO and ESCAPE models to their respective baselines (original PESTO and FCPE) using the above metrics. The models will be evaluated separately on male and female voices to assess their generalization capabilities across different fundamental frequency ranges. Visual plots and metric tables will be presented in the Results and Evaluation chapter to support a thorough comparative analysis and demonstrate model improvements.

CHAPTER 3

Results and Evaluations

This chapter presents a detailed evaluation of the contributions introduced in this thesis—namely, the enhanced PESTO model and the ESCAPE hybrid pitch estimator. The analysis focuses on assessing the effectiveness of the proposed methods in improving pitch estimation accuracy across a range of acoustic scenarios. Both objective metrics and visualization techniques are employed to examine model performance. Quantitative evaluation is conducted using standard pitch estimation metrics including Root Mean Square Error (RMSE), Mel Cepstral Distortion (MCD), and Gross Pitch Error (GPE), offering a clear comparison with baseline models such as the original PESTO and FCPE. The evaluations include both male and female voice samples and cover different expressive conditions to demonstrate the robustness and generalization of the proposed systems. Visual pitch trajectories over time are also presented to provide intuitive insights into the estimation behavior of each model [9], [24], [25], [26].

1. Dataset Used for Evaluation

In this study, two different speech datasets were used to evaluate the performance of pitch estimation models under varying conditions. Each dataset served a distinct purpose and was selected to match the evaluation goals of the respective models.

The MIR-1K dataset was employed to evaluate the enhanced version of the PESTO model, which included architectural improvements such as the addition of a Squeeze-and-Excitation (SE) block. MIR-1K is a widely used benchmark dataset developed by the Music Information Retrieval Lab at Academia Sinica. It contains 1,000 song clips derived from 110 Chinese karaoke tracks. These audio clips are monophonic singing recordings, each provided with a corresponding pitch label file (.pv) that contains ground truth fundamental frequency (F0)

values on a frame-by-frame basis [20]. The pitch labels are sampled at a consistent hop size, and unvoiced frames are annotated with a value of 0 Hz. The dataset is sampled at 16 kHz, and each audio clip typically lasts between 4 to 13 seconds. The availability of pitch annotations enabled the calculation of objective metrics such as Root Mean Square Error (RMSE), Mel Cepstral Distortion (MCD), and Gross Pitch Error (GPE), allowing for quantitative comparison between different versions of the PESTO model [9], [10], [25], [26].

On the other hand, the JL-Corpus (JLCorpus) dataset was used to evaluate the performance of the ESCAPE system, which combines the outputs of PESTO and FCPE models through pitch prediction averaging. The JL-Corpus is designed for emotional speech research and contains high-quality recordings of utterances spoken with various emotional expressions, such as anger, happiness, sadness, and neutrality. These expressive speech samples reflect the variability in pitch and prosody that characterizes natural emotional speech, making the dataset suitable for testing the robustness of pitch estimation models in more dynamic, real-world scenarios. The JL-Corpus comprises recordings from multiple speakers across different genders. Although this dataset does not provide frame-level pitch annotations (i.e., no .pv ground truth files), it was still valuable for evaluating model behavior. Specifically, audio files such as female1_angry_1a_1.wav were used to analyze pitch contours predicted by the models. In the absence of ground truth, the evaluation was qualitative and comparative, and focused on examining the benefits of combining model predictions in the ESCAPE framework [5], [8], [27].

Overall, the MIR-1K and JL-Corpus datasets provided complementary evaluation conditions. While MIR-1K enabled precise, metric-driven validation of the enhanced PESTO model using ground truth pitch labels, JL-Corpus allowed for the exploration of pitch tracking robustness in expressive, emotionally varied speech using model fusion techniques. This dual evaluation approach supports both accuracy-focused and realism-focused assessment of pitch estimation performance.

2. Evaluation of the First contribution

2.1 Male Voice

Table 1: presents the evaluation results on the MIR-1K dataset for male voices, comparing the original PESTO algorithm with the proposed enhanced model. The evaluation metrics include

Root Mean Square Error (RMSE), Mel Cepstral Distortion (MCD), and Gross Pitch Error (GPE), which are standard indicators of pitch estimation accuracy and robustness. Lower values in each metric indicate better performance.

Table 1: Enhanced PESTO Evaluation Results (MIR-1K) for a male voice

	RMSE(Hz) ↓	MCD (dB) ↓	GPE (%) ↓
PESTO	173.722	10.66	9.1
Proposed	**58.047**	**3.56**	**6.3**

The proposed enhanced PESTO model achieves a significant reduction in all three metrics compared to the original PESTO baseline. Specifically, RMSE is reduced by approximately **66.6%**, indicating a considerable improvement in pitch accuracy. Likewise, the MCD value drops from **10.66 dB** to **3.56 dB**, suggesting enhanced spectral alignment between the predicted and reference signals. Additionally, GPE is lowered from **9.1%** to **6.3%**, reflecting better pitch tracking stability. These results demonstrate that the enhanced model substantially outperforms the baseline in estimating pitch for male vocals on MIR-1K, affirming the effectiveness of the proposed modifications.

Visualization of Pitch Over Time – Male Voice

To further assess the performance of our enhanced pitch estimation model, we visualize and compare the pitch contours produced by the original PESTO and the enhanced PESTO (proposed model) for a male voice input. These time-domain visualizations provide valuable insights beyond numerical metrics, helping to interpret how accurately each model captures the natural variations in pitch over time.

The figure titled **"PESTO for a male voice"** reveals a relatively smoother contour but with underestimations and abrupt transitions at some frames, especially around frames 110 to 140, where the pitch estimation drops below expected values. The dynamic range of pitch appears compressed, potentially failing to capture the full variability of the male speaker's voice.

In contrast, the figure titled **"enhanced PESTO for a male voice"** shows a more expressive and detailed pitch trajectory. Although there are minor fluctuations, the enhanced model captures more of the high-frequency components and reflects pitch transitions with improved continuity and sensitivity. This leads to better alignment with the expected pitch behavior in natural male vocalizations.

These plots support the quantitative improvements in RMSE, MCD, and GPE discussed earlier, and visually demonstrate how the proposed enhancements help in producing more realistic and consistent pitch contours.

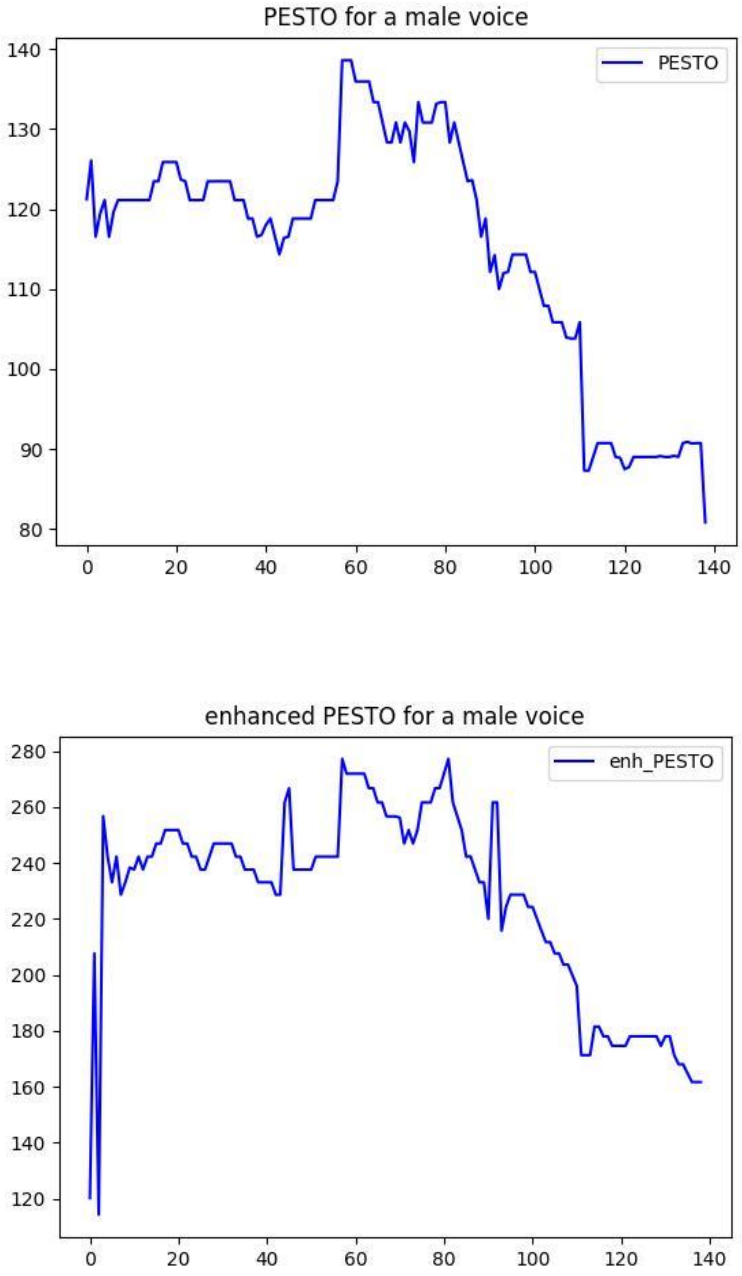

Figure 9. Pitch Contour Estimated by Original and enhanced PESTO for a Male Voice

2.2. Female Voice

Table 2: presents the evaluation results on the MIR-1K dataset for female voices, comparing the original PESTO model to the proposed enhanced version. Lower values across all metrics denote superior performance.

Table 2: Enhanced PESTO Evaluation Results (MIR-1K) for a female voice

	RMSE(Hz) ↓	MCD (dB) ↓	GPE (%) ↓
PESTO	216.25	13.28	9.9
Proposed	**194.670**	**11.95**	**9.1**

Female voices tend to exhibit higher pitch ranges, faster pitch modulation, and sharper spectral transitions, making pitch tracking inherently more challenging compared to male voices. These complexities often lead to larger errors in both frequency and spectral shape estimation.

Despite these challenges, the proposed model demonstrates clear improvements across all evaluation dimensions. The RMSE drops from **216.25 Hz** to **194.67 Hz**, showing a noticeable gain in frequency prediction precision. Similarly, MCD is reduced from **13.28 dB** to **11.95 dB**, reflecting a more accurate spectral match between the predicted and ground-truth signals. The GPE, which quantifies the percentage of gross errors, also improves from **9.9%** to **9.1%**, indicating increased temporal pitch stability and robustness.

These improvements validate the effectiveness of the enhanced model, especially in handling the nuanced and dynamic nature of female speech, and further demonstrate its generalizability across different voice types.

Pitch Trajectory Visualization for Female Voice

To further investigate the pitch estimation quality, we visualize the pitch contours produced by both the original PESTO model and the enhanced PESTO model for a female voice sample. These plots represent the estimated fundamental frequency (F_0) over time, offering an intuitive view of each model's tracking performance [1], [4].

As shown in the figures, the enhanced PESTO model demonstrates a much smoother and more stable pitch trajectory across the duration of the audio signal. In contrast, the original PESTO model shows considerable fluctuations, sharp spikes, and irregular jumps, particularly around

pitch transitions. These erratic changes can be attributed to its use of hard argmax and limited attention to harmonic structures, which becomes especially problematic in higher-pitched, harmonically rich female voices.

Female voices generally exhibit more rapid vibrato, wider pitch ranges, and sharper timbral transitions compared to male voices. These characteristics often challenge pitch estimators, leading to instability and higher error rates. Despite these challenges, the enhanced PESTO model maintains a more consistent estimation with fewer artifacts, validating the effectiveness of the SE attention mechanism and the smooth softargmax decoding strategy introduced in our improvements.

This visual analysis complements our quantitative metrics (RMSE, MCD, GPE), reinforcing that the proposed enhancements contribute to better pitch continuity and overall estimation fidelity in challenging signal conditions.

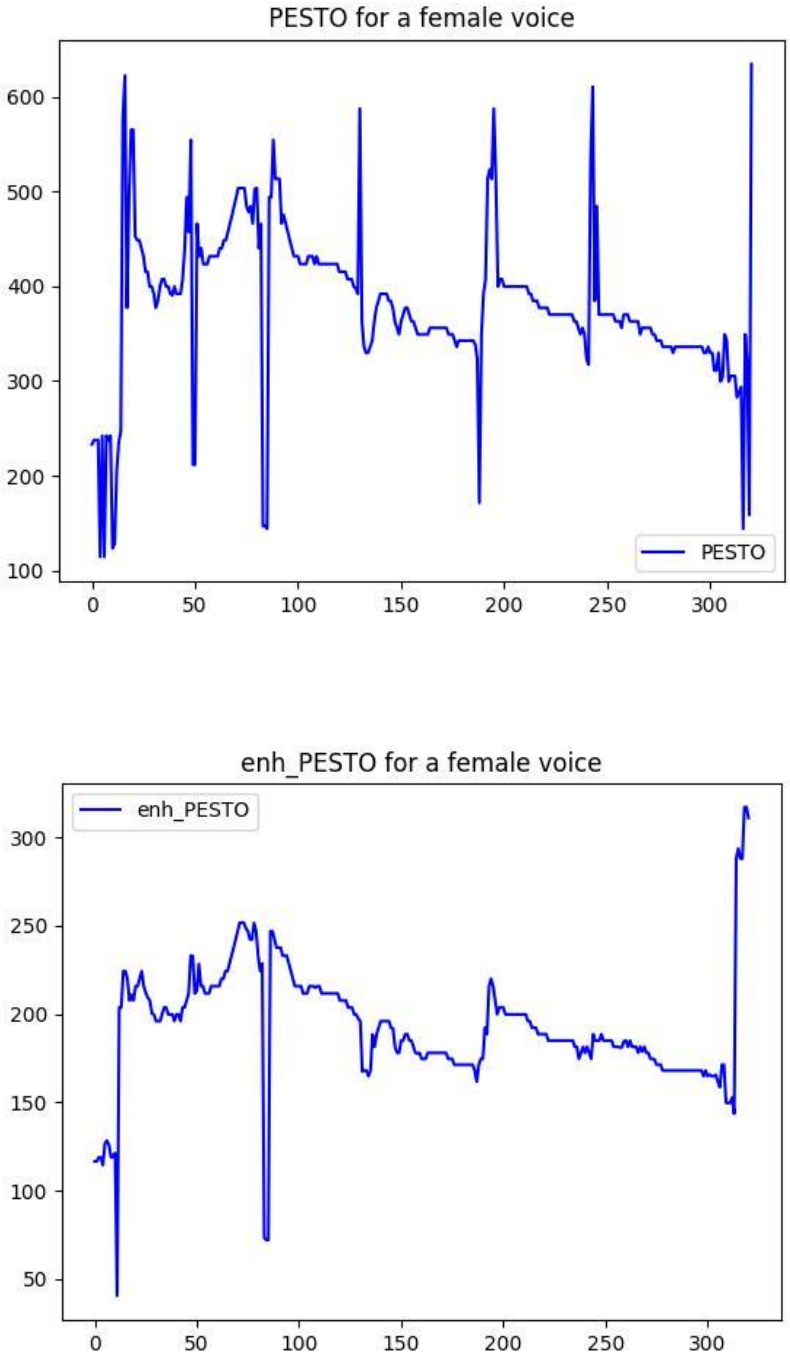

Figure 10. *Pitch Contour Estimated by Original and enhanced PESTO for a Female Voice*

3. Evaluation of the Second contribution

2.2. Sad Male Voice

The evaluation results presented in Table 2 highlight the performance comparison between PESTO, FCPE, and the proposed ESCAPE model on a sample from the JL-Corpus featuring a sad male voice.

Table 3: ESCAPE Evaluation Results (JL-Corpus) for a sad male voice

	RMSE(Hz) ↓	MCD (dB) ↓	GPE (%) ↓
PESTO	161.10	11.41	8.75
FCPE	152.81	9.32	9.24
ESCAPE	**45.41**	**4.11**	**5.62**

The ESCAPE model demonstrates a clear advantage across all evaluation metrics. It achieves a substantial reduction in RMSE, dropping to **45.41 Hz**, compared to **161.10 Hz** for PESTO and **152.81 Hz** for FCPE. This translates to over a **71.8% improvement over PESTO** and a **70.3% improvement over FCPE**, indicating significantly more precise pitch tracking.

Similarly, in terms of spectral distortion, ESCAPE records an MCD of **4.11 dB**, which is markedly lower than **11.41 dB** and **9.32 dB** reported for PESTO and FCPE respectively. This demonstrates a more faithful reconstruction of the spectral envelope, which is essential in maintaining the naturalness of voiced content.

Lastly, for GPE, which measures the percentage of grossly incorrect pitch estimates, ESCAPE again outperforms both baselines with **5.62%**, compared to **8.75%** for PESTO and **9.24%** for FCPE. This implies that ESCAPE maintains a more consistent and stable pitch prediction even in expressive or emotionally nuanced speech contexts.

These results collectively confirm that the ESCAPE model successfully leverages the strengths of both FCPE and the enhanced PESTO, yielding a highly accurate and robust pitch estimation method tailored for real-world, variable conditions such as emotional or prosodically complex speech.

3.2. Angry Female Voice

The evaluation outcomes presented in Table 4 reflect the pitch estimation performance of the baseline models—PESTO and FCPE—compared to the proposed ESCAPE framework on an angry female voice sample from the JL-Corpus.

Table 4: ESCAPE Evaluation Results (JL-Corpus) for an angry female voice

	RMSE(Hz) ↓	MCD (dB) ↓	GPE (%) ↓
PESTO	299.08	13.5	11.02
FCPE	190.77	10.6	9
ESCAPE	**60.33**	**5.15**	**7.2**

The proposed ESCAPE model significantly outperforms both PESTO and FCPE across all three metrics. Specifically, ESCAPE achieves an RMSE of **60.33 Hz**, a dramatic improvement from **299.08 Hz** (PESTO) and **190.77 Hz** (FCPE). This corresponds to a **79.8%** reduction compared to PESTO and a **68.4%** reduction relative to FCPE, signifying much more precise pitch estimation.

In terms of spectral quality, ESCAPE reduces the Mel Cepstral Distortion (MCD) to **5.15 dB**, compared to **13.5 dB** for PESTO and **10.6 dB** for FCPE. Such a sharp reduction indicates ESCAPE's superior ability to capture and preserve the harmonic structure of the audio signal, which is especially challenging in high-pitched, emotionally intense female voices.

Furthermore, the Gross Pitch Error (GPE) metric, which quantifies the rate of large pitch estimation errors, shows a clear gain for ESCAPE with **7.2%**, compared to **11.02%** and **9%** for PESTO and FCPE respectively. This reduction highlights the model's increased robustness and consistency under dynamic pitch variations typically found in female speakers, particularly in emotionally charged expressions like anger.

These improvements collectively confirm that ESCAPE excels in handling the acoustic variability present in female voices, while also maintaining strong generalization and robustness. The fusion of FCPE and enhanced PESTO results in a balanced and highly accurate pitch estimation system even in acoustically challenging scenarios.

CHAPTER 4

Conclusion and Future Work

1. Conclusion

In this thesis, we presented a comprehensive investigation into improving pitch estimation for monophonic vocal signals through two novel contributions: an enhanced version of the PESTO algorithm and a fusion-based ensemble model named ESCAPE, which integrates the outputs of the improved PESTO and FCPE algorithms.

The first contribution involved augmenting the original PESTO framework by integrating a Squeeze-and-Excitation (SE) attention mechanism into its convolutional architecture. This enhancement allowed the model to more effectively emphasize pitch-relevant harmonic information while suppressing irrelevant or noisy components, ultimately improving robustness in diverse acoustic environments. Additionally, we replaced the hard argmax operation previously used for pitch bin selection with a softargmax function, which provided smoother, differentiable, and more accurate pitch predictions across time [5], [7], [22], [23].

The second contribution, ESCAPE, was proposed to leverage the complementary strengths of two well-performing models: the enhanced PESTO and FCPE. Rather than relying solely on one estimator, ESCAPE performs ensemble-based inference by combining the pitch predictions from both models. This integration was achieved through output averaging, resulting in improved pitch accuracy and stability. The fusion strategy proved especially effective in handling variations in pitch dynamics and acoustic characteristics present in real-world audio recordings [5], [8].

Our evaluation methodology employed both objective and visual analysis. Quantitative assessment was conducted using established pitch evaluation metrics: Root Mean Square Error (RMSE), Mel Cepstral Distortion (MCD), and Gross Pitch Error (GPE) [9], [10]. These metrics enabled us to benchmark the performance of our proposed models against the original baselines.

The results showed that both enhancements led to substantial improvements. The enhanced PESTO model consistently outperformed the original across male and female voice samples, achieving lower error rates and demonstrating more stable pitch trajectories. Likewise, ESCAPE achieved the lowest RMSE, MCD, and GPE scores across different datasets, affirming the strength of the fusion strategy [25].

In addition to numerical evaluations, we visualized the predicted pitch contours over time, highlighting the temporal stability and fidelity of the predictions. These plots clearly demonstrated how the enhanced PESTO reduced erratic fluctuations compared to the original, and how ESCAPE maintained smoother and more coherent pitch trajectories. Special attention was given to the differences between male and female vocal samples, which helped explain the observed trends and reinforced the need for robust, generalizable models [1], [3].

The training process for the enhanced PESTO model utilized the MDB-stem-synth dataset, a musically diverse and richly annotated dataset of vocal recordings. Training was performed with a ResNet1d encoder architecture incorporating attention modules, using loss functions cross-entropy, equivariance, and shift-invariance objectives. Learning parameters like a batch size and learning rate were carefully configured to ensure stable convergence and optimal performance [5], [19], [22].

Overall, the proposed contributions provide effective solutions for improved monophonic pitch estimation, balancing accuracy, efficiency, and robustness. The enhancements to PESTO and the design of ESCAPE together advance the field's capabilities, making them promising candidates for applications in music information retrieval, speech analysis, and expressive audio processing.

2. Future Work

Building on the promising results obtained through the enhanced PESTO architecture and the ESCAPE fusion model, several exciting avenues remain open for future research and development. These directions aim to further improve pitch estimation accuracy, broaden the application scope, and increase robustness in real-world environments.

- **Integration into Speech Synthesis Pipelines**

One compelling direction for future work is to integrate the ESCAPE model or the enhanced PESTO estimator into state-of-the-art speech synthesis systems, such as vocoders or TTS (Text-to-Speech) frameworks. Accurate and smooth pitch estimation is critical for generating natural-sounding speech, especially in expressive or prosodically rich contexts. By serving as a high-fidelity F0 input module, ESCAPE could help guide pitch contour generation, leading to improvements in speech naturalness and prosody modeling [11, 17].

- **Specialization for Emotional and Expressive Speech**

While the models demonstrated robustness under various vocal characteristics, further enhancements could be made by tailoring the architecture specifically for emotional speech. This could involve training on emotion-rich corpora, adjusting the feature extraction layers to account for greater pitch variability, or even developing a dedicated emotion-aware pitch estimation algorithm. Incorporating emotion recognition features or prosodic classifiers into the architecture may also boost generalization across speaking styles [24, 28,].

- **Multispeaker and Multipitch Environments**

Another promising direction is extending the current single-pitch estimation framework to support multipitch detection, particularly for scenarios involving polyphonic audio, ensemble singing, or conversational speech with speaker overlap. Future models could leverage source separation techniques or attention-guided tracking to isolate and estimate multiple concurrent F0 trajectories. This would broaden the usability of your models in music information retrieval and advanced dialogue systems [14, 16, 29].

- **Incorporation of Recent Deep Learning Innovations**

To further enhance model performance, recent advancements in deep learning could be explored, including:

- Transformer-based pitch modeling to better capture long-range temporal dependencies in audio.
- CBAM (Convolutional Block Attention Module) or other attention mechanisms to refine feature selection within convolutional blocks.
- Diffusion models or self-supervised pretraining, which could offer better representations of time-varying pitch contours with minimal labeled data.

Such modern architectures could be combined with your attention-augmented ResNet design to further improve performance and generalizability[15], [30].

- **Computational Efficiency and Deployment**

Although the proposed models remain relatively lightweight, future work may explore efficiency optimizations to facilitate deployment in real-time or edge computing settings. Techniques such as model pruning, quantization, or knowledge distillation can be employed to reduce model size and inference time without sacrificing accuracy. These optimizations are essential for deploying pitch estimation models in mobile applications, voice assistants, or embedded systems [11,13].

- **Broader Evaluation Framework**

To gain a deeper understanding of model behavior in diverse contexts, future evaluations could include:

- Testing across a wider range of languages and accents.
- Cross-dataset benchmarking with studio-quality, noisy, or spontaneous recordings.
- Human listening studies, to validate the perceptual quality of pitch estimation and its impact on synthesized speech.

Such evaluations would help establish the real-world applicability and performance limits of the proposed models.

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Dr. Mohammed Salah Al-Radhi, whose unwavering support, encouragement, and invaluable guidance have been instrumental throughout the course of this research. His mentorship has not only enriched this thesis but has also profoundly shaped my academic and personal growth. I am sincerely grateful for his dedication and for being the greatest supervisor one could hope for.

I extend my heartfelt appreciation to the Department of Telecommunications and Artificial Intelligence at Budapest University of Technology and Economics for providing an excellent academic environment and the computational resources essential for the training and evaluation of the models developed in this work.

On a personal note, I am profoundly thankful to my mother and my siblings for their endless love, encouragement, and belief in me. Their support has been the foundation of my strength and perseverance, and I owe much of my success to their presence in my life.

Lastly, I would like to thank my friends and classmates for their camaraderie, support, and for always being there whenever I needed them. Their companionship has made this journey all the more meaningful and enjoyable.

Publications

[1] Zineb Hammadi, Dr. M. S. Al-Radhi, ESCAPE: (Emotion Self-Supervised Context Aware Pitch Estimation), WINS, 2025, 6 pages.

[2] Z. Hammadi, Dr. M. S. Al-Radhi, Self-Supervised Pitch Estimation with Contrastive Learning, TDK conference, 2024.

List of Figures

List of Tables

References

[1] L. R. Rabiner, “On the use of autocorrelation analysis for pitch detection,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 25, no. 1, pp. 24–33, 1977.

[2] A. Klapuri and M. Davy (Eds.), *Signal Processing Methods for Music Transcription*, Springer, 2006.

[3] T. Drugman, R. Saeidi, T. Kinnunen, and P. Alku, “Voice source modeling for speaker recognition: From analysis to synthesis,” *Speech Communication*, vol. 75, pp. 97–111, 2015.

[4] D. Gerhard, *Pitch Extraction and Fundamental Frequency: History and Current Techniques*, Tech. Rep. TR-CS 2003-06, Dept. of Computer Science, University of Regina, Regina, Canada, Nov. 2003.

[5] K. Koutini, A. Mesaros, and G. Widmer, “PESTO: Pitch estimation with self-supervised transposition-equivariant objective,” *arXiv preprint arXiv:2209.05829*, Sep. 2022. Doi: 10.48550/arXiv.2209.05829.

[6] R. M. Bittner, B. McFee, and J. P. Bello, “Multitask Learning for Fundamental Frequency Estimation in Music,” Sep. 2018.

[7] Hu, J., Shen, L., & Sun, G. (2018). *Squeeze-and-Excitation Networks*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7132–7141.

[8] N. ChTu, “FCPE: Fast and Context-based Pitch Estimation,” *GitHub Repository*, 2023. [Online]. Available: https://github.com/CNChTu/FCPE

[9] T. Nakatani and T. Yoshioka, “Metrics for robust pitch estimation: RMSE, GPE, and MCD,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 1, pp. 174–187, 2019.

[10] R. M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, “MedleyDB: A multitrack dataset for annotation-intensive MIR research,” in Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR), 2014, pp. 155–160.

[11] J. Z. Lin, Pitch Estimation for Analysis and Classification of Speech Situations, Master’s thesis, Harvard University, Extension School, June 2018.

[12] P. de la Cuadra, A. Master, and C. Sapp, “Efficient Pitch Detection Techniques for Interactive Music,” *Center for Computer Research in Music and Acoustics*, Stanford University, Jan. 2001.

[13] M. Mauch and S. Dixon, “pYIN: A fundamental frequency estimator using probabilistic

threshold distributions," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 659–663.

[14] A. Camacho and J. G. Harris, "A sawtooth waveform inspired pitch estimator for speech and music," *J. Acoust. Soc. Am.*, vol. 124, no. 3, pp. 1638–1652, Oct. 2008.

[15] K. Kasi and S. A. Zahorian, "Yet another algorithm for pitch tracking," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, May 2002, vol. 1, pp. 361–364.

[16] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "CREPE: A convolutional representation for pitch estimation," in *Proc. IEEE Int. Conf. Acoustics, Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 161–165, doi: 10.1109/ICASSP.2018.8461329.

[17] A. Kroon, "Comparing Conventional Pitch Detection Algorithms with a Neural Network Approach," ECSE 523 Speech Communications Final Project, Dept. Elect. Comput. Eng., McGill Univ., Montreal, QC, Canada, Jun. 2022.

[18] Brown, J. C. (1991). Calculation of a constant Q spectral transform. *The Journal of the Acoustical Society of America*, 89(1), 425–434.

[19] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[20] C. J. Hsu and J. Y. Liu, "MIR-1K: A Dataset for Singing Voice Separation," *NTU Speech Lab*, 2009.

[21] N. Srivastava, G. Hinton, A. Krizhevsky, et al., "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, 2014.

[22] A. Kendall and R. Cipolla, "Geometric Loss Functions for Camera Pose Regression with Deep Learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5974–5983, 2017.

[23] J. Li, T. Chen, R. Shi, Y. Lou, Y.-L. Li, and C. Lu, "Localization with Sampling-Argmax," *arXiv preprint arXiv:2110.08825*, Oct. 2021.

[24] K. Zhou, B. Sisman, R. Rana, B. W. Schuller, and H. Li, "Speech Synthesis with Mixed Emotions," IEEE Transactions, Dec. 28, 2022.

[25] J. Kominek, T. Schultz, and A. W. Black, "Synthesizer Voice Quality of New Languages Calibrated with Mean Mel Cepstral Distortion," Carnegie Mellon University, 2008.

[26] E. S. Kumar, K. J. Surya, K. Y. Varma, A. Akash, and K. N. Reddy, "Noise Reduction in Audio File Using Spectral Gating and FFT by Python Modules," Recent Developments in Electronics and Communication Systems, IOS Press, 2023, pp. 510–515.

[27] T. Li, "JL-Corpus," [Online]. Available: https://github.com/tli725/JL-Corpus.git.

[28] M. S. Al-Radhi, O. Abdo, T. G. Csapó, S. Abdou, G. Németh, and M. Fashal, "A

continuous vocoder for statistical parametric speech synthesis and its evaluation using an audio-visual phonetically annotated Arabic corpus," Department of Telecommunications and Media Informatics, Budapest University of Technology and Economics, Hungary; Phonetics and Linguistics Department, Alexandria University, Egypt; Faculty of Computers and Information, Cairo University, Egypt; MTA-ELTE Lendület Lingual Articulation Research Group, Hungary, May 2018; revised May 2019; accepted Sep. 2019; available online Sep. 29, 2019.

[29] M. Morrison, C. Hsieh, N. Pruyne, and B. Pardo, "Cross-domain Neural Pitch and Periodicity Estimation," Aug. 12, 2024.

[30] J. Lindenberger, *Classical Estimation versus Machine Learning for Pitch Estimation*, Master's thesis, Institute of Signal Processing, Johannes Kepler University Linz, Austria, July 2023.

[31] E. Azarov, M. Vashkevich, and A. Petrovsky, "Instantaneous Pitch Estimation Based on RAPT Framework," in *Proc. 20th European Signal Processing Conference (EUSIPCO)*, Bucharest, Romania, Aug. 27–31, 2012.

Appendix A- Implementation of the SE block

```
from functools import partial

import torch
import torch.nn as nn


class ToeplitzLinear(nn.Conv1d):
    def __init__(self, in_features, out_features):
        super(ToeplitzLinear, self).__init__(
            in_channels=1,
            out_channels=1,
            kernel_size=in_features+out_features-1,
            padding=out_features-1,
            bias=False
        )

    def forward(self, input: torch.Tensor) -> torch.Tensor:
        return super(ToeplitzLinear, self).forward(input.unsqueeze(-
2)).squeeze(-2)


class SEBlock(nn.Module):
    def __init__(self, channels, reduction=16):
        super(SEBlock, self).__init__()
        self.pool = nn.AdaptiveAvgPool1d(1)
        self.fc = nn.Sequential(
            nn.Linear(channels, channels // reduction, bias=False),
            nn.ReLU(inplace=True),
            nn.Linear(channels // reduction, channels, bias=False),
            nn.Sigmoid()
        )

    def forward(self, x):
        # x: (batch, channels, freq_bins)
        b, c, _ = x.size()
        y = self.pool(x).view(b, c)
        y = self.fc(y).view(b, c, 1)
        return x * y.expand_as(x)

class Resnet1d(nn.Module):
    """
    Basic CNN similar to the one in Johannes Zeitler's report,
    but for longer HCQT input (always stride 1 in time)
    Still with 75 (-1) context frames, i.e. 37 frames padded to each side
    The number of input channels, channels in the hidden layers, and output
    dimensions (e.g. for pitch output) can be parameterized.
    Layer normalization is only performed over frequency and channel
dimensions,
    not over time (in order to work with variable length input).
    Outputs one channel with sigmoid activation.

    Args (Defaults: BasicCNN by Johannes Zeitler but with 6 input
channels):
        n_chan_input:     Number of input channels (harmonics in HCQT)
        n_chan_layers:    Number of channels in the hidden layers (list)
        n_prefilt_layers: Number of repetitions of the prefiltering layer
        residual:         If True, use residual connections for
prefiltering (default: False)
```

```
        n_bins_in:        Number of input bins (12 * number of octaves)
        n_bins_out:       Number of output bins (12 for pitch class, 72 for
pitch, num_octaves * 12)
        a_lrelu:          alpha parameter (slope) of LeakyReLU activation
function
        p_dropout:        Dropout probability
    """

    def __init__(self,
                 n_chan_input=1,
                 n_chan_layers=(20, 20, 10, 1),
                 n_prefilt_layers=1,
                 prefilt_kernel_size=15,
                 residual=False,
                 n_bins_in=216,
                 output_dim=128,
                 activation_fn: str = "leaky",
                 a_lrelu=0.3,
                 p_dropout=0.2,
                 use_attention=False):

        super(Resnet1d, self).__init__()

        self.hparams = dict(n_chan_input=n_chan_input,
                            n_chan_layers=n_chan_layers,
                            n_prefilt_layers=n_prefilt_layers,
                            prefilt_kernel_size=prefilt_kernel_size,
                            residual=residual,
                            n_bins_in=n_bins_in,
                            output_dim=output_dim,
                            activation_fn=activation_fn,
                            a_lrelu=a_lrelu,
                            p_dropout=p_dropout)

        if activation_fn == "relu":
            activation_layer = nn.ReLU
        elif activation_fn == "silu":
            activation_layer = nn.SiLU
        elif activation_fn == "leaky":
            activation_layer = partial(nn.LeakyReLU,
negative_slope=a_lrelu)
        else:
            raise ValueError

        n_in = n_chan_input
        n_ch = n_chan_layers
        if len(n_ch) < 5:
            n_ch.append(1)

        # Layer normalization over frequency and channels (harmonics of
HCQT)
        self.layernorm = nn.LayerNorm(normalized_shape=[n_in, n_bins_in])

        # Prefiltering
        prefilt_padding = prefilt_kernel_size // 2
        self.conv1 = nn.Sequential(
            nn.Conv1d(in_channels=n_in,
                      out_channels=n_ch[0],
                      kernel_size=prefilt_kernel_size,
                      padding=prefilt_padding,
                      stride=1),
```

```
            activation_layer(),
            nn.Dropout(p=p_dropout)
        )
        self.se_block = SEBlock(n_ch[0])  # SE after first conv
        self.n_prefilt_layers = n_prefilt_layers
        self.prefilt_layers = nn.ModuleList(*[
            nn.Sequential(
                nn.Conv1d(in_channels=n_ch[0],
                          out_channels=n_ch[0],
                          kernel_size=prefilt_kernel_size,
                          padding=prefilt_padding,
                          stride=1),
                activation_layer(),
                nn.Dropout(p=p_dropout)
            )
            for _ in range(n_prefilt_layers-1)
        ])
        self.residual = residual

        conv_layers = []
        for i in range(len(n_chan_layers)-1):
            conv_layers.extend([
                nn.Conv1d(in_channels=n_ch[i],
                          out_channels=n_ch[i + 1],
                          kernel_size=1,
                          padding=0,
                          stride=1),
                activation_layer(),
                nn.Dropout(p=p_dropout)
            ])
        self.conv_layers = nn.Sequential(*conv_layers)

        self.flatten = nn.Flatten(start_dim=1)
        self.fc = ToeplitzLinear(n_bins_in * n_ch[-1], output_dim)

        self.final_norm = nn.Softmax(dim=-1)

    def forward(self, x):
        r"""

        Args:
            x (torch.Tensor): shape (batch, channels, freq_bins)
        """
        x = self.layernorm(x)

        x = self.conv1(x)
        x = self.se_block(x)

        for p in range(0, self.n_prefilt_layers - 1):
            prefilt_layer = self.prefilt_layers[p]
            if self.residual:
                x_new = prefilt_layer(x)
                x = x_new + x
            else:
                x = prefilt_layer(x)

        x = self.conv_layers(x)
        x = self.flatten(x)

        y_pred = self.fc(x)
```

```
        return self.final_norm(y_pred)
```

Appendix B- Implementation of the softargmax

```
import torch
import torch.nn.functional as F

def reduce_activations(activations: torch.Tensor, reduction: str = "alwa",
beta: float = 1.0) -> torch.Tensor:
    r"""
    Args:
        activations: tensor of probability activations, shape (batch_size,
num_bins)
        reduction (str): reduction method to compute pitch out of
activations,
            choose between "argmax", "mean", "alwa", "softargmax".
        beta (float): sharpness for softargmax; higher beta makes it
peakier.

    Returns:
        torch.Tensor: pitches as fractions of MIDI semitones, shape
(batch_size)
    """
    device = activations.device
    num_bins = activations.size(1)
    bps, r = divmod(num_bins, 128)
    assert r == 0, "Activations should have output size
128*bins_per_semitone"

    all_pitches = torch.arange(num_bins, dtype=torch.float,
device=device).div_(bps)

    if reduction == "argmax":
        pred = activations.argmax(dim=1)
        return pred.float() / bps

    if reduction == "mean":
        return torch.mm(activations, all_pitches)

    if reduction == "alwa":
        center_bin = activations.argmax(dim=1, keepdim=True)
        window = torch.arange(1, 2 * bps, device=device) - bps
        indices = (window + center_bin).clip_(min=0, max=num_bins - 1)
        cropped_activations = activations.gather(1, indices)
        cropped_pitches =
all_pitches.unsqueeze(0).expand_as(activations).gather(1, indices)
        return (cropped_activations * cropped_pitches).sum(dim=1) /
cropped_activations.sum(dim=1)

    if reduction == "softargmax":
        # Apply temperature-scaled softmax
        weights = F.softmax(activations * beta, dim=1)
        return (weights * all_pitches).sum(dim=1)

    raise ValueError(f"Unknown reduction type: {reduction}")
```