Learning Deep Representation of The Emotion Speech Signal

Junyi Duan[1], Zheng Song[2], Jianfeng Zhao[3*]
{duanjy@chinatowercom.cn[1], 1031761899@qq.com[2], nmgzjf@buaa.edu.cn[3*]}

Inner Mongolia Branch of China Tower Corporation Limited, 010000, Hohhot, Inner Mongolia, China [1], School of Electronic Information Engineering, Inner Mongolia University, 010021, Hohhot, Inner Mongolia, China[2], Hangzhou Innovation Institute, Beihang University, 310000, Hangzhou, China[3*]

Abstract. This paper aims at learning deep representation of emotion speech signal directly from raw audio clip using a 1D convolutional encoder, and reconstructing the audio signal using a 1D deconvolutional decoder. The learned deep features which contain the essential information of the signal, should be robust enough to reconstruct the speech signal. The location of the maximal value in the pooled receptive field of the max pooling layer is passed to the corresponding unpooling layer for reconstructing the audio clip. Residual learning is adopted to ease the training process. A dual training mechanism was developed to enable the decoder to reconstruct the speech signal from the deep representation more accurate. After completing the training of the convolutional-deconvolutional encoder-decoder as a whole, the decoder with transferred features was trained again. Experiments conducted on Berlin EmoDB and SAVEE database achieve excellent performances.

Keywords: Deep representation, emotion speech signal, 1D deconvolutional decoder, convolutional-deconvolutional encoder-decoder

1 Introduction

With the development of deep learning, there emerge a trend towards deriving a deep representation of the input signal directly from raw data. The learned deep representation of the unprocessed signal can lead to better performance in many practical tasks. The basic approach of learning deep representation is to construct a deep architecture to extract the high-level features, which can represent the input signal. So, a series of new neural network architectures have been introduced to learn deep representation, such as autoencoder network, convolutional neural network (CNN), or recurrent neural network (RNN) [1], [2], [3].

[1] Junyi Duan(1976~), male, engineer, Inner Mongolia Branch of China Tower Corporation Limited.

[2] Zheng Song(1995~), female, Master candidate, School of Electronic Information Engineering, Inner Mongolia University.

[3] Jianfeng Zhao(1978~), corresponding author, male, associate professor, Hangzhou Innovation Institute, Beihang University.

EAI GreeNets 2021, June 06-07, Dalian, People's Republic of China

DOI 10.4108/eai.6-6-2021.2307539

Due to the high learning capacity, deep representation learning has made a great improvement in several fields, especially in learning hieratical image representation. Several very deep autoencoders have been proposed to learn the deep representation of the image, and used for many tasks [4], [5], [6], [7]. The deep convolutional encoder-decoders with deep residual learning blocks (RLBs) [8], [9], [10], which allow skip connections, have also achieved great success and obtained better performance in learning deep representation from images.

As regards the research fields relevant to speech, there are a lot of researchers have devoted to complete the recognition or prediction task using different deep networks. The autoencoder, CNN, and RNN have reformed the speaker, speech, and speech emotion recognition by learning deep features from different hand-crafted features, especially from mel-frequency cepstral coefficients (MFCCs) and short-time Fourier transform (STFT) [11], [12], [13], [14]. The deep features learned by the deep neural network (DNN) can form a hierarchical representation of raw data, and perform better in some experiments, such as emotion prediction [15]. The learning of the deep representation, which composes of multiple linear and non-linear transformations, from raw audio clip also has been brought into focus.

Inspired by the success of image restoration using deep residual learning, we designed a 1D deep convolutional-deconvolutional encoder-decoder with RLBs to learn the deep representation from raw audio clip using convolution and pooling, and reconstruct the audio signal from deep representation using deconvolution and unpooling. To unpooling properly, the exact location of the maximal value [16], which is lost during the max pool, is transferred from the layer of encoder to the corresponding layer of decoder. Residual learning block (RLB) is used to ease the training process. A dual training mechanism was developed to enable the decoder to reconstruct the speech signal from the deep representation more accurate. After the completion of the training of encoder-decoder as a whole, the decoder with transferred features was trained again. The experimental results show that the dual training mechanism can make the deconvolutional decoder reconstruct the emotion speech signal from the deep representation effectively with the help of the dual training mechanism.

Our key contributions are as follows: 1) we complete the design of 1D deep convolutional-deconvolutional encoder-decoder to learn deep representation from speech signal using convolution and pooling, and restore the signal from the learned deep representation using deconvolution, and unpooling; 2) we adopt a different RLBs in encoder and decoder to ease the training of the very deep encoder-decoder; 3) we develop a dual training mechanism, which enable the encoder-decoder perform better. The dual training mechanism can enable the decoder to restore the speech signal from the deep representation more accurate. To our knowledge, it is the first framework that applies such a convolutional-deconvolutional encoder-decoder to learn deep representation from speech signal, and the results show that the designed architecture can learn the deep representation and reconstruct the speech signal with high fidelity Successfully.

2 Related Work

With the development of deep learning, deep representation learning has made great strides over the last several years, especially in image processing, such as image restoration. Hinton and Krizhevsky designed a very deep autoencoder to learn the deep representation of the image, which was used to implement the application of content-based image retrieval [4]. Pascal Vincent *et al*. proposed a stacked denoising autoencoder to learn useful representations

with a local denoising criterion, which could learn Gabor-like edge detectors from natural image patches and larger stroke detectors from digit images contrary to ordinary autoencoders [5]. Junbo Zhao *et al*. presented a stacked what-where auto-encoder, which produced the reconstruction of the image by integrating discriminative and generative pathways and providing a unified approach without relying on sampling during training [6]. Yiyi Liao *et al*. proposed a graph regularized auto-encoder, which preserved the local connectivity from the original image space to the representation space and provided explicit encoding model for fast inference and powerful expressive capacity for complex modelling [7]. When residual learning framework was present by Kaiming He *et al*. [8], [9] the deep convolutional encoder-decoder designed by Xiao-Jiao Mao *et al*. [10] with symmetric skip connections, which was composed of multiple layers of convolution and deconvolution operators, achieved better performance in recovering the original image.

Learning deep features from raw speech waveform to implement speech-related tasks has also attracted increasing attention of many researchers. In addition to the above deep representation learning approaches on image-related tasks, there were many other encouraging achievements [17], [18], [19], [20], [21] that also motivated us to achieve better performance in learning deep representation from the raw speech signal. Jaitly and Hinton proposed learning an intermediate representation by training a restricted Boltzmann machine (RBN) directly on the speech time signal for automatic speech recognition (ASR). Experiments on the TIMIT phoneme recognition task achieved state-of-the-art results at the time [22]. Bhargava and Rose presented a stacked bottleneck deep neural network, which was trained on windowed speech waveforms. The experimental results were only worse than corresponding MFCCs on the same architecture [23]. Palaz *et al*. proposed a CNN architecture, which was directly trained on the speech signal to estimate phoneme class conditional probabilities. The experimental results showed that the architecture had a more robust performance in noisy conditions [24], [25]. Sander Dieleman and Benjamin Schrauwen designed a CNN architecture to learn features directly from raw audio signals, which was successfully applied to an automatic tagging task [26].

3 The Proposed Approach

To learn compact and robust features, a deep convolutional-deconvolutional encoder-decoder was designed according to the requirements of the experiments and the hardware resources. Convolution and pooling are exactly proved to outperform on learning features, which are inspired from biological visual cortex and designed to emulate the behaviour of the visual system [29]. With the distinguishing excellences, spatially-local connectivity and shared weights [30], convolution and pooling perform the function of the learning filters in the encoder. In order to reconstruct the speech signal, deconvolution and unpooling were adopted to build the decoder for symmetry.

3.1 1D Convolution and Deconvolution

The 1D convolution layer plays the role of feature extractors and learns the local features which restrict the receptive fields of the hidden layers to be local (see Figure 1a). The convolution kernel, which has a small receptive field, is convolved across the input. The convolution will produce a features map by computing the dot product between the entries of the filter and the input. It is the core building block of a CNN.

If 1D convolution layer takes as input a speech signal $x(n)$, the feature map $z_k(n)$ can be obtained by convolving the signal $x(n)$ with the *k*-th convolution kernel $c_k(n)$ of size *l*.

$$z_k(n) = x(n) * c_k(n) = \sum_{m=-l}^{l} x(m) \cdot c_k(n-m) \quad (1)$$

To restore the speech signal *x*, the deconvolution, which reverses the effects of convolution, needs to be applied. The restored speech signal $\hat{x}(n)$ can be represent as a linear sum of feature map $z_k(n)$ convolved with filter $c_k(n)$ [37].

$$\hat{x}(n) = \sum_{k=1}^{K} z_k(n) * c_k(n) \quad (2)$$

Where K is the number of the filters.

The 1D deconvolution performs a transformation going in the opposite direction of a normal 1D convolution. To complete the transposed convolutions, the connectivity pattern of the transposed convolution operation needs to be compatible with the convolution. The input of the deconvolution is the output of the convolution, while the output of the deconvolution is the input of the convolution.

3.2 1D pooling and Unpooling

The 1D pooling layer, which makes the features robust against noise and distortion [31], performs the non-linear downsampling function and reduces the resolution of the features. The most common used non-linear functions to implement pooling is max pooling. Max pooling will partition the input into a set of non-overlapping regions and output the maximal value of each sub-region. But the max pooling will lose the exact location of the maximal value in the pooled receptive field, which is very useful for the reconstruction of the input. The features produced by max-pooling layer can be expressed as:

$$z_k^o(n) = \max_{\forall p \in \Omega_k} z_p^{o-1}(n) \quad (3)$$

Where $z_k^o(n)$ and $z_p^{o-1}(n)$ represent the maximum feature of the k-th pooling region at the o-th layer and the input features at the $(o-1)$-th layer, Ω_k represents the pooling region with index k.

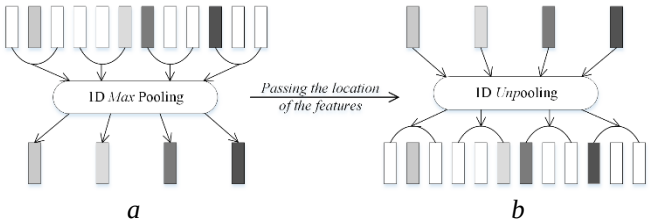


Fig. 1. Illustration of 1D max pooling and unpooling

The most common used inverse operation of 1D pooling is upsampling, which will repeat the data along the time axis (see Figure 1b). However, the upsampling function cannot correctly place the features in decoder, which will become noise and affect the quality of the signal reconstruction.

To reconstruct the signal with much higher fidelity, Goroshin *et al.* introduced a soft version of *max* and *argmax* pooling operators to hand the exact location of the maximal value from a pooling layer to the corresponding unpooling layer [38]. The *max*-pooling operator will indicate the position of the maximum value in each sub-region. Whereas the *argmax*-pooling operator will return the position of the maximum value defined over the sub-region.

Inspired by the idea of Goroshin *et al.*, we achieve this goal in our architecture using a simple method. We use hard version pooling to output the maximal value of each pooling region in encoder, and upsampling to repeat the maximal value in each unpooling region in decoder. The location of the maximal value in each pooling region in encoder will be handed to the corresponding unpooling region in decoder by a mask.

$$\underset{\Omega_k}{mask} = \partial \sum z_k^o(n) / \partial z_k^o(n) \tag{4}$$

Where $z_k^o(n)$ denotes the features of the k-th pooling region at the o-th layer, n stands for temporal location, Ω_k represents for the k-th pooling region.

The mask contains switches indicating. The value of the maximum value location of the pooling region is 1 in the mask, and the value of other location is 0. Therefore, the reconstructed signal in a unpooling region can be represented as:

$$\hat{\hat{z}}_k^o(n) = \underset{\Omega_k}{mask} \cdot \hat{z}_k^o(n) \tag{5}$$

Where $\hat{z}_k^o(n)$ denotes the upsampled feature of the k-th unpooling region at the o-th layer, $\hat{\hat{z}}_k^o(n)$ denotes the unpooled feature of the k-th unpooling region at the o-th layer, n stands for temporal location, Ω_k represents for the k-th unpooling region.

3.3 Residual Learning Block

Deep residual learning allows skip connections, which enable the network to be as linear or non-linear as the data sees fit. It was first introduced by Kaiming He *et al.* [8], [9] for image recognition. Residual learning block always consists of several layers, and can ease the training of deep neural networks (see Figure 2). The skip connections in a RLB are connections which skips one or several layers and connects to the next available layer.

Different RLBs were used in encoder and decoder to ease the training of the designed very deep encoder-decoder. The RLB adopted in encoder contains two convolution layers (see Figure 2a), and the RLB used in decoder includes two deconvolution layers (see Figure 2b). RLB can be expressed in a form:

$$x_{l+1} = F(x_l) + x_l \tag{6}$$

Where x_l and x_{l+1} are the inputted features and outputted features, and F is the residual function which is determined by the building layers. The activation function (ELU) acts as a pre-activation of the convolution and deconvolution layers, in contrast with the conventional method of post-activation.

3.4 Architecture

3.4.1 Architecture Design

The designed architecture, which is illustrated in Figure 3, consists of two parts: the encoder and decoder. The encoder is formed by a stack of convolution and pooling layers,

because of the symmetry requirement between the encoder and decoder, the decoder is built by the same number of deconvolution and unpooling layers.

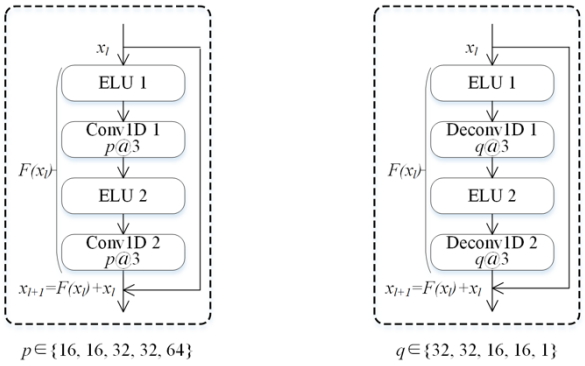


Fig. 2. Illustration of residual learning blocks

The encoder consists of five RLBs, five convolution and pooling layers. The RLB in the encoder is constructed by stacking two ELU and convolution layers with a kernel size of 3 (see Figure 2a). In the same RLB, the convolution will produce the same number of feature maps. The number of the feature maps in each RLB is 16, 16, 32, 32 and 64. The 1D convolution is applied with a kernel size of 3 on the audio data. The number output of filters in each convolution layer is 16, 16, 32, 32 and 64. That is to say, there are fifteen convolution layers in the designed encoder in total. In each pooling layer, the 2 downsampling is applied on each feature map in each RLB. This leads to the same number of feature maps with reduced temporal resolution. When the audio clip, a 64000-dimensional vector, is entered into the encoder, the dimension will be reduced to 2000.

The corresponding decoder of the designed encoder consists of five RLBs, five deconvolution and unpooling layers. The RLB in the decoder is built by linking two ELU and deconvolution layers with a kernel size of 3 (see Figure 2b). In the same RLB, the deconvolution will produce the same number of feature maps. The number of the feature maps in each RLB is 32, 32, 16, 16 and 1. The 1D deconvolution is applied with a kernel size of 3 on the audio data, and will output 32, 32, 16, 16 and 1 feature maps in each deconvolution layer. So, there are fifteen deconvolution layers in the decoder in total. In each unpooling layer, the 2 upsampling is applied on each feature map of each RLB. This also will produce the same number of feature maps with increased temporal resolution. When the deep representation, a 2000-dimensional vector, is entered into the decoder, the audio clip will be rebuilt.

3.4.2 Optimization Algorithm

The aim of hyperparameter optimization is choosing a set of hyperparameters for a deep architecture, usually with the goal of optimizing the deep network's performance on an independent data set. An optimization algorithm will find the “best fitting” model by solving a convex minimization or concave maximization problem [32].

Adaptive moment estimation (Adam) optimizer is adopted to train the network for faster backpropagation and smoother convergence [33]. We use the Adam with the learning rate of $\gamma = 0.001$ and exponential decay rates of $\beta_1 = 0.9$, $\beta_2 = 0.999$.

The objective function is mean squared error (MSE), which will minimize the loss between the recovered speech signal and the ground truth during the training of the network.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \tag{7}$$

Where $\hat{Y}_i$ is the recovered signal of the inputs, and Y_i is the speech signal corresponding to the inputs. After the network has been trained with optimized parameters, the best model will be returned.

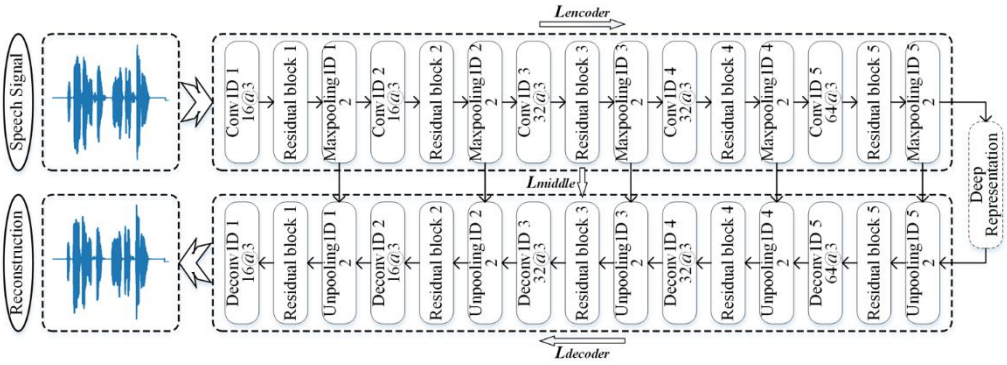


Fig. 3. The overall architecture of the designed 1D deep convolutional-deconvolutional encoder-decoder. The upper half of the architecture is an encoder, the bottom half of the architecture is a decoder. The exact location of the maximal value in the pooled receptive field is handed from the layer of encoder to the corresponding layer of decoder.

3.4.3 Regularization Methods

Overfitting is a common problem which arises with the deep architectures. Overfitting always occurs when the deep architecture is very complex, especially when the deep network has too many parameters relative to the number of observations. If a deep network has been overfit, it will have poor predictive performance. So there are many regularization methods have been proposed to overcome overfitting, such as weight decay, dropout, early stopping [34], [35], [36]. This paper used weight decay and early stopping to combat overfitting.

4. Experiments and Results

The raw data contain the whole information, so abundant features can be obtained from the audio clips. So two emotion speech databases, which contain clear features, were selected to accomplish the experiments. The audio clips were preprocessed to meet the requirements of the experiments and balance the hardware limitation against time-consuming.

4.1 Data Preprocessing

We select two emotional databases, the Berlin emotional database (Berlin EmoDB) and the Surrey Audio-Visual Expressed Emotion (SAVEE) database, both of which possess a high degree of naturalness to test and verify the designed deep convolutional-deconvolutional encoder-decoder. The two selected databases are all acted speech database, in which pre-determined sentences with the required emotions are expressed by the invited actors.

4.1.1 Databases Introduction

Berlin EmoDB - The Berlin EmoDB which was recorded in 2005 provides labeled audio clips and some analysis results. It contains seven emotions, and each emotion comprises nearly the same number of utterances to properly evaluate the classification accuracy. These acted emotional utterances were spoken by ten professional actors in an angry, boredom, disgust, fear, happy, neutral and sadness way. Five actresses and five actors feigned five short and five longer German utterances with the simulated emotions. There are 535 sentences of the utterances which come from everyday communication and can be interpretable in all applied emotions. The utterances were recorded in an anechoic chamber and were evaluated in a perception test regarding the recognisability of emotions and their naturalness [27].

SAVEE Database - The SAVEE database was released in April 2011 and provides the audio, visual, audio-visual data and other analysis data. It consists of 480 British English utterances which expressed by four native English male actors in 7 different emotions. These emotions have been described psychologically in discrete categories: anger, disgust, fear, happiness, sadness, surprise and neutral. The sentences of the emotional utterances were picked from the standard TIMIT corpus and phonetically-balanced for every emotion. The utterances were recorded, processed and labeled in a visual media lab. And ten subjects evaluated the recordings for checking the quality of data [28].

The two acted databases, which are recorded by the professional institutions, have the same number of the emotion categories. They share the following emotions: angry, disgust, fear, happy, neutral and sadness following the palette theory. But there is the boredom but without the surprise in Berlin EmoDB when compared with SAVEE database, and vice versa.

Evaluation protocol - In experiments each dataset was split into two sets randomly, one is training set which takes 80% of the clips, and the other one is testing set which takes the remaining 20% of the clips.

4.1.2 Data Preprocessing

The audio clips are processed simply to meet the requirements of learning deep representation. The audio clips of each database vary in sample rate and length. The audio files of the Berlin EmoDB consist of audio WAV files sampled at 16 kHz, and the audio clips of the SAVEE database sampled at 44.1 kHz. The sample rate of audio clips used in the experiments is 16 kHz. The audio clips of the SAVEE database are all resampled to 16 kHz. All the raw audio clips in the experiments are segmented to 4 s long. At 16 kHz sampling rate, they can be represented as a 64000-dimensional vector. The audio clips of the two selected databases are truncated or padded up to the length of 64000. If the length of the audio clips is longer than 4 s, they are truncated to the length of 64000. Otherwise, they will be padded to 4 s long. Then all the audio data are pre-processed to have zero mean and unit variance. To use the hard softmax, which is applied as a bias, all the audio data are mapped to [0, 1].

4.2 Network Pre-training

The designed 1D deep convolutional-deconvolutional encoder-decoder as a whole, which consists of the encoder and decoder, was trained firstly. During the training, the experimental data was split into two sets randomly, one is training set which takes 80% of the data from the whole dataset, and the other one is testing set which takes the remaining 20% of the data for testing the network.

Early stopping is adopted to combat the overfitting, which leads the model to lost predictive performance. When the monitored quantity stops improving after the patience, the training will stop. The training accuracy, training loss, validation accuracy or validation loss is always the quantity to be monitored in training progress. The patience is the number of epochs with no improvement of the monitored quantity. If the patience is too small, the training of the model will stop when a monitored quantity does not improve; on the contrary, the training will last for a long time. In the pre-training, the monitored quantity is validation accuracy, the patience is set to 8. This patience not only prevents the model from prematurely cancelling training, but also avoids a long training.

4.3 Network Fine-tuning

After the pre-training, the decoder of the designed 1D deep convolutional-deconvolutional encoder-decoder was fine-tuned separately to obtain better performance. The main steps of the fine-tuning are as follows: 1) save the weights of the 1D deep convolutional-deconvolutional encoder-decoder after the pre-training, 2) transfer the learned features of the designed 1D deep convolutional-deconvolutional encoder-decoder to the encoder, 3) save the prediction of the experimental data using the encoder, 4) transfer the learned features to the decoder, and fine-tune the decoder using the prediction and the experimental data. The processing is illustrated in Figure 4.

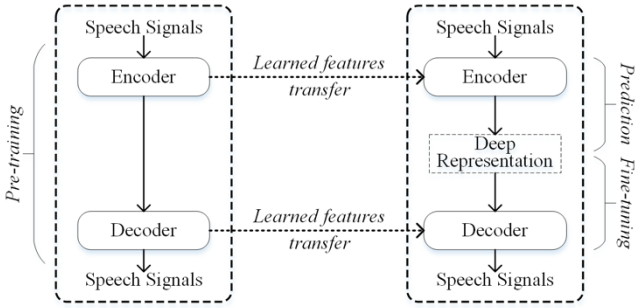


Fig. 4. Illustration of the pre-training and the fine-tuning

Early stopping is also used to combat the overfitting during the fine-tuning. The monitored quantity is validation accuracy, and the patience is also set to 8. In the experiments, the fine-tuning lasted much longer than the training.

4.4 Experimental Results

4.4.1 Experimental Results on Berlin EmoDB

The experimental results on Berlin EmoDB are shown in Table 1. The results were obtained on all the 535 audio clips, which were split into two sets randomly in the ratio of 8:2. From the table, we can see that the training loss and the testing loss in fine-tuning are much smaller than that in pre-training.

Table 1. Berlin EmoDB results for prediction of audio clips. The loss is mean squared error.

Loss	Pre-training	Fine-tuning
Training loss	0.00136	0.00077
Testing loss	0.00105	0.00080

The training and validation losses in pre-training and fine-tuning over epochs on Berlin EmoDB are illustrated in Figure 5. From the figures, we can see that the training and validation losses in fine-tuning are significantly smaller than that in pre-training.

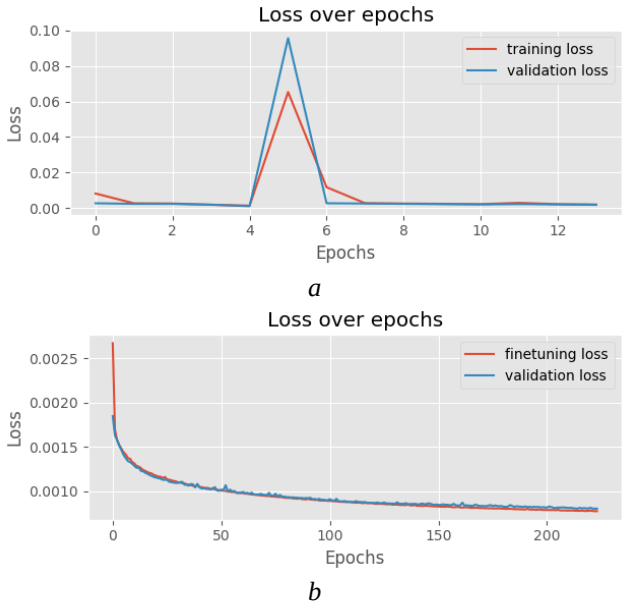


Fig. 5. The training and validation losses in pre-training and fine-tuning over epochs on Berlin EmoDB

The waveforms of the original signal, and the reconstructions produced by the pre-trained model and the fine-tuned model are illustrated in Figure 6. The first waveforms in Figure 6 are the graphical representation of the original signal. The second waveforms are restored by the pre-trained model, and the third waveforms are reconstructed by the fine-tuned model. From the figure, we can see that the waveforms of the reconstruction produced by the fine-tuned model have much higher fidelity when compared to that of the signal restored by the pre-trained model.

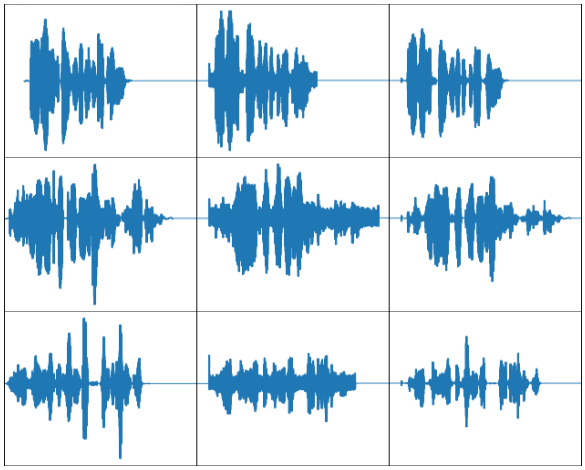


Fig. 6. The waveforms of the original signal, and the reconstructions produced by the pre-trained model and the fine-tuned model

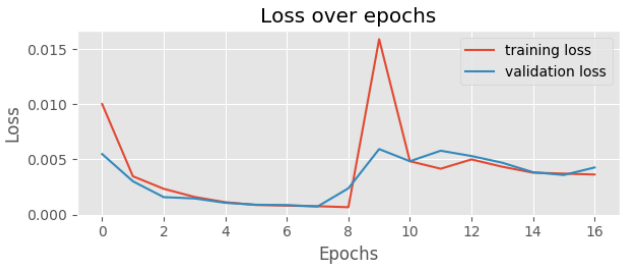
4.4.2 Experimental Results on SAVEE Database

The experimental results on SAVEE Database are shown in Table 2. The results were gained on all the 480 audio clips, which were split into two sets randomly, one is training set which takes 80% of the clips, and the other one is testing set which takes the remaining 20% of the clips. We can see that the training loss and the testing loss of the fine-tuning are much smaller than that of the pre-training.

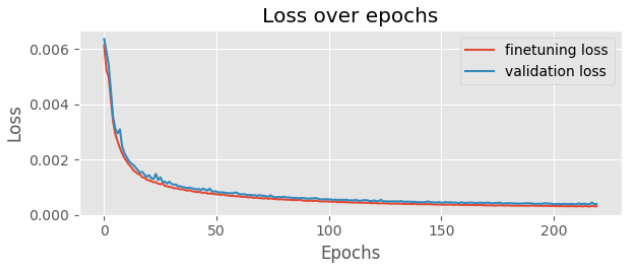
Table 2. SAVEE Database results for prediction of audio clips. The loss is mean squared error.

Loss	Pre-training	Fine-tuning
Training loss	0.00075	0.00031
Testing loss	0.00072	0.00039

The training and validation losses in pre-training and fine-tuning over epochs on SAVEE database are showed in Figure 7. These figures also show that the training and validation losses in fine-tuning are also smaller than that in pre-training. Therefore, in order to learn the more accurate representation of the signal, it is import to fine-tune the model after pre-training.



a



b

Fig. 7. The training and validation losses in pre-training and fine-tuning over epochs on SAVEE database

The waveforms of the original signal, and the reconstructions restored by the pre-trained model and the fine-tuned model are illustrated in Figure 8. The layout of the waveforms in Figure 8 is the same as that in Figure 6 We can also see that the waveforms of the signal produced by the fine-tuned model have much higher fidelity that of the reconstruction produced by the pre-trained model.

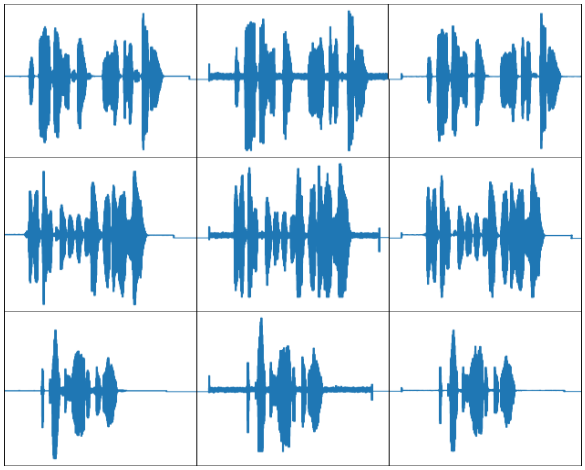


Fig. 8. The waveforms of the original signal, and the reconstructions produced by the pre-trained model and the fine-tuned model

4.5 Results Analysis

The loss of the designed 1D deep convolutional-deconvolutional encoder-decoder as a whole contains three parts:

$$L = L_{encoder} + L_{decoder} + L_{middle} \tag{8}$$

Where $L_{encoder}$ and $L_{decoder}$ are the losses of the encoder and decoder, L_{middle} is intermediate reconstruction loss.

In pre-training, the 1D convolutional-deconvolutional encoder-decoder will be trained as a whole, so the loss is composed of $L_{encoder}$, $L_{decoder}$ and L_{middle}. But in fine-tuning, the decoder is trained separately, so the loss consists only of $L_{decoder}$. The exact location of the maximal value will not be transferred from the encoder to the decoder, but the location information relating to the maximal value will be handed to the decoder by transferred features. The upsampling will not affect the quality of the signal reconstruction significantly. Therefore, the training and validation losses in fine-tuning are significantly smaller than that in pre-training in our experiments.

5. Future Work

The designed 1D deep convolutional-deconvolutional encoder-decoder has obtained better results in the experiments conducted on the Berlin EmoDB and the SAVEE database, and the waveform of the speech restoration has high fidelity. But there are many aspects worthy of further discussion and study:

1. In our experiments, only the maximal value of a pooling region was restored correctly by unpooling. So how to restore all the values of a pooling region by unpooling deserves deeply study.
2. To restore a speech signal from convoluted features accurately, the same filters adopted in convolution should be applied in deconvolution. So how to hand the information of convolutional kernels to deconvolution layer also need for further research.
3. Comparing to the distortion of the restored image, the alteration of the reconstructed speech signal is more likely to be perceived. So how to detect the distortion of the restored speech signal is worth delving into.
4. As we know, the toleration level of a person to an image is higher than that to the speech. So how to measure the distortion level also need for further research.

6. Conclusion

In this paper, we design one 1D deep convolutional-deconvolutional encoder-decoders to learn high-level representation from the raw audio clips. Further, we adopt a dual training mechanism to train the designed architecture, which can leverage the training in a more effective way. By using our proposed mechanism, the loss dropped dramatically, and the reconstruction has much higher fidelity. The results show that the designed architecture can learn deep representation from raw speech signal, and model high-level abstractions of the speech signal from the raw audio clips.

Acknowledgments. Zheng Song and Jianfeng Zhao's work in this paper was supported in part by the National Natural Science Foundation of Inner Mongolia under Grant No. 2020MS06024.

References

[1] Hinton, Geoffrey E., and Richard S. Zemel. "Autoencoders, minimum description length and Helmholtz free energy." *Neural Information Processing Systems* (1993): 3-10.

[2] Lecun, Y., et al. "Backpropagation applied to handwritten zip code recognition." *Neural Computation* 1.4 (1989): 541-551.

[3] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." *Nature* 323.6088 (1986): 533-536.

[4] Krizhevsky, Alex, and G. E. Hinton. "Using very deep autoencoders for content-based image retrieval." *Esann 2011, European Symposium on Artificial Neural Networks, Bruges, Belgium, April 27-29, 2011, Proceedings* DBLP, 2011.

[5] Vincent, Pascal, et al. "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion." *Journal of Machine Learning Research* 11.12(2010):3371-3408.

[6] Zhao, Junbo, et al. "Stacked What-Where Auto-encoders." *Computer Science* (2016).

[7] Liao, Y., Y. Wang, and Y. Liu. "Graph Regularized Auto-Encoders for Image Representation. " *IEEE Transactions on Image Processing A Publication of the IEEE Signal Processing Society* PP.99(2016):1-1.

[8] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[9] He, Kaiming, et al. "Identity mappings in deep residual networks." *European Conference on Computer Vision*. Springer International Publishing, 2016.

[10] Mao, Xiaojiao, Chunhua Shen, and Yu-Bin Yang. "Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections." *Advances in Neural Information Processing Systems*. 2016.

[11] Deng, Li, et al. "Binary coding of speech spectrograms using a deep auto-encoder." *Interspeech*. 2010.

[12] Graves, A., N. Jaitly, and A. R. Mohamed. "Hybrid speech recognition with Deep Bidirectional LSTM." *Automatic Speech Recognition and Understanding* 2013:273-278.

[13] Han, Kun, D. Yu, and I. Tashev. "Speech Emotion Recognition Using Deep Neural Network and Extreme Learning Machine." *Interspeech* 2014.

[14] Huang, Zhengwei, et al. "Speech emotion recognition using CNN." *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014.

[15] Trigeorgis, George, et al. "Adieu features? End-to-end speech emotion recognition using a deep convolutional recurrent network." *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016.

[16] Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." *European conference on computer vision*. Springer International Publishing, 2014.

[17] Shen, F., et al. "Hashing on nonlinear manifolds." *IEEE Transactions on Image Processing a Publication of the IEEE Signal Processing Society* 24.6(2015):1839-51.

[18] Yao, Yazhou, et al. "Exploiting Web Images for Dataset Construction: A Domain Robust Approach." *IEEE Transactions on Multimedia* PP.99(2016):1-1.

[19] Huang, Chen, et al. "Learning Deep Representation for Imbalanced Classification." *IEEE Conference on Computer Vision and Pattern Recognition* IEEE, 2016:5375-5384.

[20] Shen, Fumin, et al. "A Fast Optimization Method for General Binary Code Learning." *IEEE Transactions on Image Processing* 25.12(2016):5610-5621.

[21] Yao, Yazhou, et al. "A Domain Robust Approach for Image Dataset Construction." *ACM on Multimedia Conference* ACM, 2016:212-216.

[22] Jaitly, Navdeep, and Geoffrey Hinton. "Learning a better representation of speech soundwaves using restricted boltzmann machines." *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011.

[23] Bhargava, Mayank, and Richard Rose. "Architectures for deep neural network based acoustic models defined over windowed speech waveforms." *Interspeech*. 2015.

[24] Palaz, Dimitri, Ronan Collobert, and Mathew Magimai Doss. "Estimating phoneme class conditional probabilities from raw speech signal using convolutional neural networks." *arXiv preprint arXiv:1304.1018* (2013).

[25] Palaz, Dimitri, Mathew Magimai Doss, and Ronan Collobert. "Analysis of CNN-based Speech Recognition System using Raw Speech as Input." *conference of the international speech communication association* (2015).

[26] Dieleman, Sander, and B. Schrauwen. "End-to-end learning for music audio." *IEEE International Conference on Acoustics, Speech and Signal Processing* IEEE, 2014:6964-6968.

[27] Burkhardt, Felix, et al. "A database of German emotional speech." *Interspeech 2005 - Eurospeech, European Conference on Speech Communication and Technology*, Lisbon, Portugal, September 2005:1517-1520.

[28] Jackson, Philip J. B., and S. U. Haq. "Surrey Audio-Visual Expressed Emotion (SAVEE) database." (2011).

[29] Boser, Bernhard E., et al. "An analog neural network processor and its application to high-speed character recognition." *Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on*. Vol. 1. IEEE, 1991.

[30] Behnke, Sven. "Discovering hierarchical speech features using convolutional non-negative matrix factorization." *Neural Networks, 2003. Proceedings of the International Joint Conference on*. Vol. 4. IEEE, 2003.

[31] Shen, Fumin, et al. "Face image classification by pooling raw features." *Pattern Recognition* 54(2016):94-103.

[32] Shen, F., et al. "Approximate least trimmed sum of squares fitting and applications in image analysis." *IEEE Transactions on Image Processing* 22.5(2013):1836-1847.

[33] Kingma, Diederik, and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).

[34] Krogh, A. "A Simple Weight Decay Can Improve Generalization." *Advances In Neural Information Processing Systems* 4 1992:950--957.

[35] Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research* 15.1(2014):1929-1958.

[36] Loughrey, John, and P. Cunningham. "Using Early Stopping to Reduce Overfltting in Wrapper-Based Feature Weighting." *Trinity College Dublin Department of Computer Science* (2005).

[37] Zeiler, Matthew D., et al. "Deconvolutional networks." *Computer vision and pattern recognition* (2010).

[38] Goroshin, Ross, Michael Mathieu, and Yann Lecun. "Learning to linearize under uncertainty." *Neural information processing systems* (2015): 1234-1242.