https://doi.org/10.7840/kics.2023.48.9.1064

중첩 U-NET 기반 음성 향상 기술의 실시간 구현

차 재 빈◆, 황 서 림*, 박 성 욱**, 박 영 철°

A Real-Time Implementation of a Nested U-NET-Based Speech Enhancement

Jae-Bin Cha◆, Seo-Rim Hwang*, Sung Wook Park**, Young-Cheol Park°

요 약

음성 향상 기술은 음성의 잡음을 제거하고 명료도를 높이는 기술로서, 음성 인식, 화상 회의 등 다양한 분야에서 활용된다. 최근에는 DNN 기반의 음성 향상 기술이 주목받고 있으며, 특히 음성 신호의 로컬 및 글로벌 정보를 효과적으로 활용하는 중첩 U-Net 모델이 우수한 성능을 보인다. 음성 향상 기술의 활용도를 높이기 위해서는 스마트폰 같은 엣지 디바이스에서 실시간 실행이 가능해야 한다. 본 논문에서는 중첩 U-Net 기반 최신 모델 중 하나인 NUNet-TLS를 스마트폰 앱에서 실시간 구현하였다. NUNet-TLS에 사용된 dilated convolution은 빈번한 메모리 사용으로 긴 연산 시간을 요구하는데, dilated convolution을 LSTM(Long-Short Term Memory)으로 대체함으로써 메모리 사용량과 연산 시간을 크게 줄일 수 있었다. 제안 모델은 데이터 입/출력을 프레임 단위로 처리하도록 변경하여 앱으로 구현되었으며, 이 앱은 실시간 실행에서 기존 모델보다 메모리 사용량이 적지만 동등한 음성 향상 성능을 보여주었다.

Key Words : Deep Learning, Speech Enhancement, Real-time Implementation, Mobile

ABSTRACT

Speech enhancement is a technology that removes noise and enhances speech intelligibility and is used in many fields, such as voice recognition, video conferencing, etc. Recently, DNN-based speech enhancement techniques have gained attention, and the Nested U-Net model, which effectively utilizes local and global information of the speech signal, has shown excellent performance. To extend the usage of speech enhancement, real-time execution should be possible on edge devices such as smartphones. In this paper, NUNet-TLS, one of the latest models based on Nested U-Net, was implemented in real-time in a smartphone app environment. Due to frequent memory usage, the dilated convolution used in NUNet-TLS requires a long computation time. Still, by replacing it with LSTM(Long-Short Term Memory), we significantly reduced memory usage and computation time. The proposed model was implemented as an app, with data input/output processed frame-by-frame. This app demonstrated lower memory usage than previous models while maintaining comparable speech enhancement performance in real-time execution.

◆ First Author : Yonsei University Department of Computer Science, jbcha7@yonsei.ac.kr, 학생회원
° Corresponding Author : Yonsei University Division of Software, young00@yonsei.ac.kr, 종신회원
* Yonsei University Department of Computer Science, allmindfine@yonsei.ac.kr, 학생회원
** Gangneung-Wonju National University Department of Electronic Engineering, swpark@gwnu.ac.kr, 정회원
논문번호 : 202305-111-A-RN, Received May 30, 2023; Revised July 12, 2023; Accepted July 12, 2023

Ⅰ. 서 론

음성 향상(speech enhancement) 기술은 배경 잡음으로부터 원하는 음성 신호를 분리하여 음성의 품질과 명료도를 향상시키는 기술이다. 음성 향상 기술은 보청기, 음성 인식, 화상 회의 등 다양한 분야의 전처리 기술로써 이용된다. 기존의 확률통계 기반 알고리즘은 시간에 따른 통계적 특성이 변화하는 경우, 잡음 제거가 어렵다는 단점이 있다. 한편, 최근 활발하게 연구되고 있는 DNN(Deep Neural Network) 기반 알고리즘들은 통계적 특성이 변화하는 상황에서 발생하는 다양한 유형의 소음에 대해서도 우수한 음성 향상 성능을 보인다[1].

DNN 기반 음성 향상은 일반적으로 음성 신호의 특징을 효과적으로 분석하기 위해 STFT(Short-Time Fourier Transform)을 이용하여 음성 신호를 시간 영역에서 시간-주파수 영역으로 변환한 뒤 향상된 음성을 스펙트럼 매핑 방법으로 직접 추정하거나, 잡음 제거를 위한 마스크를 추정한 후 마스크와 입력 음성을 곱하여 향상된 음성을 얻는 시간-주파수 마스킹 방법을 주로 사용한다[2].

최근 중첩 U-Net[3] (Nested U-Net) 모델을 사용한 음성 향상 기법이 제안되어 우수한 성능을 보인 바 있다. 중첩 U-Net은 다양한 스케일의 U-Net을 중첩하여 반복적으로 사용함으로써 작은 크기의 필터를 사용하면서도, 여러 스케일로 음성 신호를 분석할 수 있다. 보다 최근에는 중첩 U-Net의 계층 간 연결을 확장한 NUNet-TLS[4] (Nested U-Net with Two-Level Skip Connections)가 제안되어 기존 모델보다 우수한 성능을 보여주었다.

음성 향상 기술을 음성 인식이나 화상 회의 같은 여러 응용 분야에서 활용하려면 스마트폰 같은 다양한 장치에서 실시간 실행이 가능해야 한다. 이에 따라 딥러닝 모델을 엣지 디바이스에 구현하기 위한 연구가 활발히 진행되어왔다[5]. 실시간 구현을 위해 TFLite 플랫폼[6]을 사용할 수 있는데, 이는 딥러닝 모델의 학습된 가중치를 양자화하여 모델의 메모리를 줄임으로써 딥러닝 모델을 소형화할 수 있게 해 준다.

본 논문에서는 DNN 기반의 음성 향상 기술을 스마트폰 앱에서 실시간 구현하는 방법을 제안한다. 이를 위해 NUNet-TLS를 베이스라인 모델로 선정하고, 모델의 구조를 분석하여 실시간 구현에 용이하도록 개선한다. 최종적으로 개선된 모델을 TFLite를 이용하여 스마트폰 앱에서 실시간 구현한다.

논문의 2장에서 음성 향상 기술과 베이스라인 모델인 NUNet-TLS를 설명하고, 3장에서는 NUNet-TLS의 실시간 구현 측면에서의 구조적인 특징과 개선 방법을 제시한다. 4장에서는 개선한 음성 향상 모델을 스마트폰 앱으로 구현하여 동작과 성능을 분석하고, 5장에서는 개선한 모델의 성능과 구현된 앱의 실시간 성능을 비교, 분석한 후, 6장에서 결론을 제시한다.

Ⅱ. DNN 기반 음성 향상 모델

2.1 DNN 기반의 음성 향상

본 논문에서는 음성 향상 기법으로 시간-주파수 마스킹 기법을 이용하였고, 프레임 단위로 입력되는 신호에 대해 시간-주파수 마스킹 기법을 적용한 DNN 기반 음성 향상 수식은 다음과 같다.

$$|X_{t,f}|e^{j\theta_{X_{t,f}}} = |Y_{t,f}|e^{j\theta_{Y_{t,f}}} + |N_{t,f}|e^{j\theta_{N_{t,f}}} \tag{1}$$

$$\hat{M} = \Xi(|X_{t,f}|) \tag{2}$$

$$\hat{Y}_{t,f} = \hat{M} \cdot |X_{t,f}|e^{j\theta_{X_{t,f}}} \tag{3}$$

$$\hat{y}_t = ISTFT(\hat{Y}_{t,f}) \tag{4}$$

위 수식의 $|X_{t,f}|$, $|Y_{t,f}|$, $|N_{t,f}|$는 차례대로 잡음이 섞인 음성 신호 x_t, 깨끗한 음성 신호 y_t, 잡음 n_t를 STFT를 이용하여 주파수 영역으로 변환한 신호의 크기를 의미하며, 수식 (2)의 Ξ는 음성 향상 모델을 나타낸다. 모델을 통해 추정한 마스크 $\hat{M}$을 기존의 음성 신호와 곱하여 잡음을 제거한다. 잡음이 제거된 음성 신호는 ISTFT(Inverse STFT)를 통하여 시간 영역 신호 $\hat{y}_t$로 복원하고, 프레임 간 중첩-합 과정을 거쳐 최종 향상된 음성 신호를 얻는다.

2.2 베이스라인 모델

NUNet-TLS는 중첩 U-Net에 추가적인 skip connection과 CTFA(Causal Time-Frequency Attention)을 사용함으로써 우수한 음성 향상 성능을 달성한다. 그림 1은 NUNet-TLS의 구조를 나타내는데, 인코더 단계와 디코더 단계 사이에는 다운 샘플링(D·S)과 업 샘플링(U·S) 계층이 존재하며, 병목 블록(Bottleneck Block)은 Dilated Dense 블록[7]으로 구성된다. 인코더-디코더 단계 안에 중첩된 U-Net은 입력 계층(I·L), 인코더 계층(E·L), 병목 블록(B·B), 디코더 계층(D·L)으로 구성된다. 이때 인코더와 디코더 계층은 L-계층을 이루며 이때 $L = \{6,5,4,3\}$이다.

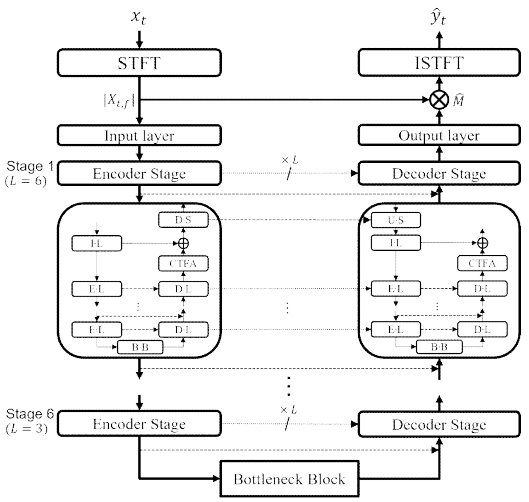


그림 1. 시간-주파수 마스킹 기법이 적용된 NUNet-TLS 기반 음성 향상 시스템 구조
Fig. 1. Structure of NUNet-TLS-based speech enhancement system with time-frequency masking technique

Ⅲ. 실시간 구현을 위한 모델 최적화

3.1 데이터 입출력 설계

일반적인 DNN 기반 음성 시스템에서는 특정 청크의 음성 클립을 입력받아 일괄 처리하는 방식을 사용한다. 그러나 이러한 방식은 실시간 구현에 적합하지 않기 때문에 본 논문에서는 그림 2와 같은 데이터 입출력 방식을 사용하였다.

실시간 시스템은 연속된 음성 입력을 중첩된 윈도우를 통해 얻은 짧은 프레임 단위로 분할하여 처리한다. 이 과정에서 출력된 프레임 $\hat{y}_t$으로부터 프레임 간의 중첩-합 과정을 통해 최종 향상된 음성 신호 b_t^{out}를 얻을 수 있다. 시간 t에서의 입력 프레임 x_t를 중심으로 프레임 간 중첩을 허용한다. 프레임 간에 중첩된 샘플 수는 실시간 시스템의 연산량과 직결된다. 본 논문에서는 길이가 512 샘플인 Hann 윈도우를 사용하였고 프레임 간 50% 중첩을 허용하였다. 따라서 샘플링 주파수가 16kHz일 경우, 입력 프레임당 허용 연산 시간은 16msec(256 샘플)가 된다. 한편, 그림 2와 같은 입출력 구조에서 모델에 입력과 출력 간에는 반 프레임(16msec)의 시간 지연이 발생하게 되는데, 이는 일반적인 음성 통신 시스템에서 허용 가능한 범위이다.

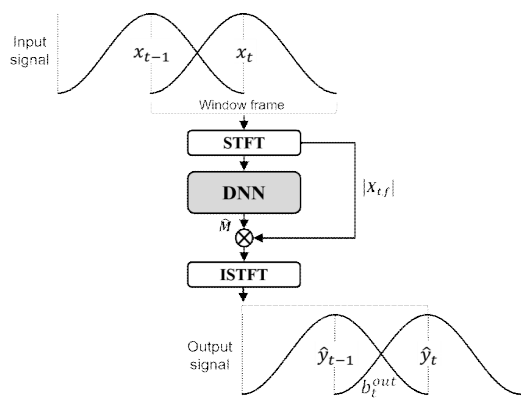


그림 2. 음성 향상 알고리즘의 실시간 처리 구조
Fig. 2. Real-time processing structure of speech enhancement algorithm

3.2 모델 최적화를 위한 연산 시간 분석

DNN 모델의 구조 최적화를 위해 먼저 DNN 모델을 구성하는 블록별 연산 시간을 측정하였다. 표 1은 그림 1의 베이스라인 DNN 모델에서 인코더 계층(E·L), 병목 블록(B·B), 디코더 계층(D·L)별 처리 시간을 측정한 결과이다. 처리 시간은 Intel(R) Core(TM) i7-11700 @ 2.50GHz를 탑재한 PC에서 측정하였으며, 텐서플로우(TF)에서 훈련된 모델을 TFLite 모델로 변환한 후, 블록별 1초 음성 처리 시간을 100번 측정하여 평균하였다.

측정 결과, 파라미터 수가 가장 작은 병목 블록이 전체 처리 시간 중 가장 긴 49%를 차지함을 보였다. 이는 병목 블록에서 사용하는 dilated convolution[8] 연산 과정에서 필요한 메모리 할당과 접근 횟수에 기인한다.

표 1 베이스라인 모델의 계층별 실시간 처리 시간
Table 1. Real-time processing time per layer in the baseline model

	Params [M]	Processing time [sec]	Ratio [%]
Encoder Layer	0.67	0.27	23
Bottleneck(DDense)	0.16	0.58	49
Decoder Layer	1.58	0.37	28

3.3 베이스라인 모델의 최적화

베이스라인 모델의 인코더, 디코더 계층과 병목 블록은 모두 (2, 3) 필터의 2D 컨볼루션 모듈을 사용한다. 그림 3은 컨볼루션 모듈의 시간 축 필터 크기에 따른 수용 영역(receptive field)[9]을 예시하여 보여준다. 수용 영역은 컨볼루션 출력 프레임 $\hat{y}_t$을 생성하기 위해 사용되는 입력 프레임 x_t의 범위를 의미한다. 시간 축 필터 크기가 2인 그림 3(a)의 경우에는 출력 프레임 $\hat{y}_t$를 얻기 위해 3 프레임의 입력이 필요하지만, 시간 축 필터 크기가 1인 (b)에서는 현재 입력 프레임만으로 출력을 얻을 수 있다. 컨볼루션 모듈의 시간 축 필터 크기

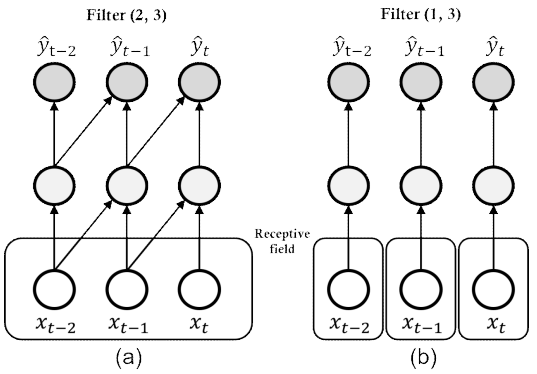


그림 3. 컨볼루션 모듈의 필터 크기에 따른 수용 영역
Fig. 3. Receptive field according to filter size of convolution layer

가 2 이상일 경우, 필터 크기만큼의 과거 입력 프레임을 확보하기 위한 추가적인 버퍼가 필요하며, 이 과정에서 발생하는 버퍼의 메모리 할당과 접근 시간이 처리 시간을 지연시킨다. 따라서 컨볼루션 모듈의 시간 축 필터 크기를 줄임으로써 DNN 모델의 처리 시간을 단축할 수 있다. 하지만 인코더와 디코더 블록의 컨볼루션 필터 크기는 음성 향상 모델의 성능에 큰 영향을 미치기 때문에 본 논문에서는 베이스라인 모델에서 사용하는 (2, 3) 크기의 2D 컨볼루션 모듈을 그대로 사용하였다.

그림 4는 모델의 병목 블록에서 사용하는 dilated convolution의 동작을 나타낸다. Dilated convolution은 인코더와 디코더 블록과 마찬가지로 (2, 3) 컨볼루션 모듈을 사용한다. 그러나 기존의 컨볼루션 모듈과 달리 필터 연산을 확장률(dilation rate)만큼의 간격을 두어 수행하기 때문에 수용 범위가 매우 넓어진다. 따라서 dilated convolution은 기존의 컨볼루션과 비교하여 처리 시간을 더욱 지연시킨다. 베이스라인에서 사용되는 Dilated Dense 블록은 다량의 dilated convolution 계층을 사용하므로 처리 시간은 확장률과 계층 수에 비례하여 증가한다.

한편 dilated convolution과 같이 시간의 흐름에 따른 데이터의 정보를 활용하는 모듈로 LSTM(Long-Short Term Memory)[10]이 있다. LSTM은 망각 게이트를 통해 데이터의 중요도에 따라 저장과 삭제를 결정하며 이전 데이터를 다음 입력으로 전달하기 위해 상태 하나만 전달하면 되기 때문에 다량의 과거 데이터를 버퍼링하고 처리하는 데 소요되는 시간을 절감할 수 있다. 즉, LSTM을 사용하는 것이 dilated convolution보다 실시간 처리 면에서 더욱 효율적으로 동작할 수 있다.

실제로 베이스라인 모델의 병목 블록의 dilated

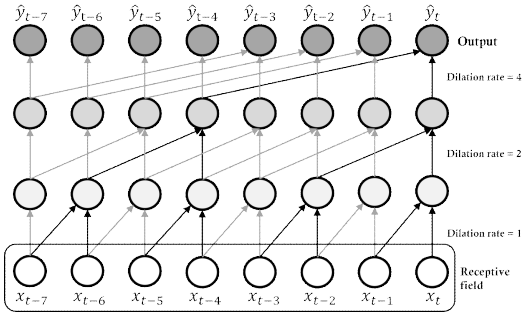


그림 4. Dilated convolution의 확장률에 따른 수용 영역
Fig. 4. Receptive field according to the dilation rate of the dilated convolution

convolution을 LSTM으로 바꾸어 처리 시간을 비교하여 보았다. 본 논문에서는 베이스라인 모델의 파라미터 수와 유사하게 유지하기 위하여 LSTM의 유닛을 21로 고정하여 실험하였다.

측정된 처리 시간을 표 2에 비교하였다. 측정 환경은 표 1과 동일하다. 표 2의 결과는 병목 블록이 LSTM으로 대체되었을 때 처리 시간이 대폭 감소함을 보여준다. 이는 앞서 설명한 바와 같이 dilated convolution 수행을 위해서는 빈번한 메모리 사용이 필요하고, 그 결과 과도한 연산 시간으로 이어지기 때문이다. 이러한 결과를 바탕으로 본 논문에서는 베이스라인 모델(NUNet-TLS)의 병목 블록을 기존의 Dilated Dense 블록에서 LSTM으로 변경하였다.

빈번한 메모리 사용이 연산 시간 지연의 원인임을 확인하기 위해 모델의 파라미터 수와 연산량을 비교하여 보았다. 제안된 모델의 파라미터 수는 베이스라인 모델과 유사한 대략 2.83M(Million)이며, 연산량은 초당 10.51G MACs(Giga Multiply-Accumulate operations)으로 측정되었다. 이는 베이스라인 모델의 10.75G MACs에 비해 소폭 하락한 결과이다. 이때 MACs는 MLTK(Silicon Labs Machine Learning Toolkit)[11]을 이용하였으며, 1초 음성의 추론 시간을 기준으로 측정

표 2. DDense(Dilated Dense 블록)을 사용하는 모델과 LSTM을 사용하는 모델의 계층별 실시간 처리 시간
Table 2. Real-time processing time per layer for a model using Dilated Dense Block (DDense) and a model using LSTM

	Processing time [sec]	
	DDense	LSTM
Encoder Layer	0.27	0.27
Bottleneck	0.58	**0.08**
Decoder Layer	0.37	0.37

하였다. 결과적으로 빈번한 메모리 사용이 처리 시간 지연의 주된 원인임을 확인할 수 있다.

Ⅳ. 실시간 음성 향상 앱

4.1 실시간 음성 향상 앱 구현

실시간 음성 향상 앱은 스마트폰 안드로이드 OS 환경에서 안드로이드 스튜디오(Android Studio)[12]를 통해 JAVA 언어로 구현하였다. 이때 스마트폰은 Samsung Galaxy Tab S7 FE를 사용하였다.

DNN 기반 실시간 음성 향상 앱은 다음과 같은 단계를 거쳐 구현하였다. 먼저, TF 기반 딥러닝 모델을 프레임 단위의 입출력이 가능하도록 변형하고 학습을 통해 얻은 가중치를 로드한다. 그 후 가중치를 32비트 부동소수점에서 8비트 정수로 양자화하여 TFLite 모델로 변환[13]하고 스마트폰 앱에 탑재하여 실시간 실행하도록 하였다.

그림 5는 앱의 UI와 동작을 나타내는데, 앱은 녹음, 오디오 업로드, 오디오 다운로드, 오디오 재생, 실시간 음성 향상 온/오프 기능을 제공한다. 녹음 혹은 오디오 업로드를 통해 실시간으로 음성을 입력시키고 음성 향상 버튼을 활성화하면 실시간으로 향상된 음성을 출력한다. 후에 출력된 음성을 다운로드하여 입력 음성과 비교할 수 있다.

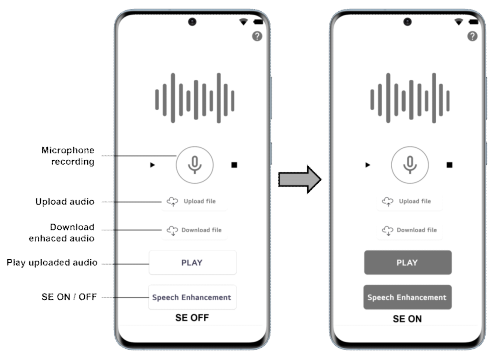


그림 5. 실시간 음성 향상 앱 UI 및 동작 설명
Fig. 5. Real-time speech enhancement app UI and description.

4.2 앱 구현 결과

음성 향상 시스템의 실시간 구현 여부를 확인하기 위해 RTF(Real-time Factor)을 측정하였다. RTF는 시스템의 추론 시간을 입력 시간으로 나눈 값으로, RTF가 1 미만일 경우 실시간 처리가 가능하다. RTF는 그림 6(a)에서 볼 수 있다. 베이스라인 모델로 설정한

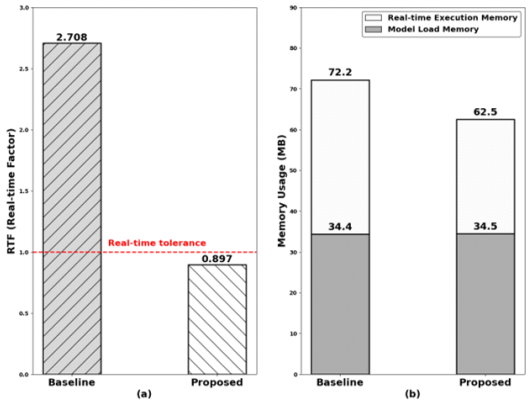


그림 6. 앱에서의 모델 별 RTF(Real-time Factor) 및 메모리 사용량
Fig. 6. Real-time factor (RTF) and memory usage per model in the app

NUNet-TLS를 변형 없이 실시간 구현하였을 때 RTF는 2.708로 1을 대폭 상회하여 실시간 처리가 불가능하지만, 병목 블록을 LSTM으로 변경한 제안된 모델은 RTF가 0.897로 1 이내로 처리됨을 확인할 수 있다.

그림 6(b)는 앱에 탑재된 TFLite 모델의 메모리 사용량을 비교하여 보여주고 있다. 메모리 사용량은 모델 적재 메모리와 실시간 실행 메모리로 구분하였다. 모델 적재 메모리는 음성 향상 모델이 기기에 로드된 후 증가한 메모리를 의미하며, 실시간 실행 메모리는 로드한 모델을 실시간 실행할 때 증가한 메모리를 의미한다. 이때 메모리 사용량은 안드로이드 스튜디오에서 제공하는 CPU 프로파일러를 사용하여 측정했으며, 최대 메모리 사용량을 기준으로 측정하였다.

그림 6의 (b)를 보면 베이스라인과 제안된 모델의 적재 메모리는 거의 같지만, 실시간 실행 메모리는 베이스라인 모델이 대략 10MB를 더 사용함을 알 수 있다. 메모리 사용량은 모델의 연산량과 파라미터에 따라 달라지는데[5], 두 모델이 파라미터와 MACs가 비슷함에도 실시간 실행 시 메모리 사용량이 차이가 나는 이유는 dilated convolution에 의한 메모리 할당과 접근 때문이다. RTF가 약 3배 가량 감소한 것도 dilated convolution을 LSTM으로 교체하여 메모리 사용을 줄였기 때문에 가능한 것이다.

Ⅴ. 실 험

5.1 음성 향상 모델의 훈련 및 테스트

훈련을 위한 음성 데이터는 음성 데이터는 81명의 영어권 발화자가 뉴스 기사를 읽은 22시간 30분의 Wall

Street Journal (WSJ0)[14] 데이터를 사용하였다. 입력 음성의 SNR은 0, 5, 10, 15dB의 SNR로 구성하였다. 이 때 잡음 데이터는 Chime2[15], Chime3[16], NoiseX-92[17] 데이터를 사용하였다. 샘플링 레이트는 16kHz이다. 테스트 데이터로는 훈련에 사용하지 않은 음성과 ETSI[18] 잡음 데이터를 사용하였다. 테스트 음성의 SNR 또한 0, 5, 10, 15dB로 구성하였다.

딥러닝 모델은 텐서플로우 프레임워크에서 훈련하였으며, 학습률은 0.001, 배치 크기는 3, 에포크는 100, 옵티마이저는 Adam을 사용하였다. 훈련 시 사용한 STFT의 FFT 길이는 32ms, 윈도우는 Hann 윈도우를 사용하였다. 손실 함수는 시간 영역 자승 평균 오차와 주파수 영역 스펙트럼 크기 차이를 조합하여 사용하였다[4].

$$L_t = \sum_t |\hat{x_t} - x_t|_2^2 \tag{1}$$

$$L_f = \sum_{t,f} ||\hat{X}_{t,f}| - |X_{t,f}||_1 \tag{2}$$

$$L = \lambda_1 L_t + \lambda_2 L_f \tag{3}$$

위 식에서 $|\cdot|_1$, $|\cdot|_2$는 각각 $l1\,norm$와 $l2\,norm$을 나타내며, λ_1, λ_2는 결합 계수이다.

5.2 훈련된 음성 향상 알고리즘의 성능

음성 향상 성능 평가를 위한 지표는 PESQ(Perceptual Evaluate of Speech Quality)와 STOI(Short Time Objective Intelligibility)를 사용하였다.

평가 결과를 표 3에서 확인할 수 있다. 제안된 모델의 PESQ는 베이스라인과 비교하여 약 0.09 증가하였으며 STOI도 향상되었다. 이러한 성능의 개선은 공정한 비교 실험을 위하여 파라미터 개수를 동일하게 맞추면서, LSTM으로 구현된 병목 블록이 Dilated Dense 블록으로 구현된 병목 블록보다 더 음성 향상에 기여한 결과로 추정된다.

표 3. PC에서 측정한 TF 모델의 음성 향상 성능
Table 3. Speech enhancement performance of Tensorflow model measured on PC

Model	Params [M]	PESQ				
		0dB	5dB	10dB	15dB	Avg.
Noisy	-	1.152	1.353	1.616	2.091	1.553
Baseline	2.83	2.292	2.803	3.201	3.588	2.971
Proposed		**2.404**	**2.905**	**3.290**	**3.651**	**3.062**

Model	Params [M]	STOI				
		0dB	5dB	10dB	15dB	Avg.
Noisy	-	0.801	0.892	0.945	0.973	0.902
Baseline	2.83	0.923	0.955	0.971	0.981	0.958
Proposed		**0.928**	**0.957**	**0.972**	**0.981**	**0.960**

5.3 구현된 앱의 실시간 성능 분석

마지막으로 그림 7에선 PC에서 실행된 TF 기반 음성 향상 시스템의 파일 단위 처리결과와 앱에서의 TFLite 기반 음성 향상 시스템의 프레임 단위 처리결과를 PESQ와 스펙트로그램을 통해 비교하고 있다.

그림 7(c)와 (d)를 보면 출력 신호의 스펙트로그램과 PESQ 결과가 매우 유사한 것을 확인할 수 있는데, 이는 TF를 기반으로 훈련된 원 모델과 TFLite를 거쳐 실시간 앱으로 구현된 모델의 음성 향상 성능이 유사함을 의미한다. 이 결과를 통해 NUNet-TLS 음성 향상 시스템이 스마트폰에서 성공적으로 실시간 구현되었음을 확인할 수 있다.

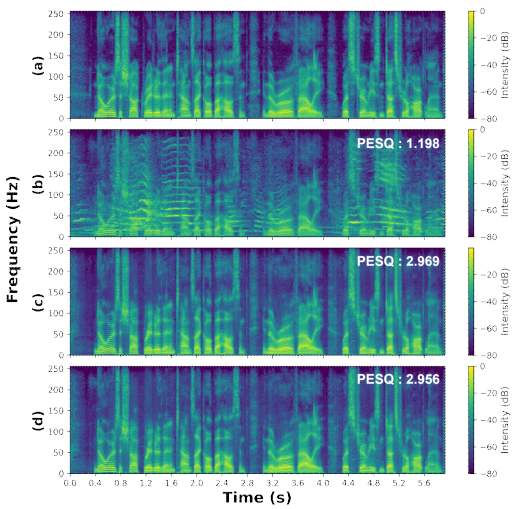


그림 7. (a) 잡음이 없는 깨끗한 타겟 음성의 스펙트로그램 (b) 잡음이 섞인 입력 음성의 스펙트로그램 (c) PC에서 TF 모델의 파일 단위의 추론을 통해 얻은 향상된 음성의 스펙트로그램 (d) 앱에서 TFLite 모델의 프레임 단위의 추론을 통해 얻은 향상된 음성의 스펙트로그램
Fig. 7. (a) Spectrogram of clean target speech without noise (b) Spectrogram of input speech mixed with noise (c) Spectrogram of enhanced speech through file-to-file inference in PC (d) Spectrogram of enhanced speech obtained through frame-by-frame inference of the TFLite model in the app

Ⅵ. 결 론

본 논문에서는 최근 우수한 성능을 보인 NUNet-TLS를 스마트폰 환경에서 실시간 음성 향상 앱으로 구현하였다. 베이스라인 모델은 병목 블록에 사용된 dilated convolution의 구조적 특성으로 인하여 추가적인 처리 시간 지연이 발생하고, 이로 인해 실시간 구현이 어려움을 확인하였다. 본 논문은 실시간 구현을 위하여 NUNet-TLS의 병목 블록을 LSTM으로 대체하였으며, 실험 결과 개선된 모델은 기존의 모델보다 실시간 처리 속도가 약 3배 빨라졌고, 음성 향상 성능 평가 지표인 PESQ 또한 소폭 증가하였음을 확인하였다. 개선된 모델은 TFLite를 통해 안드로이드 기반 스마트폰에서 동작하는 실시간 앱으로 최종 구현되었다.[1)]

References

[1] D. L. Wang and J. Chen, "Supervised speech separation based on deep learning: An overview," *IEEE/ACM Trans. Audio, Speech, and Lang. Process.,* vol. 26, no. 6, pp. 1702-1726, 2018. (https://doi.org/10.1109/taslp.2018.2842159)

[2] S.-R. Hwang, S. W. Park, and Y.-C. Park, "Performance comparison evaluation of real and complex networks for deep neural network-based speech enhancement in the frequency domain," *J. Acoustical Soc. Korea,* vol. 41, no. 1, pp. 30-37, 2022. (https://doi.org/10.7776/ASK.2022.41.1.030)

[3] Z. Zhou, et al., "Unet++: A nested u-net architecture for medical image segmentation," in *DLMIA 2018, and 8th Int. Wkshp., ML-CDS 2018, Held in Conjunction with MICCAI 2018*, pp. 1-10, Granada, Spain, Sep. 2018. (https://doi.org/10.1007/978-3-030-00889-5_1)

[4] S. Hwang, S. W. Park, and Y.-C. Park, "Monoaural speech enhancement using a nested u-net with two-level skip connections," in *Proc. Interspeech 2022,* pp. 191-195, 2022. (https://doi.org/10.21437/interspeech.2022-1002)

[5] M. Xu, et al., "A first look at deep learning apps on smartphones," *The World Wide Web Conf.*, pp. 2125-2136, May 2019. (https://doi.org/10.1145/3308558.3313591)

[6] *Tensorflow Lite*, Retrieved 2023, from https://www.tensorflow.org/lite

[7] X. Xiang, X. Zhang, and H. Chen, "A nested u-net with self-attention and dense connectivity for monaural speech enhancement," *IEEE Signal Process. Lett.,* vol. 29, pp. 105-109, 2021. (https://doi.org/10.1109/lsp.2021.3128374)

[8] A. Pandey and D. L. Wang, "TCNN: Temporal convolutional neural network for real-time speech enhancement in the time domain," *IEEE ICASSP 2019*, pp. 6875-6879, 2019. (https://doi.org/10.1109/icassp.2019.8683634)

[9] S. Drgas, "A survey on low-latency dnn-based speech enhancement," *Sensors,* vol. 23, no. 3, p. 1380, 2023. (https://doi.org/10.3390/s23031380)

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735-1780, 1997. (https://doi.org/10.1162/neco.1997.9.8.1735)

[11] *Silicon Labs Machine Learning Toolkit (MLTK),* Retrieved 2023, from https://siliconlabs.github.io/mltk/

[12] *Android Studio*, Retrieved 2023, from https://developer.android.com/

[13] P. P. Ray, "A review on TinyML: State-of-the-art and prospects," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 4, pp. 1595-1623, 2022. (https://doi.org/10.1016/j.jksuci.2021.11.019)

[14] D. B. Paul and J. Baker, "The design for the wall street journal-based CSR corpus," in *Proc. Speech and Natural Lang.*, New York, Feb. 23-26, 1992. (https://doi.org/10.21437/icslp.1992-277)

[15] E. Vincent, J. Barker, S. Watanabe, J. Le Roux, F. Nesta, and M. Matassoni, "The second 'chime' speech separation and

1) https://github.com/JaeBinCHA7/Nested-U-Net-based-Real-time-Speech-Enhancement-Mobile-App

recognition challenge: Datasets, tasks and baselines," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, pp. 126-130, 2013. (https://doi.org/10.1109/icassp.2013.6637622)

[16] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third 'chime' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. IEEE Wkshp. ASRU*, pp. 504-511, 2015. (https://doi.org/10.1109/asru.2015.7404837)

[17] A. Varga and H. J. M. Steeneken, "Assessment for automatic speech recognition: Ii. noisex-92: A database and an experiment to study the effect of additive noise on speech recognition systems," *Speech Commun*, vol. 12, no. 3, pp. 247-251, 1993. (https://doi.org/10.1016/0167-6393(93)90095-3)

[18] E. ETSI, "202 396-1: Speech quality performance in the presence of background noise," 2023.

차 재 빈 (Jae-Bin Cha)

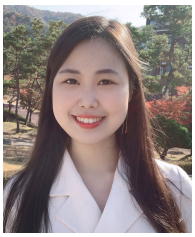
2023년 2월 : 연세대학교 컴퓨터정보통신공학부 학사

2023년 3월~현재 : 연세대학교 전산학과 석사과정

<관심분야> 음성향상, 음성신호처리, 디지털신호처리, 딥러닝

[ORCID:0000-0002-9294-8362]

황 서 림 (Seo-Rim Hwang)



2021년 8월 : 연세대학교 컴퓨터정보통신공학부 학사

2021년 9월~현재 : 연세대학교 전산학과 석사과정

<관심분야> 음성향상, 음성신호처리, 디지털신호처리, 딥러닝

[ORCID:0000-0002-7407-2440]

박 성 욱 (Sung Wook Park)

1993년 2월 : 연세대학교 전자공학과 학사

1995년 2월 : 연세대학교 전자공학과 석사

1998년 8월 : 연세대학교 전자공학과 박사

2009년 3월~현재 : 강릉원주대학교 전자공학과 교수

<관심분야> 음성향상, 딥러닝, 신호처리, 임베디드 시스템, 회로설계

[ORCID:0000-0002-6840-7013]

박 영 철 (Young-Cheol Park)

1986년 2월 : 연세대학교 전자공학과 학사

1988년 2월 : 연세대학교 전자공학과 석사

1993년 2월 : 연세대학교 전자공학과 박사

2002년 3월~현재 : 연세대학교 소프트웨어학부 교수

<관심분야> 디지털신호처리, 음성신호처리, 적응신호처리, 오디오신호처리, 딥러닝

[ORCID:0000-0003-3274-076X]