PDF Download
3439797.pdf
24 December 2025
Total Citations: 1
Total Downloads: 116

Published: 21 July 2021
Accepted: 01 November 2020
Revised: 01 November 2020
Received: 01 July 2020

Citation in BibTeX format

RESEARCH-ARTICLE

Deep Neural Network Based Noised Asian Speech Enhancement and Its Implementation on a Hearing Aid App

XIAOQIAN FAN, Zhejiang University, Hangzhou, Zhejiang, China

BOWEN YANG, Zhejiang University, Hangzhou, Zhejiang, China

WENZHI CHEN, Zhejiang University, Hangzhou, Zhejiang, China

QUANFANG FAN

Open Access Support provided by:

Zhejiang University

Transactions on Asian and Low-Resource Language Information Processing, Volume 20, Issue 5 (September 2021)
https://doi.org/10.1145/3439797
EISSN: 2375-4702

Deep Neural Network Based Noised Asian Speech Enhancement and Its Implementation on a Hearing Aid App

XIAOQIAN FAN, BOWEN YANG, and WENZHI CHEN, Zhejiang University
QUANFANG FAN, Hangzhou Youting Technology Co., Ltd.

This article studies noised Asian speech enhancement based on the deep neural network (DNN) and its implementation on an app. We use the THCHS-30 speech dataset and the common noise dataset in daily life as training and testing data of the DNN. To stack the frequency data of multiple audio frames to improve the effect of speech enhancement, the system compares the best number of stacked frames during training and testing. At the same time, the influence of training rounds on the PESQ is compared, and the best number of rounds is obtained. On this basis, the best model is implemented on the hearing aid app, and the real-time performance of the device is tested. The experiment shows that based on the DNN, using an appropriate number of rounds for training and using an appropriate number of audio frames stacking to improve the speech enhancement effect, and transplanting this speech enhancement model to the hearing aid app, can effectively improve speech clarity and intelligibility within a reasonable time delay range.

CCS Concepts: • **Computing methodologies → Feature selection**;

Additional Key Words and Phrases: Feature selection, neural networks, noise reduction, health care information systems, natural language processing

ACM Reference format:
Xiaoqian Fan, Bowen Yang, Wenzhi Chen, and Quanfang Fan. 2021. Deep Neural Network Based Noised Asian Speech Enhancement and Its Implementation on a Hearing Aid App. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 20, 5, Article 78 (July 2021), 14 pages.
https://doi.org/10.1145/3439797

1 INTRODUCTION

With the increase of the world's aging population, hearing aids are increasingly used to solve the problem of hearing loss in the elderly [1]. With the continuous popularization of hearing aids, people have higher and higher requirements for the quality of hearing aids, among which speech enhancement is a problem to be considered in the design process of hearing aids. Speech enhancement, also known as speech noise reduction, is to reduce the noise of the input speech, and finally output clean speech, so as to ensure that hearing loss patients can hear the voice of surrounding speakers. The goal of speech enhancement is to improve the definition and quality of degraded

Authors' addresses: X. Fan (corresponding author), B. Yang, and W. Chen, School of computer science and technology, Zhejiang University, No. 38, Zheda Road, Xihu District, Hangzhou, Zhejiang, 310007, China; emails: bungle@zju.edu.cn, yangbowen@zju.edu.cn, chenwz@zju.edu.cn; Q. Fan, Hangzhou Youting Technology Co., Ltd., No. 220, East Zone, Building A, No. 525, Xixi Road, Hangzhou, Zhejiang, 310007, China; email: fanqf@earhear.cn.

2375-4699/2021/07-ART78 $15.00
https://doi.org/10.1145/3439797

noisy speech signals under adverse conditions [2]. However, in the actual acoustic environment, the performance of speech enhancement is not always satisfactory.

In the past few decades, many speech enhancement methods have been developed. Some traditional speech enhancement methods include spectral subtraction [3] and Wiener filtering [4], among others. In the spectral subtraction method proposed by Boll [3], by intercepting the non-speech gap as the noise signal, and subtracting the estimated noise spectrum from the noisy audio spectrum in the frequency domain, the spectrum of the denoised speech signal can be obtained. Since the method is based on the assumption that the noise signal is generally stable or changes slowly, the problem of a residual noise spectrum easily appears in practical application, which leads to the music noise phenomenon. The Wiener filtering method proposed by Lim and Oppenheim [4] is the optimal filtering method in a statistical sense. In this method, a linear filter is used to process the input noisy speech signal to minimize the expected mean square error of the output clean speech signal. Although this method can change music noise into white noise, it easily causes speech distortion. Chen et al. [5] studied the quantization performance behavior of the Wiener filter in the context of noise reduction and proved that speech distortion can be better controlled in three different ways. In recent years, more and more machine learning algorithms have been used to solve speech enhancement problems. For example, Ephraim and Malah [6] introduced the MMSE estimator in their work [6]. In the work of Cohen and Berdugo [7] and Cohen [8], an improved OM-LSA speech estimator and MCRA noise estimator were proposed. Some supervised machine learning algorithms are also used more and more in the field of speech enhancement. Among them, the effectiveness of a **deep neural network (DNN)**-based speech enhancement method has been proved in many works.

In terms of structure, an early stage of the neural network model is the shallow model [9, 10]. In 2006, Hinton et al. [11, 12] proposed a greedy hierarchical unsupervised learning algorithm. Each layer is pretrained without supervision to learn the advanced representation of its input (or output of the previous layer). In the work of Xia and Bao [13] and Lu et al. [14], as a depth model, **stack denoising auto-coding (SDA)** is used to build a relationship model between clean and noise features.

The structure used by Xu et al. [15] is a feed-forward neural network with multiple nonlinear levels [16]. They are allowed to represent a highly nonlinear regression function and map noisy speech features to clean speech features. The type of hidden cell is sigmoid, and the output cell is linear. It has a visible layer of Gaussian variables connected to a hidden binary layer.

To avoid falling into local minimum [17] when training the deep network, the back-propagation algorithm based on MMSE objective function is used to train the DNN. Experiments show that the DNN performance of 11 frames of context speech frame and three layers of hidden layer is the best. At present, the speech enhancement function in these studies is only implemented on Windows or the Ubuntu host operating system, and it has not been transplanted to the hearing aid app. In addition, real-time use of the hearing aid app has not been considered.

Traditional speech enhancement methods have the advantages of a small amount of calculation, good real-time performance, and simple implementation. However, due to the limitation of many unreasonable assumptions, the performance of traditional speech enhancement is often not good in the noisy environment. The nonlinear relationship between noisy speech and enhanced pure speech can well estimated by the DNN model, which has been proved to have excellent noise reduction performance, especially in an unstable noise environment.

For Android-embedded devices, Google has launched TensorFlow Lite, a lightweight solution for TensorFlow, which is specially designed for mobile and embedded devices. It can run machine learning models locally in embedded systems without calling cloud computing resources. For the trained model, the translator provided by Google is used to convert it into a Lite model suitable for embedded devices. The Lite model changes the format of model storage and introduces optimization measures, which can reduce the size of the model file, improve the speed of model operation, and ensure accuracy of the model.

In this article, we refer to the algorithm of Xu et al. [15], transplant it to the Android system, modify the number of layers and stacked frames of the DNN, make a comparative experiment on the number of stacked frames (1–11 frames), and test the real-time performance of the voice enhancement system on a mobile phone.

The rest of the article is organized as follows. In Section 2, we introduce the related work. Section 3 outlines the speech enhancement system. Section 4 analyzes and discusses the experimental design. Section 5 analyzes and discusses the experimental results. Section 6 presents our summary and plans for future work.

2 RELATED WORK

In recent years, due to the rise of deep learning research, speech enhancement technology based on deep learning received more and more attention, showing a very bright application prospect, gradually becoming a new research trend in speech enhancement. At present, many speech enhancement methods based on deep learning have been proposed. In 1989, Tamura [18] used a four-layer feed-forward neural network for speech noise reduction. Each layer of the neural network contains 60 neurons. However, due to the limitation of computational power, compared with the traditional speech enhancement scheme, the noise reduction effect of this method is not ideal. Hinton et al. [19] show how to use "complementary priors" to eliminate the explaining-away effects that make inference difficult in densely connected belief nets that have many hidden layers. In 2013, Wang and Wang [20] trained the DNN as a binary classifier to estimate the ideal binary mask of noisy speech, which overcomes the computational complexity of the kernel-based machine learning method for large-scale data, improves the adaptability to unknown noise, and achieves better speech enhancement performance than the traditional method. In the work of Wang et al. [21], a more effective **ideal ration mask (IRM)** is used to replace the ideal binary mask as the training target. Experiments show that compared with other methods, the speech enhancement method based on the DNN significantly improves the quality and intelligibility of enhanced speech.

In 2015, Weninger et al. [22] used long short-term memory to estimate speech and noise features from noisy speech, and masking to remove the noise spectrum from the noisy speech spectrum to obtain clean speech. In 2015, Xu et al. [15] used the DNN to estimate clean speech from noisy speech and adopted the dropout method to alleviate the overfitting problem in the training process. This method has a good suppression effect on nonstationary noise. In 2017, Park and Lee [23] used convolutional neural networks to estimate the spectrum relationship between clean speech and noisy speech. This method achieves good performance with fewer model parameters.

Different from the masking-based training target used in the preceding methods, Xu et al. [24] put the logarithmic power spectrum of pure speech as the training target. The logarithmic power spectrum of noisy speech is taken as the training feature, and a highly nonlinear regression function is obtained by training the DNN to establish the mapping relationship between the logarithmic power spectrum of noisy speech and the logarithmic power spectrum of pure speech. When compared with traditional speech enhancement, the DNN-based algorithm tends to get significant improvements in the matter of various objective quality measures. In the work of Xu et al. [15], global variance equalization, dropout training, and noise-aware training are used. The three strategies further improve the performance of speech enhancement in a low **signal-to-noise ratio (SNR)** and nonstationary noise environment. To fully consider the phase information in speech enhancement, a complex IRM is proposed by Williamson et al. [25]. By estimating the real part

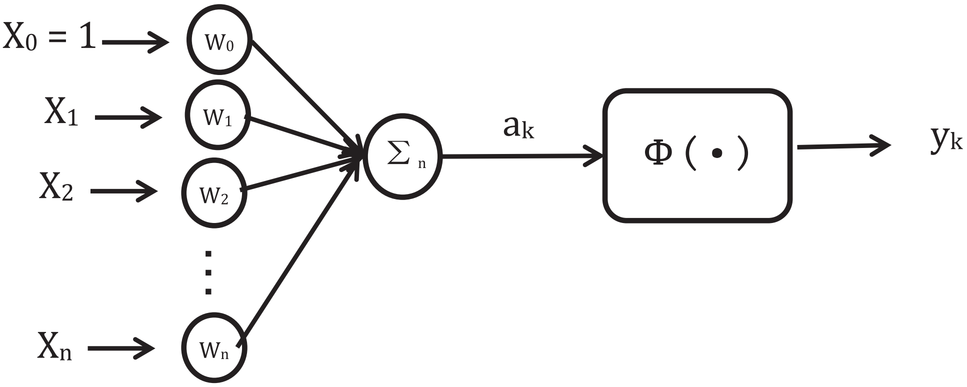

Fig. 1. Structure of the perceptron model.

and the virtual part of the masking target at the same time, the speech enhancement performance is further improved compared with other training targets.

In the work of Xu et al. [15], the speech enhancement function based on the DNN is realized by the method of multiframe input speech and multilayer DNN structure, and the final speech enhancement effect is tested.

In the research of time delay of speech enhancement based on the DNN, the traditional statistical-based speech enhancement methods generally use time domain smoothing to calculate the correlation between adjacent frames, and the smoothing process only uses the current frame (the n^{th} frame) and the preceding frame (the $(n\text{-}1)^{\text{th}}$ frame), and to reduce the delay caused by speech enhancement as much as possible to ensure the real-time performance of speech communication [26, 27]. In addition, other works [28–30] mentioned the latest technology for optimizing the depth model, which can optimize the delay of the depth neural network.

To solve the practical application of speech enhancement in hearing aids, we need to transplant the DNN to an Android app. At present, there is some research on the application of the speech enhancement algorithm in hearing aids. In the work of Yermeche et al. [31], the traditional speech enhancement algorithm is transplanted to DSP. In the work of Valin [32], a speech enhancement algorithm based on deep learning is transplanted to embedded devices. In the work of Fan et al. [33], scene classification based on the DNN is transplanted to the Android system.

3 THE DNN AND TRANSPLANTATION ALGORITHM FRAMEWORK

3.1 Perceptron Algorithm

The DNN originated from a simple perceptron algorithm, which is the basic neural unit of the artificial neural network. Its model structure is shown in Figure 1.

The algorithm is realized by software and transplanted to the Android platform. The underlying hardware can be controlled by changing the software settings, and the processing effect can be changed by software optimization.

The basic perceptron model contains multiple inputs, an adder, an activation function, and an output, and its operation expression is as follows, where x is the input eigenvector, W is the weight coefficient vector, and W_0 is the offset. The commonly used activation functions include Threshold function, Sigmoid function, Tanh function, Softmax function, and Relu function.

The selection of activation function in the output layer needs to be combined with specific application scenarios and tasks. For example, Softmax function is generally selected in

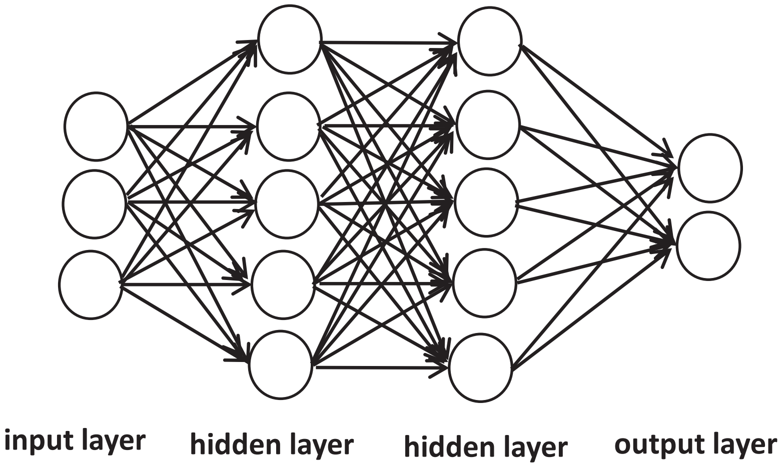

Fig. 2. Basic structure of the DNN.

multiclassification tasks, whereas Sigmoid function with continuous and smooth is generally selected in regression tasks such as speech separation.

The perceptron model is often used as a linear classifier to realize two or more classifications of machine learning according to the number of output units. However, in practical applications, the perceptron has some limitations because it requires the input characteristics to be linearly separable. For this reason, a multilayer perceptron model is proposed, which can meet the needs of nonlinear classification by adding multiple hidden layers to the perceptron.

3.2 Basic Structure of the DNN

Multilayer perceptron is also referred to as DNN, and its basic model structure is shown in Figure 2. The DNN is generally composed of three or more fully connected layers—that is, the neurons in each layer are connected with all of the neurons in the adjacent layers, whereas the neurons in the same layer are isolated and unrelated. Due to its inherent multilayer nonlinear structure, the DNN can automatically learn the deep feature representation and approach any complex nonlinear function. Therefore, with the development and popularization of deep learning technology, the DNN is widely used in speech and image processing and shows strong advantages.

3.3 Training Process of the DNN

The DNN generally uses the learning model weight coefficient of the back-propagation algorithm. However, due to the limited computing power and slow convergence speed, the early neural network algorithm does not show outstanding advantages compared with other machine learning algorithms. In 2006, an unsupervised greedy pretraining method based on the **restricted Boltzmann machine (RBM)** was proposed, which greatly improved the parameter optimization problem of the DNN and set off a research upsurge of deep learning. The method mainly consists of two stages.

First, RBM is layer-wise pretrained through forward propagation to obtain the initial parameters. RBM1 is an undirected graph model, as shown in Figure 3. It is mainly composed of two layers of neurons: visible layer (input layer) v and hidden layer (output layer) h. There is an undirected full join mode between the visible layer and the hidden layer, and there is no connection between the nodes in the same layer. Therefore, we say that RBM is "restricted" and multiple RBMs are connected in turn to form deep belief networks. v and h are random variables. It is generally

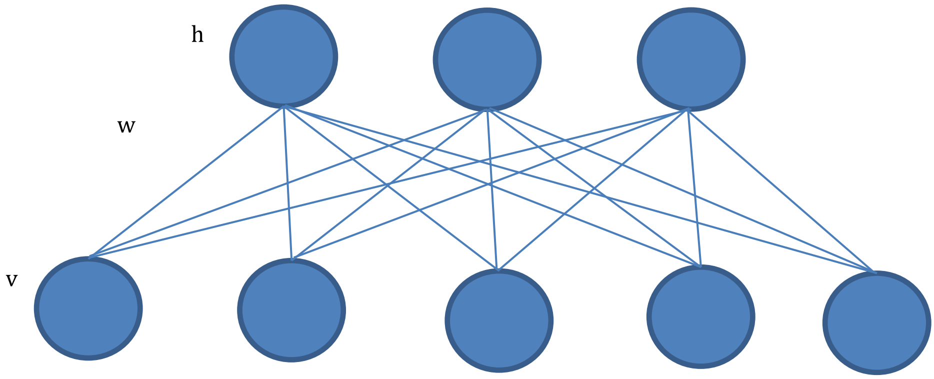

Fig. 3. Undirected graph model of the back-propagation algorithm.

assumed that v-layer nodes conform to binomial distribution, whereas the probability distribution functions commonly used in h-layer nodes are binomial distribution and Gauss distribution. The corresponding RBMs are called *Bernoulli RBM* and *Gauss Bernoulli RBM*, respectively.

Then, the back-propagation algorithm is used to conduct supervised parameter fine tuning to make the model converge effectively. The averaged squared deviation between the output of the model and the target of clean speech is first calculated according to the minimum mean square error criterion:

$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{m}\left(y^{(i)} - h_\theta\left(x^{(i)}\right)\right)^2, \tag{1}$$

where x is the input sample, y is the ideal target value of clean speech, and $h_\theta(x^{(i)})$ is the output value of network prediction. The preceding formula is sometimes called *cost function* or *objective function*. Then, the parameters are updated iteratively by the gradient descent method to solve the optimal model parameters. The weight update formula is as follows:

$$\theta_j = \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta), \tag{2}$$

where α is the learning rate. Choosing the appropriate learning rate has a great influence on the result of network training. If α is too small, the convergence speed of the network will be very slow. If α is too large, the loss function may swing around its minimum value—that is, the function will not converge.

3.4 Overall Design of Algorithm Transplantation

This topic is mainly to realize the transplantation of the speech enhancement algorithm based on the DNN on the Android platform and to verify it in the Android system environment.

The algorithm is realized by software and transplanted to the Android platform. The underlying hardware can be controlled by changing the software settings, and the processing effect can be changed by software optimization.

The entity of this work is the transplantation process of digital hearing aids, which mainly studies embedded products. Embedded product development needs a basic process, including demand analysis, detailed design of specific part structures, and the final product formation. For the algorithm transplantation of hearing aids, there are two parts: theoretical verification of

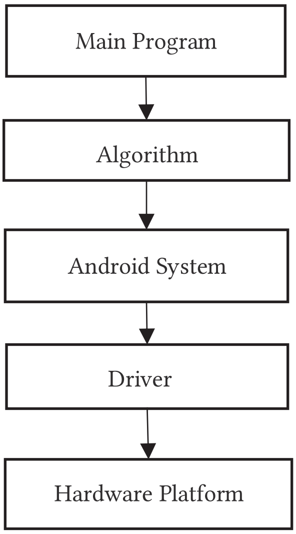

Fig. 4. Overall design pattern of algorithm migration.

algorithm accuracy and experimental verification on the platform. The specific process involved is shown in Figure 4.

The user's demand is to design an intelligent digital hearing aid for the deaf. The algorithm is mainly aimed at the algorithm of speech enhancement in the aspect of denoising and then doing an analogy optimization analysis. The algorithm is verified by different implementation methods, including simulation and actual engineering verification. The first step of the application platform mainly refers to the PC side, which makes repeated verification. Then according to the embedded development environment, it is transplanted to different platforms. At present, the mainstream embedded systems include Linux, Android, and iOS. Different platforms adopt different development methods and language specifications, but there is no need to change the algorithm itself, so there is an urgent need for a simpler way to transplant the algorithm, to solve the cross-platform continuous replication of the same functional code development. The traditional development mode is difficult to transplant in the cross-development system, as sometimes the same platform and cross level need to be copied constantly, so the algorithm migration of integrated development is a worthy research direction. The platform display is to verify the performance of the transplantation effect in specific application scenarios.

For any embedded development, we should start from the architecture of the system itself. Different embedded platforms are similar, but there are many differences. This article aims at the Android embedded development environment. For this reason, the text first validates the algorithm under the PC platform, then writes the software package under the Linux platform, and finally transplants it into the Android-embedded platform. Therefore, we need to make a deep analysis of the Android system, understand its architecture mode, and provide the basis for the software design and development.

Part of the code of algorithm transplantation is as follows:

```
public void speechEnhancement(){
 getInput();
 loadModel();
 long timeCost = System.currentTimeMillis();
```

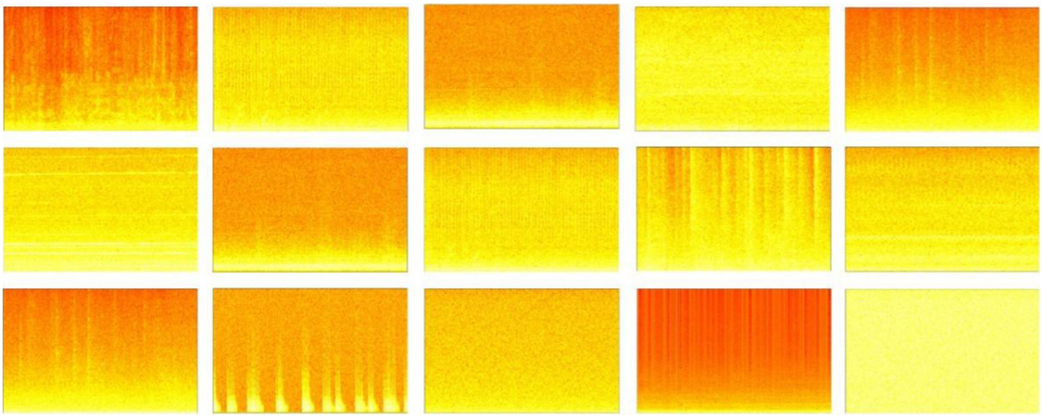

Fig. 5. The spectrum of test noises.

```
for(int i=0;i<this.inputSpectrogram.length;i++){
   tflite.run(this.inputSpectrogram[i],this.outputSpectrogram[i]);
}
timeCost = System.currentTimeMillis() - timeCost;
}
```

4 EXPERIMENTAL DESIGN

4.1 Experimental Data

4.1.1 Noise Dataset. The noise dataset is divided into *the* training dataset and *the* test dataset.

The training dataset contains 115 pieces of noise data, covering the common noises in daily life, such as crowd-speaking noise, wind noise, running water noise, machine noise, car noise, animal noise, brushing teeth, clapping hands, and coughing.

The test dataset contains 15 pieces of noise data, including white noise, interior noise, military vehicle noise, tank interior noise, restaurant noise, high-frequency channel noise, pink noise, machine gun noise, factory workshop noise 1, factory workshop noise 2, F16 fighter cockpit noise, destroyer engine room noise, destroyer combat room noise, airliner cockpit noise, and airliner Cabin noise. Figure 5 presents the spectrum of all test noises. The training dataset and the test dataset do not use the same noise data.

4.1.2 Voice Dataset. For *the DNN*, we use *the* THCHS-30 dataset for Chinese speech training.

The THCHS-30 dataset is voice data recorded in a quiet office environment. Most of the subjects were college students who could speak Mandarin fluently, and the dataset was in Chinese. The dataset is divided into the training set and the test set. The training set contains 10,000 pieces of voice data with a total time of 25.5 hours, wherease the test set contains 2,495 pieces of voice data with a total time of 6.3 hours. Figure 6 presents part of the THCHS-30 voice spectrum.

4.1.3 Generation of Training and Test Data. All voice and noise signals are uniformly compressed into a single channel, and the sampling frequency is set to 16 KHz. The frame size of voice signal is set to 512 frames, and the frame overlap rate is set to 50% (i.e., 256 frames).

In the part of training dataset, we randomly select speech from the THCHS-30 training dataset to participate in the training. Each speech is randomly mixed with two training noises, and a total of 31 hours of training data are generated for training.

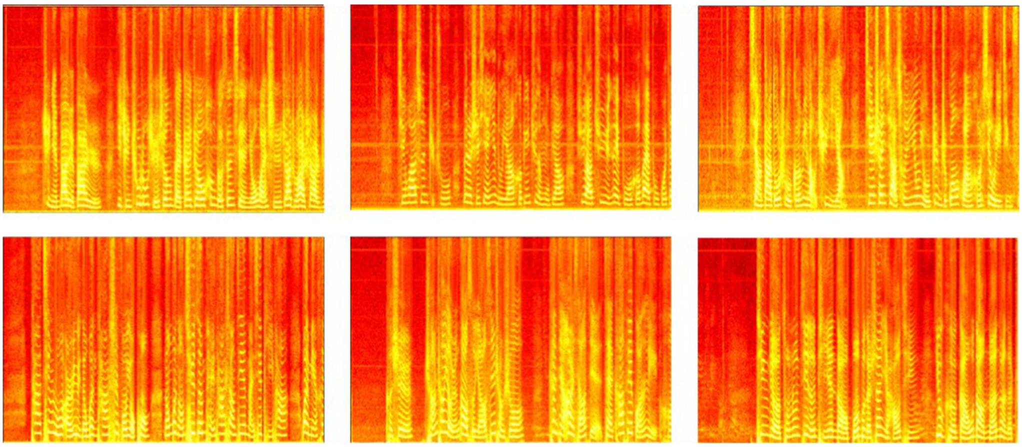

Fig. 6. Spectrum **of** the partial **THCHS-30** speech spectrum.

In the part of the test dataset, to observe the effect of different noises, we mix each voice in the THCHS-30 test dataset with all noises and generate 94.6 hours of test data.

4.2 Environment Setting

4.2.1 Experimental Environment. The neural network model training equipment is a GPU server, AMD Ryzen 7 processor, eight cores and 16 threads, 32 GB of memory, an NVIDIA RTX 2080ti video card, and 11 GB of video memory, and an Ubuntu 18.04 system is installed. The development environment includes Python 3.7 and Android Studio 3.6.

The embedded prototype system test equipment is a Huawei P30 Pro mobile phone, Kirin 980 processor, 8 GB of memory, and an Android 10.0 operating system.

4.2.2 Training and Speech Separation Test. First, the Python training and testing module on the computer successively sets the stacked frames of the input audio: 0 (i.e., only mixing does not increase efficiency), 1, 3, 5, 7, 9, 11.

In the second step, the Python training and testing module on the computer successively sets different rounds of training.

4.2.3 Android Migration Test. The following steps are taken. Upload the sound material file in the “read folder” corresponding to the hearing aid app in the mobile phone. After that, the app uses the optimal number of frames and rounds. Click the test to test the sound dataset. You can get the delay of each speech enhancement and the enhanced effect speech in the “results folder.”

5 ANALYSIS AND DISCUSSION

5.1 Impact of Stacked Frame Size on Speech Enhancement Performance

To study the effect of the number of stacked frames in the speech spectrum on the quality of speech enhancement, we set the stacked frames of the speech spectrum as 1 frame, 3 frames, 5 frames, 7 frames, 9 frames, 11 frames, and hop frame as 3 (hop3), and set the SNR as 1: 1 (0 dB), set the number of training rounds of neural network as 10,000, set the batch size as 500, set the learning rate as 0.001, and train the neural network model.

To implement the stacking of audio frames for a piece of mixed audio segment, we take 1, 3, 5, 7, 9, or 11 audio frames to form an audio block from the starting position of the audio segment, then move the starting position forward by 3 frames (if the stacked frames number is 5, 7, 9, or 11) or

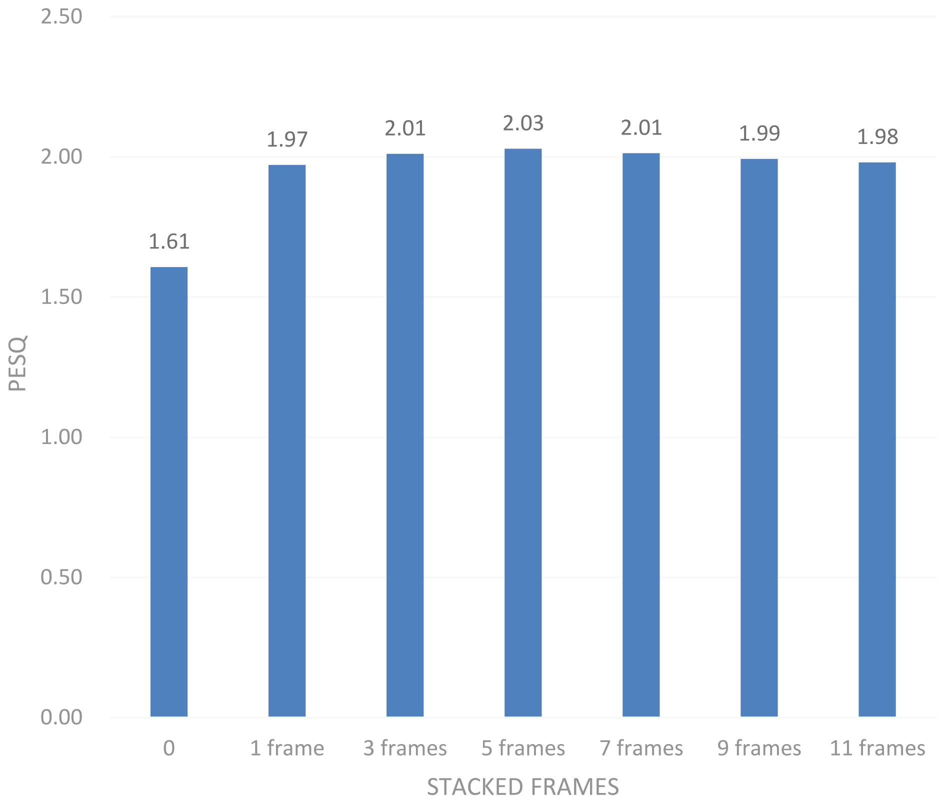

Fig. 7. Speech PESQ scores under different stacked frames.

1 frame (if the stacked frames number is 1 or 3), and then take 1, 3, 5, 7, 9, or 11 audio frames once again to form an audio block, which are repeated continuously so as to complete the operation of taking 1, 3, 5, 7, 9, or 11 frames as a block for the whole audio data finally.

We use the learning model to enhance all test speech. The PESQ score of the enhanced speech signal is shown in Figure 7.

As can be seen from Figure 7:

(1) Compared with pure speech noise without enhancement (0 frames), the PESQ scores of 1 to 11 stacked frames are significantly improved.
(2) From frame 1 to frame 5, the PESQ score of the test speech increased in turn, then the PESQ score fluctuated around 1.99.
(3) To sum up, the noise reduction performance of the model is better with 5 stacked frames.

5.2 Effect of Different Training Rounds on Speech Enhancement Performance

We set the stack frame size of speech spectrum as 7 frames, hop frame as 3 (hop3), SNR as 1:1 (0 dB), the batch size of training data as 500, and the learning rate as 0.001. We trained the neural network in 5,000 rounds, 10,000 rounds, 15,000 rounds, 20,000 rounds, and 25,000 rounds, respectively. Each model was used to enhance the test data, and the PESQ score was counted.

The training loss and test loss of the neural network model on the training dataset are shown in Figure 8.

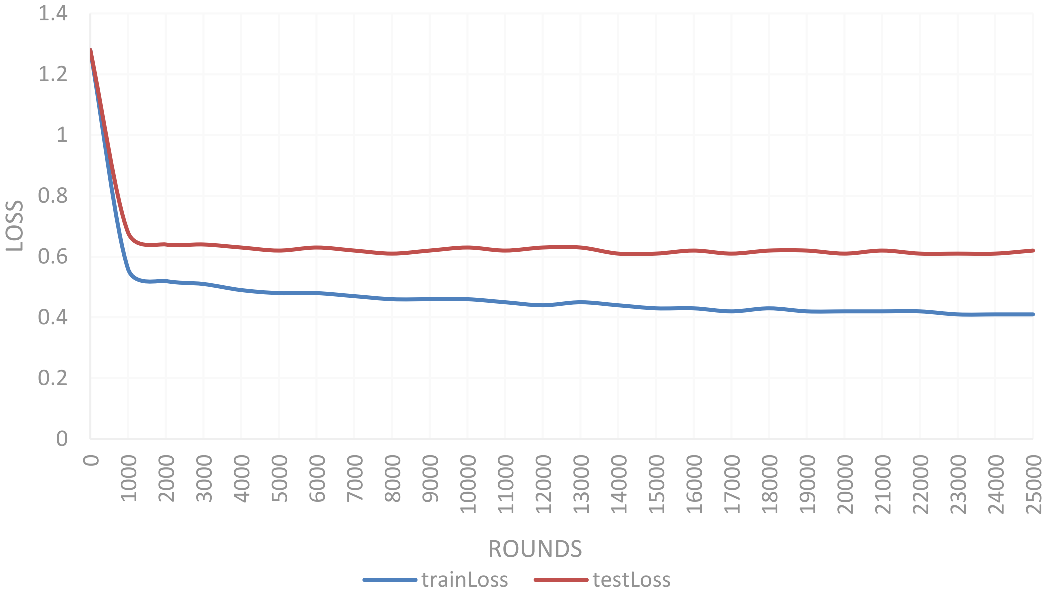

Fig. 8. Model training loss and test loss.

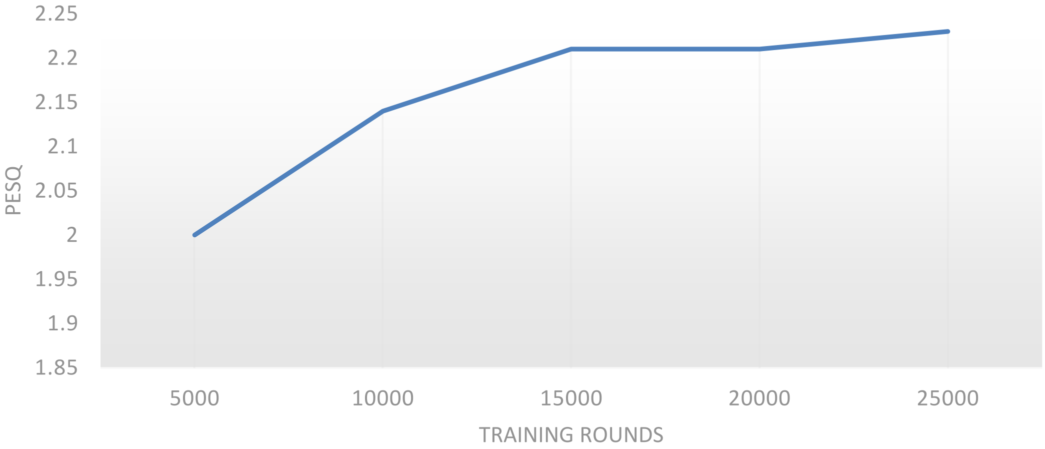

Fig. 9. Voice PESQ scores under different training rounds.

From Figure 8, we can see that for training loss, it is reduced to 0.43 at 15,000 rounds of training, and then it changes very little; for test loss, it is constantly fluctuating after it is reduced to 0.62.

To further observe the impact of different training rounds on speech enhancement performance, we use each model to conduct speech enhancement respectively for the test speech, and make statistics on the final PESQ average score, as shown in Figure 9.

From Figure 9, we can see that from 5,000 to 15,000 rounds, the PESQ score of the test voice increased significantly, then the PESQ score fluctuated around 2.2.

To sum up, after 15,000 rounds of training, the model has excellent noise reduction performance.

Table 1. Speech Enhancement Performance Under Different SNR Conditions Using the Best Model (Five Stacking Frames and 15,000 Training Rounds)

SNR	−5 dB	0 dB	5 dB	10 dB	15 dB	20 dB
PESQ (ONLY MIXED)	1.21	1.61	2.01	2.39	2.76	3.13
PESQ (five frames, 15,000 rounds)	1.68	2.07	2.44	2.77	3.08	3.33

Table 2. Impact of Stacked Frame Size on Speech Enhancement Performance

Stacked Frames (#)	Frames Enhanced (#)	Average Total Time Consumed (ms)	Average Time Consumed per Frame (ms)
1	1,877	7,362	3.92
3	1,875	7,956	4.24
5	1,873	8,498	4.54
7	1,871	9,105	4.87
9	1,869	9,587	5.13
11	1,867	10,442	5.59

5.3 Speech Enhancement Performance Test with the Best Model (Five Stacking Frames and 15,000 Training Rounds) Under Different SNR Conditions

We use the best model (five stacking frames and 15,000 training rounds) to test the performance of speech enhancement under different SNR conditions. Set skip frame to 3 (hop3), batch size to 500, and learning rate to 0.001. The trained model was used to enhance all of the test data, and the PESQ score of the enhanced speech was calculated and the average score was counted. The results in Table 1 were obtained.

5.4 Analysis of Real-Time Test Results on the Hearing Aid App

To study the impact of different stacked frame sizes on the speech enhancement performance on the hearing aid app, we use a 30-second-long recording as the test audio, and use 1-frame, 3-frame, 5-frame, 7-frame, 9-frame, and 11-frame models respectively for three times of speech enhancement, statistical average time, and, according to the number of frames, statistical consumption time of each frame of speech enhancement, and we get the results in Table 2.

It can be seen from Table 2 that the average time consumed per frame increases with the increase of the number of stacked frames. Therefore, it is necessary that the audio frames with this stacking number can achieve an ideal noise reduction effect, and the number of stacked frames should be as small as possible. For our speech enhancement prototype system, it is appropriate to select five frames as the stack frame size.

6 CONCLUSION AND FUTURE WORK

This article presents the implementation of speech enhancement based on the DNN on a hearing aid app. In this work, the THCHS-30 Chinese speech dataset and the common noise dataset in daily life are used as training and testing data of the DNN. Multiframe audio stacking and multiround neural network training are used to improve the effect of speech enhancement. The speech enhancement delay of the neural network is tested on the mobile phone hearing aid app, and the comprehensive comparison shows that the optimal stacking frame number is 5 and the optimal round number is 15,000. The experiment shows that based on the DNN, using an appropriate number of rounds for training and using an appropriate number of audio frame stacking to improve the speech enhancement effect, and transplanting this speech enhancement model to the hearing aid app, can effectively improve speech clarity and intelligibility within a reasonable time delay range.

The speech enhancement prototype system designed in this work needs support of the Android system. Although power consumption of the Android system is very low, there is still a certain gap compared with the power consumption level of hearing aids. In later research, we will try to implement voice enhancement in the embedded devices with lower power consumption. At the same time, how to apply speech enhancement on an Android system to hearing aids also needs to be studied.

REFERENCES

[1] Xiaoling Ma, Xun Liu, Sixing Zhang, Mengkang Zhang, Xiushan Cao, Liuming Tian, and Wen Gao. 2014. The development and prospect of hearing aids. *Journal of Minzu University of China (Natural Sciences Edition).*

[2] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra. 2001. Perceptual evaluation of speech quality (PESQ)—A new method for speech quality assessment of telephone networks and codecs. In *Proceedings of the 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, Los Alamitos, CA, 749–752.

[3] S. Boll. 1979. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 27, 2 (1979), 113–120.

[4] J. S. Lim and A. V. Oppenheim. 1978. All-pole modeling of degraded speech. *IEEE Transactions on Acoustics Speech, and Signal Processing* 26, 3 (1978), 197–210.

[5] J. Chen, J. Benesty, Y. Huang, and S. Doclo. 2006. New insights into the noise reduction Wiener filter. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 14, 4 (2006), 1218–1234.

[6] Y. Ephraim and D. Malah. 1984. Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 32, 6 (1984), 1109–1121.

[7] I. Cohen and B. Berdugo. 2001. Speech enhancement for nonstationary noise environments. *Signal Processing* 81, 11 (2001), 2403–2418.

[8] I. Cohen. 2003. Noise spectrum estimation in adverse environments: Improved minima controlled recursive averaging. *IEEE Transactions on Speech and Audio Processing* 11, 5 (2003), 466–475.

[9] S. I. Tamura. 1989. An analysis of a noise reduction neural network. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing.* 2001–2004.

[10] F. Xie and D. V. Compernolle. 1994. A family of MLP based nonlinear spectral estimators for noise reduction. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing.* 53–56.

[11] G. E. Hinton, S. Osindero, and Y. W. Teh. 2006. A fast learning algorithm for deep belief nets. *Neural Computation* 18, 7 (2006), 1527–1554.

[12] G. E. Hinton and R. R. Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313, 5786 (2006), 504–507.

[13] B.-Y. Xia and C.-C. Bao. 2013. Speech enhancement with weighted denoising auto-encoder. In *Proceedings of the 2013 INTERSPEECH Conference.* 3444–3448.

[14] X.-G. Lu, Y. Tsao, S. Matsuda, and C. Hori. 2013. Speech enhancement based on deep denoising autoencoder. In *Proceedings of the 2013 INTERSPEECH Conference.* 436–440.

[15] Y. Xu, J. Du, L. R. Dai, and C. H. Lee. 2015. A regression approach to speech enhancement based on deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23, 1 (2015), 7–19.

[16] H. Larochelle, Y. Bengio, J. Louradour, and P. Lamblin. 2009. Exploring strategies for training deep neural networks. *Journal of Machine Learning Research* 10 (2009), 1–40.

[17] Y. Bengio. 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning* 2, 1 (2009), 1–127.

[18] S. Tamura. 1989. An analysis of a noise reduction neural network. In *Proceedings of the 1989 International Conference on Acoustics, Speech, and Signal Processing (ICASSP'89).*

[19] G. E. Hinton, S. Osindero, and Y. W. Teh. 2006. A fast learning algorithm for deep belief nets. *Neural Computation* 18, 7 (2006), 1527–1554.

[20] Y. X. Wang and D. L. Wang. 2013. Towards scaling up classification-based speech separation. *IEEE Transactions on Audio, Speech, and Language Processing* 21, 7 (2013), 1381–1390.

[21] Y. X. Wang, A. Narayanan, and D. L. Wang. 2014. On training targets for supervised speech separation. *IEEE Transactions on Audio, Speech, and Language Processing* 22, 12 (2014), 1849–1858.

[22] F. Weninger, H. Erdogan, S. Watanabe, E. Vincent, J. Le Roux, J. R. Hershey, and B. Schuller. 2015. Speech enhancement with LSTM recurrent neural networks and its application to noise-robust ASR. In *Proceedings of the International Conference on Latent Variable Analysis and Signal Separation.* 91–99.

[23] Serim Park and Jin Lee. 2017. A fully convolutional neural network for speech enhancement. arXiv:1609.07132.

[24] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee. 2014. An experimental study on speech enhancement based on deep neural networks. *IEEE Signal Processing Letters* 21, 1 (2014), 65–68.

[25] D. S. Williamson, Y. X. Wang, and D. L. Wang. 2016. Complex ratio masking for monaural speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24, 3 (2016), 483–492.

[26] Jingxian Tu and Youshen Xia. 2018. Effective Kalman filtering algorithm for distributed multichannel speech enhancement. *Neurocomputing* 275, (2018), 144–154.

[27] R. K. Kandagatla and P. V. Subbaiah. 2018. Speech enhancement using MMSE estimation of amplitude and complex speech spectral coefficients under phase-uncertainty. *Speech Communication* 96 (2018), 10–27.

[28] Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. 2015. Deep learning with limited numerical precision. *Proceedings of Machine Learning Research* 37 (2015), 1737–1746.

[29] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally. 2016. EIE: Efficient inference engine on compressed deep neural network. *ACM SIGARCH Computer Architecture News* 44, 3 (2016), 243–254.

[30] N. D. Lane, S. Bhattacharya, Petko Georgiev, C. Forliveski, L. Jiao, L. Qendro, and F. Kawsar. 2016. DeepX: A software accelerator for low-power deep learning inference on mobile devices. In *Proceedings of the 2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN'16).*

[31] Z. Yermeche, B. Sallberg, N. Grbic, and I. Claesson. 2007. Real-time DSP implementation of a subband beamforming algorithm for dual microphone speech enhancement. In *Proceedings of the 2007 IEEE International Symposium on Circuits and Systems.* IEEE, Los Alamitos, CA.

[32] J. M. Valin. 2017. A hybrid DSP/deep learning approach to real-time full-band speech enhancement. arXiv:1709.08243.

[33] Xiaoqian Fan, Tianyi Sun, Wenzhi Chen, and Quanfang Fan. 2020. Deep neural network based environment sound classification and its implementation on hearing aid app. *Measurement* 159 (2020), 107790.

Received July 2020; revised November 2020; accepted November 2020