Received: 10 October 2023 | Revised: 31 January 2024 | Accepted: 8 March 2024

DOI: 10.4218/etrij.2023-0430

ORIGINAL ARTICLE

ETRI Journal WILEY

Efficient pitch-estimation network for edge devices

Chi Yoon Jeong[1,2] | **Youngmi Song**[1] | **Sungyong Shin**[1] | **Mooseop Kim**[1,2]

[1]Human Sensory Augmentation Research Section, Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea

[2]Artificial Intelligence Major, University of Science and Technology, Daejeon, Republic of Korea

Correspondence
Mooseop Kim, Human Sensory Augmentation Research Section, Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea.
Email: gomskim@etri.re.kr

Funding information
This work was supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean Government (MSIT) (no. RS-2023-00223440, Development of a multisensory music system and music education solution for the hearing-impaired).

Abstract

Pitch estimation is the task of finding the most conspicuous frequency in a complex audio signal. Many methods that use deep neural networks have significantly increased the accuracy of pitch estimation; however, their real-time performance results were achieved on high-performance devices. Because pitch estimation is widely used in real-time applications on low-power devices, we propose an efficient method for estimating pitch on edge devices. The network architecture of the proposed method uses a depth-scaling strategy and fully leverages convolutional networks. We further introduce a channel attention mechanism to increase accuracy without increasing computational overhead. We compared the proposed model with state-of-the-art (SOTA) and conventional methods using two public datasets. The experimental results show that the proposed method has a better classification accuracy than FCNF0++, which is the best performing SOTA model. Furthermore, it reduces the processing time obtained by FCNF0++ on a personal computer and two edge devices by 48% on average. These experimental results confirm that the proposed method efficiently classifies pitch on edge devices.

KEYWORDS

channel attention, edge device, fully convolutional network, fundamental frequency estimation, pitch estimation

1 | INTRODUCTION

The fundamental frequency is the lowest and most conspicuous frequency in an audio signal comprising complex periodic signals [1]. People perceive pitch based on fundamental frequencies. Thus, the fundamental frequency is associated with the physical properties of a signal, and pitch is associated with the perception of that signal. Although they are not equivalent, the fundamental frequency and pitch are highly correlated. They are hence used interchangeably in this study, as in Gfeller and others [2]. Estimating the fundamental frequency is essential in various sound signal processing tasks such as pitch annotation, speech recognition, melody extraction, and music information retrieval [3]. Therefore, many studies have been conducted to estimate pitch accurately [2–12].

Conventional methods [5, 6, 10] for estimating pitch primarily use digital signal processing (DSP) techniques and consist of three steps: preprocessing, candidate generation, and postprocessing. In the preprocessing step, framing is performed to capture an audio signal 10 ms in length. Then, pitch candidates are extracted using various functions such as the cumulative mean normalized difference and average magnitude difference. In the postprocessing step, the most predominant fundamental frequency is selected using argmax, the Viterbi algorithm,

or dynamic programming. However, these methods involve high computational costs and their performance degrades in noisy environments [9].

Recently, deep learning-based methods have outperformed conventional methods in various fields [13–15], and methods using deep neural networks (DNNs) have been proposed for the pitch-estimation task. Various network architectures, such as multilayer perceptrons [12], convolutional neural networks (CNNs) [2, 3, 7, 9], fully convolutional neural networks (FCNs) [4], and recurrent neural networks (RNNs) [8] have been used to estimate pitch accurately. In addition, residual learning using skip connections inspired by ResNet [16] has been used to increase the accuracy of pitch estimation [3, 7]. A pitch-estimation method [3] introduced variants of standard convolutions to reduce the computational cost. In addition, a recent study [11] proposed best practices for network hyperparameters and a network training strategy to increase the performance of existing models. Previous methods have mainly focused on increasing the accuracy of pitch estimation and have demonstrated real-time performance using high-performance graphic processing units (GPUs) [4, 11]. However, because pitch estimation is popularly used in real-time applications on embedded or mobile devices [17–19], research that considers the low computational power of edge devices is necessary so that pitch may be quickly estimated on edge devices.

Therefore, in this paper, we propose an efficient method for estimating the fundamental frequency on edge devices. We designed an efficient network structure based on network model scaling rules. The proposed method leverages an FCN to reduce the computational cost and introduces an attention module that can improve performance without additional computational burden. Experimental results revealed that the proposed network exhibited superior performance on public datasets in terms of both processing time and accuracy when compared with state-of-the-art (SOTA) methods.

The remainder of this paper is organized as follows: Section 2 briefly summarizes related work on pitch estimation, and Section 3 describes the proposed method for pitch estimation. Section 4 describes the experimental setup, and Section 5 presents the experimental results. Finally, Section 6 provides the concluding remarks.

2 | RELATED WORK

A representative and successful method using DSP techniques is YIN [6]. YIN introduced a cumulative mean normalized difference function to solve the problem of autocorrelation or difference functions. In addition, the fundamental frequency was estimated by determining the dip in the difference function. YIN can only estimate a single fundamental frequency, whereas probabilistic YIN (pYIN) [10] can estimate the probability of multiple fundamental frequencies using a modified YIN algorithm. However, pitch-estimation methods that use DSP techniques suffer from a high computational burden and are vulnerable to noise [9].

Pitch-estimation methods using deep learning have been proposed because deep learning-based methods have exhibited superior performance on various tasks [13–15]. Verma and Schafer [12] proposed end-to-end learning to estimate pitch states from the raw waveforms using DNNs. In this approach, various networks consisting of fully connected layers were constructed, and their performance was analyzed according to the number of layers and neurons. The best network model, consisting of three fully connected layers with 4096 neurons in each layer, achieved a performance similar to that of conventional methods.

CREPE [9] was the first method to introduce a CNN architecture to the pitch-estimation task. A network consisting of six convolutional layers and one fully connected layer was constructed. The kernel sizes for the first and later convolutional layers were set to 512 and 64, respectively. The network output contained 360 nodes that estimated the fundamental frequency from 32.70 Hz–1975.5 Hz. CREPE achieved a better performance than pYIN; however, it incurred a high computational cost.

To increase the accuracy of vocal pitch estimation, residual learning inspired by ResNet [16] was applied to a pitch-estimation network using a CNN architecture [7]. The pitch-estimation network [7] consisted of eight convolutional layers and a dense layer, and skip connections were employed to pass the identity information to every second convolutional layer. The network model has 76 states that estimate the fundamental frequency from 87.309 Hz–784.0 Hz and an additional state to indicate silence. Using residual learning, the network model can prevent the degradation in performance due to a deeper model.

Unlike previous studies based on models that classified discrete frequency states from audio signals, a regression model using an RNN was proposed to estimate pitch [8]. Two models for generating sinusoidal speech waveforms using RNNs from raw audio signals in the time domain and audio features in the frequency domain were proposed. The fundamental frequency was then estimated by applying an autocorrelation function to the generated sinusoidal speech waveforms.

Because a fully connected layer requires a large number of trainable parameters, an FCN [20], which replaces the fully connected layers with a convolutional layer, was used to reduce the computational cost of the network. In addition, an FCN can handle arbitrary input sizes

because the convolutional layers do not require a fixed-size input. Thus, FCNF0 [4], which uses the FCN architecture, was proposed to reduce the computational cost of a pitch-estimation network. This pitch-estimation network based on the FCN architecture consisted of seven convolutional layers and used a small filter size, such as 32 or 64, for the convolution operations. In addition, it used an 8-kHz sampling rate for input audio signals because the network model predicted pitch in a limited range of 30 Hz–1000 Hz. Three models with different numbers of convolutional and pooling layers were proposed. Among the three models, the best model, FCN-993, reduced the number of parameters to one third the number used by CREPE. In this paper, we refer to the FCN-993 model as the FCNF0 model.

In contrast to previous studies that used standard convolution, DeepF0 [3] introduced dilated convolution [21], which can effectively reduce the number of trainable parameters while increasing the size of the receptive field. In addition, DeepF0 used a residual connection [16], which can effectively propagate information from early layers to deeper layers in the network and prevent the vanishing gradient problem. For audio signals, a 16-kHz sampling rate was used, and 360 nodes were used to predict the fundamental frequency in a range of 32.70 Hz–1975.5 Hz, similar to CREPE. DeepF0 achieved better pitch-estimation accuracy than CREPE, despite having a quarter of the network parameters of CREPE.

Although many studies using DNNs have been conducted, the performance of the network models can vary depending on the training dataset, hyperparameters of the network, and training strategy. Thus, a recent study [11] proposed best practices for increasing the performance of previous methods such as CREPE [9], DeepF0 [3], and FCNF0 [4]. Through various experiments, it was shown that the best practices, including a larger batch size, higher frequency resolution, no early stopping, and no dropout, can be helpful in increasing the pitch-estimation accuracy. However, best practices have no effect on the overall structure of networks; they do not help reduce the inference time of the pitch-estimation networks but increase it slightly instead.

Although pitch-estimation methods using DNNs have shown impressive performance by focusing on improving estimation accuracy, it is necessary to consider the processing time for estimating pitch because pitch-estimation methods are commonly used in applications on mobile or edge devices. In addition, previous results [4, 11] on the processing time of pitch-estimation networks were measured on personal computers (PCs); therefore, it is necessary to measure the processing time of the methods on edge devices to demonstrate their performance on such devices.

3 | PROPOSED METHOD

The proposed pitch-estimation network leverages the FCN architecture because a previous study [11] showed that methods using the FCN architecture can estimate pitch more accurately than methods using the CNN architecture. In addition, a network model using the FCN architecture can effectively reduce the number of training parameters by replacing the fully connected layer with a convolutional layer. However, simply adopting the FCN architecture is not sufficient to create an efficient pitch-estimation network that can balance processing speed and pitch-estimation accuracy. The processing speed of a network model is affected by various factors, including the number of trainable parameters, type of convolution operations, and network structure [22]. Thus, we redesigned the network structure based on the FCN architecture to balance processing speed and pitch-estimation accuracy.

When designing a network architecture, there are no clear rules for ensuring optimal performance. However, there are many guidelines for network scaling that increase or decrease the network structure to adjust the network to various resource constraints or tasks [23]. These methods monotonically increase or decrease the number of channels, number of layers, or resolution of the feature map in the deeper layers of the network. Among these methods, depth scaling, which adjusts the network by tuning the number of channels, is widely used. The representative networks that use depth scaling are WideResNet [24] and MobileNets [25], which exhibit impressive performance gains. Therefore, we designed a pitch-estimation network using an FCN with a depth-scaling strategy. Figure 1 illustrates the proposed pitch-estimation network, which consists of six convolutional blocks, an attention layer, and a one-dimensional (1D) convolutional layer. Figure 1 shows that the proposed network increases the number of channels (d) as the layers become deeper.

The network structures of the proposed method and a previous method [4] are compared in Figure 2. This figure shows the change in the resolution and the number of channels from the first layer to the last layer of the network. In Figure 2, the proposed network monotonically increases the number of channels from the first layer to the last layer, whereas the network proposed by Ardaillon and Roebel [4] varies the number of channels independent of the position of the layers. Figure 2 further shows that the proposed network is more computationally efficient than the previous network because the previous network applies a large number of convolution filters on the high-resolution input, resulting in significant computational costs.

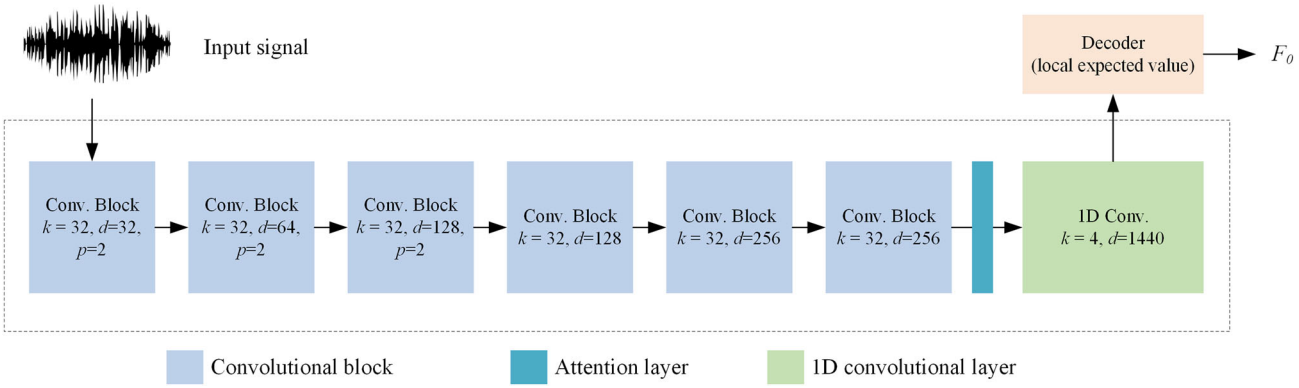


FIGURE 1 Proposed pitch-estimation network, which is based on the fully convolutional neural network (FCN) architecture, where k represents the kernel size for 1D convolution, d represents the number of channels, and p represents the kernel size for pooling.

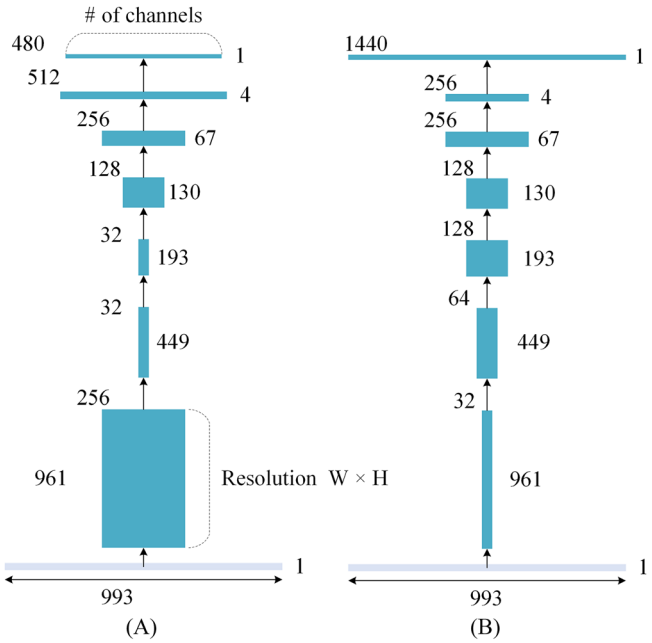


FIGURE 2 Comparison of network structures. (A) The network structure proposed by Ardaillon and Roebel [4] and (B) the proposed network structure using depth scaling.

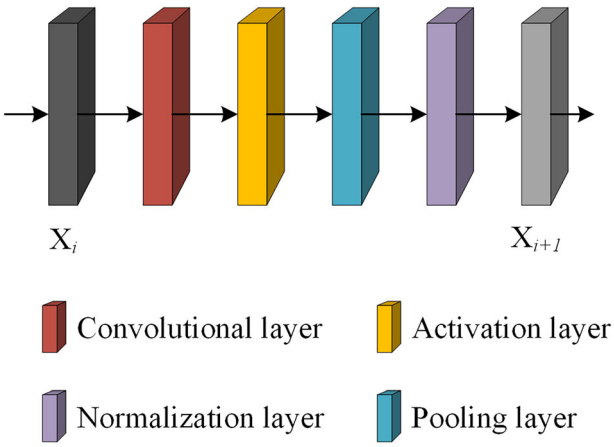


FIGURE 3 Structure of the convolutional block.

Each convolutional block in the proposed network consists of convolutional, activation, pooling, and normalization layers, as shown in Figure 3. All convolutional blocks have a 1D convolutional layer and a filter size of 32. A rectified linear unit is used as the activation layer in all convolutional blocks. For the first three convolutional blocks, max pooling is applied to summarize the features and reduce computational complexity. For the normalization layer, the layer normalization method [26] is adopted because it performs better than the batch normalization method [11].

The channel attention mechanism is a promising method that can increase the performance of a network at minimal computational cost. Thus, several methods [27–29] have been studied for weighting the channels. However, the channel attention mechanism has not been well-studied in pitch-estimation tasks. Therefore, in this study, we introduce a channel attention method for pitch-estimation networks.

Many methods for calculating channel weights use dimensionality reduction to reduce computational cost; however, this causes performance degradation in channel attention prediction. Thus, a recent study proposed the efficient channel attention (ECA) module [28], which calculates the channel attention using 1D convolution efficiently without dimensionality reduction. In addition, the ECA module has been integrated into CNNs for various tasks and has demonstrated clear performance gains [28, 30]. Thus, we adopted an ECA module for the pitch-estimation network.

Let the aggregated feature, which is calculated by applying channel-wise global average pooling to the feature map, be denoted $\boldsymbol{y}$. Then, the weight of each channel ω is calculated as follows [28]:

$$\omega = \sigma(C1D_k(\boldsymbol{y})), \tag{1}$$

where $C1D$ represents 1D convolution, k is the kernel size of the 1D convolution, and σ is the sigmoid function.

When estimating the fundamental frequency, the sampling rate of the input audio signal is an important factor that affects the pitch prediction accuracy. Unlike conventional methods [3, 9], which use high sampling rates such as 16 kHz, recent studies [4, 11] have demonstrated that using 8-kHz sampling rates can be helpful in increasing the accuracy of pitch estimation owing to the large amount of context information. Thus, the proposed network uses an 8-kHz sampling rate for the input signals.

The input signal sequentially passes through the proposed network, which consists of six convolutional blocks, an attention layer, and a 1D convolutional layer. The logits of the pitch bins are calculated in the last 1D convolutional layer. The logits of the pitch bins represent the musical intervals relative to the reference frequency. Given the pitch f and reference pitch f_{ref} in hertz, a cent is calculated by measuring the ratio between the two frequencies as follows:

$$¢(f) = 1200 \cdot \log_2 \frac{f}{f_{\text{ref}}}, \tag{2}$$

where the reference pitch is set to 31 Hz.

An octave, which is divided into 12 semitones, is represented by 1200 cents, with one semitone representing 100 cents. To increase the accuracy of the pitch estimation, we used 1440 pitch bins with intervals of 5 cents to cover the six octaves from C1 (32.70 Hz) to B7 (1975.5 Hz).

The logits of the pitch bins are fed into the decoder to calculate the fundamental frequency. Generally, argmax decoding is used; however, it suffers from compounding quantization errors [11]. Thus, we use locally normal decoding introduced by DeepF0 [3] to mitigate quantization errors. Locally normal decoding first determines the pitch bin with the highest value and then estimates the pitch value by calculating the weighted sum of the neighborhood bins. Given the logits of the pitch bins $\hat{y}$, the estimated pitch in cents using locally normal decoding is calculated using the following equation:

$$\hat{¢} = \frac{\sum_{i=m-W/2}^{m+W/2} ¢_i \hat{y}_i}{\sum_{i=m-W/2}^{m+W/2} \hat{y}_i}, \; m = \underset{i}{\text{argmax}}\, \hat{y}_i, \tag{3}$$

where W is the size of the neighborhood. We used 19 as the neighborhood size in the experiments.

The fundamental frequency (Hz) is calculated as follows:

$$\hat{f} = f_{\text{ref}} \cdot 2^{\hat{¢}/1200}. \tag{4}$$

When training the pitch-estimation network, the true fundamental frequency has continuous values; however, the network predicts the frequency using quantized bins with intervals. This causes boundary effects that degrade the network performance. To mitigate the boundary effects, existing methods [4, 9, 11, 31] primarily use techniques to blur training labels. Thus, we soften the true labels by applying Gaussian blurring. The softened label of ith bin y_i is generated as follows:

$$y_i = \exp\left(-\frac{(¢_i - ¢_{\text{true}})^2}{2 \cdot \sigma^2}\right), \tag{5}$$

where σ denotes the standard deviation. We used 25 as the standard deviation in the experiments.

The pitch-estimation model was trained to minimize the loss between the softened label y and the estimated logits of the pitch bin $\hat{y}$. In the experiment, we used categorical cross entropy as a loss function because it outperforms binary cross-entropy loss [11].

4 | EXPERIMENTAL SETUP

4.1 | Datasets

To evaluate the performance of the pitch-estimation network, we used two public datasets: music and speech. MDB-stem-synth [32] is the most popular music dataset in the field of pitch estimation. It consists of 230 tracks with various musical instruments, and the total duration of the dataset is 15.6 h. In addition, we used PTDB [33] to estimate speech pitch. It consists of 4718 speeches and the corresponding laryngographs recorded by 20 native English speakers, with a total length of 576 min. Following a previous study [11], we randomly split each dataset using a ratio of 70/15/15 for training, validation, and testing, respectively.

4.2 | Baselines

We compared the proposed methods with popular open-source programs and deep learning-based methods. The open-source programs contain two DSP-based methods, pYIN [10] and DIO [34], and one deep-learning method, torchcrepe [35], which is a PyTorch version of CREPE [9].

The deep learning-based methods include CREPE [9], DeepF0 [3], and FCNF0 [4]. A recent study [11] proposed best practices to increase the performance of pitch-estimation networks such as CREPE, DeepF0, and FCNF0. Thus, we also compared the performances of CREPE++, DeepF0++, and FCNF0++, which adopted these best practices.

A 16-kHz sampling rate was applied to pYIN, DIO, torchcrepe, CREPE, and DeepF0 as the original methods, and the other methods used an 8-kHz sampling rate.

4.3 | Implementation details

The proposed method used a window size of 1024 with 10 ms intervals for input signals with an 8-kHz sampling rate. An Adam optimizer with a learning rate of 2×10^{-4} was used to train the network models. In addition, the batch size and number of iterations were set to 128 and 2.5×10^5. The PyTorch framework was used to implement the proposed pitch prediction network and compare the methods. A PC equipped with an Intel Xeon Gold 6244 (3.6 GHz) and two NVIDIA A100 GPUs was used to train the networks. Distributed data-parallel training and automatic mixed precision methods were adopted to train the proposed network. The versions of PyTorch, CUDA, and cuDNN were v1.12, v11.3, and v8.2.1, respectively.

4.4 | Evaluation metrics

Given the true pitch f and estimated pitch $\hat{f}$ in hertz, the absolute difference in cents is defined as follows:

$$\text{¢}(f, \hat{f}) = |1200 \cdot \log_2(f/\hat{f})|. \tag{6}$$

To evaluate the accuracy of pitch estimation, we used the average difference between the true and estimated cents (Δ¢) using the following equation:

$$\Delta\text{¢} = \frac{1}{N}\sum_{i=0}^{N-1} \text{¢}(f_i, \hat{f}_i), \tag{7}$$

where N is the total number of samples.

Additionally, raw pitch accuracy (RPA) and raw chroma accuracy (RCA) were used as performance metrics. The RPA is the fraction of results in which the absolute difference in cents between the estimated and true pitch values is lower than a threshold. The RCA is similar to the RPA but ignores the octave change. Thus, the performance gap between the RCA and RPA indicates errors due to the octave shift. The RPA and RCA are calculated using the following equations:

$$\text{RPA}_\tau = \frac{1}{N}\sum_{i=0}^{N-1} [[\text{¢}(f, \hat{f}) < \tau]], \tag{8}$$

$$\text{RCA}_\tau = \frac{1}{N}\sum_{i=0}^{N-1} [[\text{¢}(f, \hat{f})\%1200 < \tau]], \tag{9}$$

where $[[\cdot]]$ is the binary operator that returns one if the expression is true and zero otherwise. The threshold (τ) was set to 50 cents in the experiments.

To assess the processing speed of the pitch-estimation methods, we measured the real-time factor (RTF) [11] by dividing the total processing time by the total number of samples in the two public datasets. The total processing time includes the processing time for all steps, such as data loading, pitch prediction, and postprocessing for inference, and uses the average values measured five times. When measuring the RTF, the batch size was set to 2048 for the PC and 128 for the edge device.

4.5 | Edge devices

Pitch-estimation methods are popular for edge devices such as smartphones, tablets, and embedded devices. Therefore, we measured the processing speed of the proposed method and compared its performance on various edge devices. Deep learning-based methods require GPU acceleration for fast computation; therefore, Jetson AGX Orin and Jetson Orin NX, which can utilize GPU computing, were used as edge devices. Detailed specifications of the edge devices are listed in Table 1. To measure the processing speed of the pitch-estimation methods, we used the maximum power mode of each edge device and locked the maximum clocks of the CPU and GPU by disabling dynamic voltage and frequency scaling.

4.6 | Evaluation procedure

The overall workflow of the performance analysis is illustrated in Figure 4. First, we trained the proposed pitch-estimation networks and the compared methods using the training datasets including MDB-stem-synth and PTDB on a PC. Then, we measured the pitch-estimation accuracy and processing time of each pitch-estimation-network using the trained models and test dataset on a PC. Finally, we measured the processing speed of each pitch-estimation network using the trained models and test dataset on two edge devices: AGX Orin and Orin NX.

TABLE 1 Detailed information on the edge devices used for real-time performance measurement.

	Jetson AGX Orin	Jetson Orin NX
AI performance	275 TOPS	100 TOPS
GPU	2048-core Ampere architecture	1024-core Ampere architecture
GPU max frequency	1.3 GHz	918 MHz
Tensor cores	64	32
CPU	12-core Arm Cortex-A78AE	8-core Arm Cortex-A78AE
CPU max frequency	2.2 GHz	2 GHz
Memory	32GB 256-bit LPDDR5	16GB 128-bit LPDDR5
Power	15–60 W	10–25 W
JetPack	5.1.1	5.1.1

Abbreviations: AI, artificial intelligence; LPDDR5, low power double data rate 5; TOPS, tera operations per second.

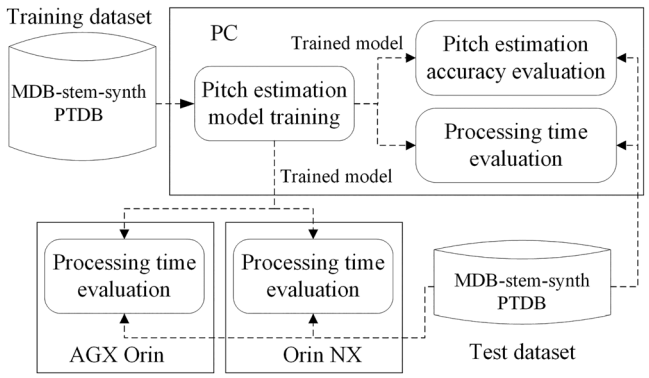


FIGURE 4 Flow of the evaluation procedure for measuring the performance of the pitch-estimation network on various devices.

5 | EXPERIMENTAL RESULTS

The quantitative evaluation results of the proposed method and compared methods are presented in Table 2. The evaluation metrics were computed using MDB-stem-synth and PTDB. In Table 2, the DSP-based methods, DIO and pYIN, yield a relatively lower accuracy of pitch estimation than the deep learning-based methods. In addition, the methods using the 8-kHz sampling rate, including the proposed method, achieved better results than the methods using the 16-kHz sampling rate, namely torchcrepe, CREPE, and DeepF0. FCNF0++, the SOTA method adopting the FCN architecture, only achieved a performance gain in terms of the average difference between the true and estimated cents with respect to DeepF0++, which adopts the CNN architecture; however, the proposed method achieved superior performance in all evaluation metrics when compared with the other methods. This implies that the proposed network improves performance because the network structure has been redesigned based on the FCN architecture.

TABLE 2 Performance comparison of the proposed method and compared methods.

Methods	Δ¢↓	RPA ↑	RCA ↑
pYIN [10]	110.51	0.8477	0.8643
DIO [34]	80.10	0.6961	0.7043
torchcrepe [35]	59.40	0.9103	0.9183
CREPE [9]	21.07	0.9748	0.9780
CREPE++ [11]	15.15	0.9783	0.9817
DeepF0 [3]	20.12	0.9778	0.9811
DeepF0++ [11]	12.66	0.9828	0.9856
FCNF0 [4]	18.01	0.9719	0.9748
FCNF0++ [11]	12.45	0.9825	0.9853
Ours	**12.02**	**0.9831**	**0.9860**

Note: Quantitative results were calculated using MDB-stem-synth and PTDB. The bold numbers indicate the best performance.
Abbreviations: RCA, raw chroma accuracy; RPA, raw pitch accuracy.

We also measured the processing speed of the proposed and compared methods on several devices: a PC and two edge devices and the Jetson AGX Orin and Jetson Orin NX. The RTFs of the various pitch-estimation methods are listed in Table 3; the RTFs of the DSP-based methods pYIN and DIO in a GPU computing environment are not listed. Table 3 shows that CREPE and its variants using a standard convolution operation and fully connected layers have three or four times more trainable parameters than the other methods. Although FCNF0 uses the FCN architecture to reduce the number of trainable parameters, it has 30% more parameters than DeepF0, which uses dilated convolution. Methods using the best practices proposed by Morrison and others [11], CREPE++, DeepF0++, and FCNF0++, have more trainable parameters than the original methods because of their higher frequency resolution. The proposed network has only 10% more trainable parameters than DeepF0, which uses dilated convolution, although it uses a higher-frequency resolution, which causes an increase in the number of trainable parameters. When comparing DeepF0++ with the proposed network at the same frequency resolution, the proposed network reduced the number of trainable parameters by 32%. This implies that we can reduce the number of trainable parameters in the proposed network by designing a network structure that uses a depth-scaling strategy.

TABLE 3 Processing time comparison of the proposed and compared methods on various devices.

Methods	Params.	PC environment		Jetson AGX Orin		Jetson Orin NX	
		RTF (GPU)	RTF (CPU)	RTF (GPU)	RTF (CPU)	RTF (GPU)	RTF (CPU)
pYIN [10]	-	-	0.0496	-	0.0566	-	0.0624
DIO [34]	-	-	**0.0166**	-	**0.0186**	-	**0.0203**
torchcrepe [35]	22.2 M	0.0073	0.4420	0.0560	3.6235	0.1416	3.9098
CREPE [9]	22.2 M	0.0046	0.2712	0.0334	2.8000	0.0843	2.8150
CREPE++ [11]	25.0 M	0.0049	0.2749	0.0373	2.8087	0.0916	2.8324
DeepF0 [3]	5.0 M	0.0067	0.8158	0.0537	8.0339	0.1365	10.0438
DeepF0++ [11]	8.3 M	0.0075	0.8467	0.0597	8.1177	0.1537	10.0711
FCNF0 [4]	6.7 M	0.0014	0.0668	0.0117	0.7557	0.0251	0.8157
FCNF0++ [11]	8.9 M	0.0016	0.0690	0.0133	0.7994	0.0287	0.8896
Ours	5.6 M	**0.0008**	0.0324	**0.0074**	0.5557	**0.0143**	0.5850

Note: The bold numbers indicate the best performance.
Abbreviation: RTF, real-time factor.

Table 3 further lists the model complexities of the various pitch-estimation networks and processing times for the PC, Jetson AGX Orin, and Jetson Orin NX devices. In Table 3, CREPE has more trainable parameters than the other methods; however, it has a shorter processing time than DeepF0. This implies that the processing speed of the network model depends on various factors in addition to the number of training parameters and floating-point operations per second, as indicated in a recent study [22]. Although FCNF0 has a slightly larger number of trainable parameters than DeepF0, it has a higher processing speed than DeepF0. However, it exhibited superior performance only with regard to the average difference between the true and estimated cents, as shown in Table 2. In addition, Table 3 reveals that the recent methods CREPE++, DeepF0++, and FCNF0++, which apply the best practices for improving the performance of existing methods, require additional computational cost to increase the accuracy of the pitch estimation.

Even though the proposed method has better classification performance than the SOTA methods DeepF0++ and FCNF0++, it still reduces the number of trainable parameters by 32% and 37%, respectively. Additionally, the proposed method reduced the processing time on the PC by 89% and 50% compared with the times of DeepF0++ and FCNF0++, respectively. These experimental results were the same for the processing time measurements on both edge devices. The proposed method reduced the processing time on the Jetson AGX Orin by 88% and 44% compared with the times of DeepF0++ and FCNF0++, respectively. Moreover, the proposed method reduced the processing time on the Jetson Orin NX by 91% and 50% compared with the times of DeepF0++ and FCNF0++, respectively. These experimental results confirm that the proposed network is more efficient than the other methods because it can balance the processing time and pitch-estimation accuracy.

When comparing the results for the PC environment in Table 3, the processing speed of the deep learning-based methods using GPU acceleration was significantly higher than those of the DSP-based methods. However, the processing speeds of deep-learning methods supported by GPU acceleration on edge devices significantly decreased, whereas DSP-based methods run in CPU environments have similar processing speeds. The proposed model, accelerated by GPU computing, outperformed all other methods in terms of inference speed on all platforms, including DSP-based methods run in CPU environments. Additionally, the proposed model is the only method that can estimate the pitch in a processing time of less than 15 ms on edge devices. The experimental results demonstrate that the proposed method can be practically used in applications that employ the pitch-estimation method for edge devices.

We also assessed the pitch-estimation performance based on network structure and attention mechanisms. Table 4 shows that the proposed network structure designed using the depth-scaling strategy can effectively decrease the processing time while simultaneously increasing the pitch-estimation accuracy. We also assessed the performance of four attention mechanisms, namely squeeze-and-excitation (SE) [27], convolutional block attention module (CBAM) [29], effective SE (eSE) [36], and ECA [28] by using them to replace the attention layers. The experimental results showed that the ECA method increases the pitch-estimation accuracy achieved by the other attention methods and incurs

TABLE 4 Performance comparison on edge devices according to the attention mechanism and network structure.

Methods	Δ¢↓	RTF (GPU)	
		AGX Orin	Orin NX
FCNF0++ [11]	12.45	0.0133	0.0287
Proposed network	12.22	0.0073	0.0142
+ SE [27]	12.55	0.0074	0.0143
+ CBAM [29]	12.61	0.0075	0.0145
+ eSE [36]	12.34	0.0073	0.0143
+ ECA [28]	12.02	0.0074	0.0143

Abbreviations: CBAM, convolutional block attention module; ECA, efficient channel attention; eSE, effective SE; FCN, fully convolutional neural network; RTF, real-time factor; SE, squeeze-and-excitation.

almost no computational burden. These experimental results confirm that the proposed method efficiently classifies pitch in edge devices.

6 | CONCLUSION

We proposed an efficient method for estimating pitch on edge devices. The proposed method uses a network architecture with a depth-scaling strategy that increases the number of channels as the resolution decreases. In addition, the proposed method leverages the FCN architecture to reduce computational complexity. Furthermore, we introduced a channel-attention mechanism for pitch estimation. The proposed network adopts the ECA module, which leverages 1D convolution to efficiently estimate channel attention without the overhead of computational cost. We compared the proposed model with SOTA- and DSP-based methods using two public datasets: MDB-stem-synth and PTDB. The experimental results demonstrate that the proposed model outperforms SOTA and DSP-based methods in terms of accuracy. Additionally, we compared the processing times on various devices, a PC and two edge devices. Although the proposed method achieved better classification performance than DeepF0++ and FCNF0++, which performed the best among the SOTA methods, it reduced the number of trainable parameters by 32% and 37%, respectively. In addition, the proposed method reduced the average processing time on all devices by 89% and 48%, respectively, compared with the times of DeepF0++ and FCNF0++. From these experimental results, it can be confirmed that the proposed method efficiently classifies the pitch in edge devices and is the only deep-learning method that outperforms all other methods in terms of inference speed on all platforms, including DSP-based methods that run in CPU environments.

Although the depth-scaling strategy of the proposed method leads to impressive results, its results depend on human intervention and the domain knowledge of experts. Recently, neural architecture search methods [37, 38] have been studied to determine the optimal network structure for a given task without human intervention. In addition, the network obtained by neural architecture search methods has shown promising results and can balance processing time and accuracy by adjusting the cost function. In the future, we plan to develop a pitch-estimation network using neural architecture search methods to optimize the performance of pitch estimation on edge devices.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflict of interest.

ORCID

Chi Yoon Jeong https://orcid.org/0000-0001-7089-2516
Mooseop Kim https://orcid.org/0000-0003-4914-0584

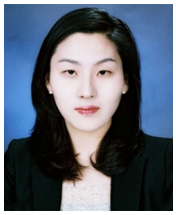
REFERENCES

1. P. de la Cuadra, A. S. Master, and C. S. Sapp, Efficient pitch detection techniques for interactive music, (International Conference on Mathematics and Computing), 2001.
2. B. Gfeller, C. Frank, D. Roblek, M. Sharifi, M. Tagliasacchi, and M. Velimirović, *Spice: self-supervised pitch estimation*, IEEE/ACM Trans. Audio, Speech Lang. Proc. **28** (2020), 1118–1128, DOI 10.1109/TASLP.2020.2982285.
3. S. Singh, R. Wang, and Y. Qiu, Deepf0: end-to-end fundamental frequency estimation for music and speech signals, (ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Toronto, Canada), 2021, pp. 61–65.
4. L. Ardaillon and A. Roebel, Fully-convolutional network for pitch estimation of speech signals, (Proc. Interspeech, Graz, Austria), 2019, pp. 2005–2009, DOI 10.21437/Interspeech.2019-2815.
5. A. Camacho and J. G. Harris, *A sawtooth waveform inspired pitch estimator for speech and music*, The J. Acoust. Soc. Am. **124** (2008), no. 3, 1638–1652, DOI 10.1121/1.2951592.
6. A. de Cheveign and H. Kawahara, *YIN, a fundamental frequency estimator for speech and music*, The J. Acoust. Soc. Am. **111** (2002), no. 4, 1917–1930, DOI 10.1121/1.1458024.
7. M. Dong, J. Wu, and J. Luan, Vocal pitch extraction in polyphonic music using convolutional residual network, (Proc. Interspeech, Graz, Austria), 2019, pp. 2010–2014, DOI 10.21437/Interspeech.2019-2286.
8. A. Kato and T. H. Kinnunen, *Statistical regression models for noise robust f0 estimation using recurrent deep neural networks*, IEEE/ACM Transa. Audio, Speech, Lang. Process. **27** (2019), no. 12, 2336–2349.
9. J. W. Kim, J. Salamon, P. Li, and J. P. Bello, Crepe: *A convolutional representation for pitch estimation*, (IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, Canada), 2018, pp. 161–165.
10. M. Mauch and S. Dixon, PYIN: A fundamental frequency estimator using probabilistic threshold distributions, (IEEE

International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy), 2014, pp. 659–663.
11. M. Morrison, C. Hsieh, N. Pruyne, and B. Pardo, Cross-domain neural pitch and periodicity estimation, arxiv preprint, 2023, DOI 10.48550/arXiv.2301.12258.
12. P. Verma and R. W. Schafer, Frequency estimation from waveforms using multi-layered neural networks, (Proc. Interspeech, San Francisco, USA), 2016, pp. 2165–2169.
13. S. B. Alex and L. Mary, *Variational autoencoder for prosody-based speaker recognition*, ETRI J. **45** (2023), no. 4, 678–689. DOI 10.4218/etrij.2021-0377.
14. S. Seo, and H. Jung, *A robust collision prediction and detection method based on neural network for autonomous delivery robots*, ETRI J. **45** (2023), no. 2, 329–337. DOI 10.4218/etrij.2021-0397.
15. C. Jeong, H. Yang, and K. Moon, *Horizon detection in maritime images using scene parsing network*, Electron. Lett. **54** (2018), no. 12, 760–762, DOI 10.1049/el.2018.0989.
16. K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, (IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA), 2016, pp. 770–778.
17. M. D. Fletcher, N. Thini, and S. W. Perry, *Enhanced pitch discrimination for cochlear implant users with a new haptic neuroprosthetic*, Sci. Rep. **10** (2020), no. 1, 10354, DOI 10.1038/s41598-020-67140-0.
18. K. Ishii, Y. Kinoshita, Y. Wakabayashi, and N. Ono, *Real-time pitch visualization with "blinky" sound-to-light conversion device*, J. Signal Process. **25** (2021), 213–220.
19. S. Shin, C. Oh, and H. Shin, Tactile tone system: a wearable device to assist accuracy of vocal pitch in cochlear implant users, (Proceedings of the 22nd International ACM Sigaccess Conference on Computers and Accessibility, ASSETS '20, Virtual event, Greece), 2020, DOI 10.1145/3373625.3418008.
20. J. Long, E. Shelhamer, and T. Darrell, Fully convolutional networks for semantic segmentation, (IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA), 2015, pp. 3431–3440.
21. L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, *DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFS*, IEEE Trans. Pattern Anal. Mach. Intell. **40** (2018), no. 4, 834–848.
22. W. Chen, X. Gong, X. Liu, Q. Zhang, Y. Li, and Z. Wang, FasterSeg: searching for faster real-time semantic segmentation, arXiv preprint, 2019, DOI 10.48550/arXiv.1912.10917.
23. M. Tan and Q. Le, Efficient net: rethinking model scaling for convolutional neural networks, *Edited by* K. Chaudhuri and R. Salakhutdinov, in Proceedings of Machine Learning Research, Vol. **97**, PMLR, 2019, pp. 6105–6114. https://proceedings.mlr.press/v97/tan19a.html
24. S. Zagoruyko and N. Komodakis, Wide residual networks, (Proc. British Machine Vision Conference, York, UK), 2016.
25. A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, MobileNets: efficient convolutional neural networks for mobile vision applications, arXiv preprint, 2017, DOI 10.48550/arXiv.1704.04861.
26. J. L. Ba, J. R. Kiros, and G. E. Hinton, Layer normalization, arXiv preprint, 2016, DOI 10.48550/arXiv.1607.06450.
27. J. Hu, L. Shen, and G. Sun, Squeeze-and-excitation networks, (Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA), 2018, DOI 10.1109/CVPR.2018.00745.
28. Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, ECA-Net: efficient channel attention for deep convolutional neural networks, (Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA), 2020, DOI 10.1109/CVPR42600.2020.01155.
29. S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, CBAM: convolutional block attention module, (Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany), 2018, DOI 10.1007/978-3-030-01234-2_1.
30. C. Y. Jeong, K. Moon, and M. Kim, *An end-to-end deep learning approach for real-time single image dehazing*, J. Real-Time Image Process. **20** (2023), no. 1, DOI 10.1007/s11554-023-01270-2.
31. R. M. Bittner, B. McFee, J. Salamon, P. Li, and J. P. Bello, Deep salience representations for F0 estimation in polyphonic music, (Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China), 2017, pp. 63–70.
32. J. Salamon, R. M. Bittner, J. Bonada, J. J. Bosch, E. Gómez, and J. P. Bello, An analysis/synthesis framework for automatic F0 annotation of multitrack datasets, (Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China), 2017, pp. 71–78.
33. G. Pirker, M. Wohlmayr, S. Petrik, and F. Pernkopf, A pitch tracking corpus with evaluation on multipitch tracking scenario, (Interspeech, Florence, Italy), 2011, pp. 1509–1512.
34. M. Morise, H. Kawahara, and H. Katayose, Fast and reliable f0 estimation method based on the period extraction of vocal fold vibration of singing voice and speech, (AES 35th International Conference: Audio for Games, London, UK), 2009.
35. M. Morrison, torchcrepe, 2022. https://github.com/maxrmorrison/torchcrepe
36. Y. Lee and J. Park, Centermask: real-time anchor-free instance segmentation, (IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA), 2020, pp. 13903–13912, DOI 10.1109/CVPR42600.2020.01392.
37. H. Xiao, Z. Wang, Z. Zhu, J. Zhou, and J. Lu, Shapley-NAS: Discovering operation contribution for neural architecture search, (IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA), 2022, pp. 11882–11891.
38. P. Ye, B. Li, Y. Li, T. Chen, J. Fan, and W. Ouyang, *β-DARTS: beta-decay regularization for differentiable architecture search*, (IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) New Orleans, LA, USA), 2022, pp. 10864–10873.

AUTHOR BIOGRAPHIES

Chi Yoon Jeong received his BS and MS degrees in Electronic and Electrical Engineering from Pohang University of Science and Technology, Pohang, Republic of Korea, in 2002 and 2004, respectively, and his PhD degree in Computer Science from the Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea, in 2018. Since 2004, he has worked at the Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea, where he is now a Principal Researcher. His current research interests include computer vision, pattern recognition, machine learning, and on-device artificial intelligence.



Youngmi Song received her PhD in Management Information Systems from the School of Business Administration, Kyungpook National University, Daegu, Republic of Korea, in 2013. Since 2013, she has worked at the Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea, where she is now a Senior Researcher. Her main research interests are artificial intelligence, multimodal interfaces, and human–computer interaction.

Sungyong Shin received his BS and MS degrees in Computer Science from the Department of Computer Science and Engineering, Korea University, Seoul, Republic of Korea, in 2008 and 2010, respectively. He is currently studying for his PhD degree in Industrial Design at the Korea Advanced Institute of Science and Technology, Daejeon, Republic of Korea. Since 2010, he has worked at the Electronics and Telecommunications Research Institute, Daejeon, Republic of Korea, where he is now a Senior Researcher. His main research interests include accessibility interfaces through sensory substitution.

Mooseop Kim received his MS in Electrical Engineering from Kyungpook National University, Daegu, Republic of Korea, and a PhD degree in Computer Science and Engineering from Chungnam National University, Daejeon, Republic of Korea, in 1998 and 2008, respectively. He was a Research Engineer in the Organic LED Group at the Device and Materials Laboratory, LG Electronics Institute of Technology, Seoul, Republic of Korea, from 1998 to 1999. Since 1999, he has been with the Electronics and Telecommunications Research Institute, Daejeon, Korea, where he is currently a Principal Researcher. His current research interests include wearable computing, activity recognition, and sensory substitution.

How to cite this article: C. Y. Jeong, Y. Song, S. Shin, and M. Kim, *Efficient pitch-estimation network for edge devices*, ETRI Journal **47** (2025), 112–122, DOI 10.4218/etrij.2023-0430.