
WEVE: Weighted Enhancement Using Variance Estimation

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 This paper presents a new robust neural frontend layer referred to as Power Nor-
2 malized Neural Frontend (PNNF) for robust speech recognition. PNNF is im-
3 plemented as a layer in an end-to-end all neural-network speech recognition sys-
4 tem. For speech recognition, features such as Mel Filterbank Cepstral Coeffi-
5 cient (MFCC) or log-Mel have been used very widely. More recently, features
6 motivated by human auditory processing have shown improvement under certain
7 conditions. Nevertheless, these auditory features are constructed using signal pro-
8 cessing blocks with hand-crafted parameters for certain conditions. Therefore, the
9 performance is not always optimal for various kind of acoustic conditions. PNNF
10 frontend layer is loosely motivated by human auditory system. This processing
11 uses the concept of the auditory masking and the auditory nonlinearity. However,
12 instead of directly using theories and parameters about human auditory masking,
13 neural networks make decision about it from the training data. Even though the
14 structure of the PNNF layer is very simple, this layer significantly boosts the per-
15 formance of the speech recognition system.

16 1 Introduction

17 2 Feature enhancement

18 In the feature enhancement process, we estimate a clean feature \vec{y} given a corrupt feature \vec{x} .

19 In this case, the following L_2 norm is widely used:

$$l(\vec{y}, \vec{x}) = \quad (1)$$

20 Feature mapping is a process of mapping noisy feature vectors to clean feature vectors to enhance
21 robustness [6, 9, 10]. This procedure may be formulated by the following equation:

$$Y() = X() \quad (2)$$

22 From noisy features, we may infer Another

23 Masking and feature mapping are closely related techniques. In feature mapping, enhanced features
24 are predicted from noisy features using machine learning models. The masking-based approaches
25 with Joint Adaptive Training (JAT) have been shown to be effective to a large training set [2].

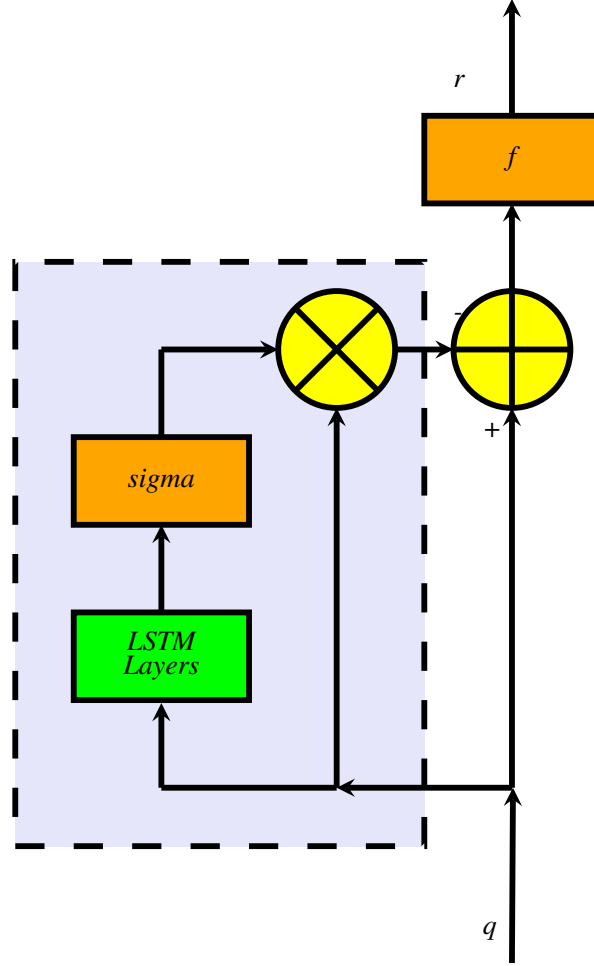


Figure 1: The structure of the Neural Background Removal (NBR) block.

3 Structure of the neural front-end

3.1 Neural nonlinearity

3.2 Neural background removal

In this section, we describe our neural background removal approach.

3.3

NIPS requires electronic submissions. The electronic submission site is

<https://cmt.research.microsoft.com/NIPS2017/>

Please read carefully the instructions below and follow them faithfully.

4 The structure of the SPL algorithm

Fig. shows the block diagram for enhancing the noisy feature.

Fig. 4 shows the structure for enhancing noisy feature.

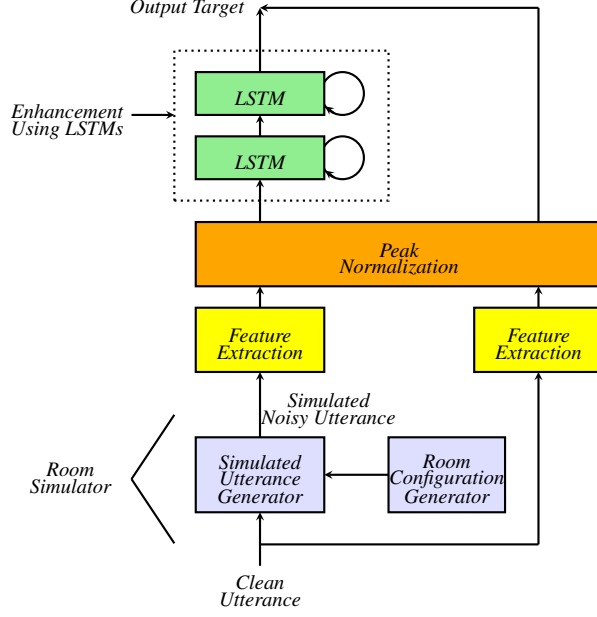


Figure 2: The structure of feature enhancement using peak value normalization and LSTMs.

4.1 Simulated utterance generation

We use the room simulation system introduced in [5] to generate noisy speech from clean speech.

$$y[n] \quad (3)$$

For

4.2 Review of the Room Simulator for Data Augmentation

In this section, we briefly review the structure of the Google Room Simulator for generating simulated utterances to train acoustic models for speech recognition systems [5]. We assume a room of a rectangular cuboid-shape as shown in Fig. 3a. Assuming that all the walls of a room reflect acoustically uniformly, we use the image method to model the Room Impulse Responses (RIRs) [3, 8, 13].

Fig. 3b shows how virtual sources appearing as empty circles are constructed in the image method. Even though the virtual rooms appear to be created in two-dimensions in Fig. 3b, in the “Room Simulator”, they are constructed in three-dimensions. For example, in our work for training the acoustic model for Google Home, we consider $17 \times 17 \times 17 = 4913$ virtual rooms for RIR calculation [5]. Following the image method, the impulse response is calculated using the following equation [3, 8]:

$$h[n] = \sum_{i=0}^{I-1} \frac{r^{g_i}}{d_i} \delta \left[n - \left\lceil \frac{d_i f_s}{c_0} \right\rceil \right], \quad (4)$$

where i is the index of each virtual sound source, and d_i is the distance from that sound source to the microphone, r is the reflection coefficient of the wall, g_i is the number of the reflections to that sound source, f_s is the sampling rate of the RIR, and c_0 is the speed of sound in the air. We use the value of $f_s = 16,000 \text{ Hz}$ and $c_0 = 343 \text{ m/s}$ for the room simulator [4]. For d_i , r , we use numbers created by a random number generator following specified distributions for each impulse response [5].

Assuming that there are I sound sources including one target source and J microphones, the received signal at microphone j is given by:

$$y_j[n] = \sum_{i=0}^{I-1} \alpha_{ij} (h_{ij}[n] * x_i[n]). \quad (5)$$

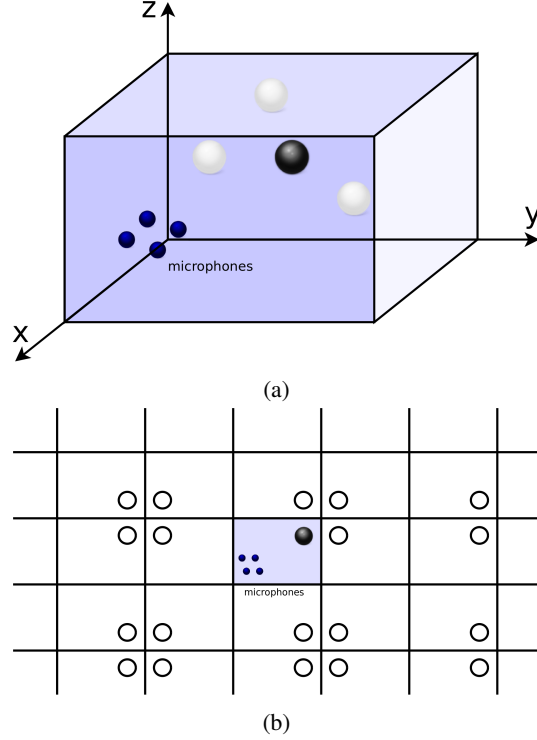


Figure 3: (a) A simulated room: There may be multiple microphones, a single target sound source, multiple noise sources in a cuboid-shape room with acoustically reflective walls. (b) A diagram showing the location of the real and the virtual sound sources assuming that the walls reflects acoustic wave like a mirror [5]. Since we used a two-microphones system in [4, 5], J is two, and for the number noise sources, we used a value from zero up to three [5].

4.3 Feature extraction

Even though the log-mel coefficients have been widely used as features for speech recognition [12], the log non-linearity has a disadvantage in that the value diverges to negative infinity as the mel filterbank coefficient approaches zero [1]. Thus, we use the power-law nonlinearity rather than the log-nonlinearity:

$$q[m, l] = (p[m, l])^{\frac{1}{15}} \quad (6)$$

We use the power coefficient of $\frac{1}{15}$ as in [1, 7].

4.4 Feature normalization using the peak-value

The power-mel coefficients $q_x[m, l]$ and $q_y[m, l]$ are obtained from the clean and the simulated noisy speech $x[n]$ and $y[n]$ respectively. $q_{x, \text{peak}}$ and $q_{y, \text{peak}}$

5 Estimation of the expected value and the variance of the enhanced feature from the noisy feature

In this section, we discuss how to predict the clean feature and the error ratio from the corrupt feature. Let us denote the clean target feature, the corrupt feature by \mathbf{x} , \mathbf{y} respectively.

Using the room simulation system described in [5], we create a pair of clean and corrupt utterances. Log-mel [XX] features are calculated from these original clean and corrupt utterances. Let us denote the clean feature by \mathbf{t}_i and the simulated corrupt feature by \mathbf{x}_i , respectively where i is the utterance index. The training set is represented by the following set:

$$\mathcal{T} = \{ \langle \mathbf{x}_i, \mathbf{t}_i \rangle \mid 0 \leq i \leq N - 1 \} \quad (7)$$

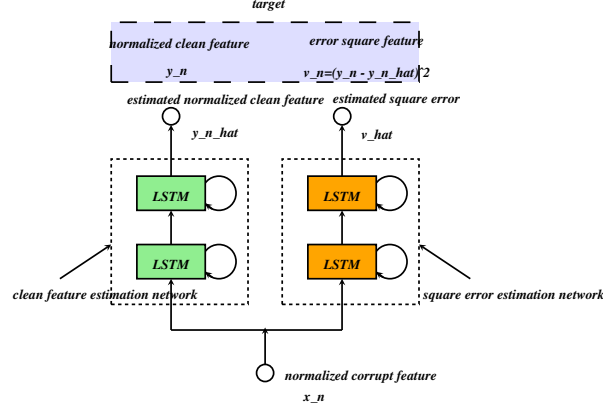


Figure 4: Training of neural networks to estimate the clean feature and error ratio from corrupt feature

79 where N is the number of training examples.

80 In the ECE algorithm, we first estimate the true target t_i . The estimation error is defined by the
 81 following equation:

$$e_i = t_i - x_i, \quad 0 \leq i \leq N - 1 \quad (8)$$

82 The norm of the error vector e_i would be generally large when the norm of the estimated clean
 83 feature is large.

84 Thus, instead of trying to estimate the error itself, the neural network tries to estimate the error ratio
 85 given as follows:

$$r_i = \quad (9)$$

86 6 Generation of simulated clean and noisy data sets

87 To train neural networks described in section 5, we need a training set consisting of pairs of clean
 88 speech and noisy speech. Unfortunately, it is not easy to have such a training set from real speech
 89 utterances. Thus, we use the simulation system described in [5] to synthetically generate noisy
 90 speech utterances from the clean speech utterances.

91 In our application, for clean training set, we used Wall Street Journal (WSJ) si-284. For noise
 92 set, we used the Resource Management 1 (RM1) [11] 50 % of time and DEMAND noise
 93 [TODO(chanwcom) Adds reference] for the remaining 50 % of time.

94 TODO(chanwcom) Cite DEMAND noise set.

95 7 Estimation of the optimal weight in the EFW algorithm

96 In this section, we describe the procedure of feature weighting approach. Suppose that the feature
 97 before enhancement is represented by \vec{x} . Let us represent the inference neural network to estimate
 98 the target by \mathcal{T} .

$$\mathcal{X} = \{\} \quad (10)$$

99 Instead of directly using the enhanced feature $\hat{\vec{y}}$, we consider the following interpolated feature \vec{z}
 100 from the original corrupt input \vec{x} and .

$$\vec{z} = \vec{w} \odot \hat{\vec{y}} + (1 - \vec{w}) \odot \vec{x} \quad (11)$$

101 where \odot denotes the Hadamard product (entry-wise product).

102 Let us denote the estimated variance vector by \hat{v} :

$$\begin{aligned}\hat{v} &= \text{Var}[\hat{y}] \\ &= E[(y - \bar{y})^2 | \vec{x}[0], \vec{x}[1], \dots, \vec{x}[T]]\end{aligned}\quad (12)$$

103 In our discussion, let us assume that the expectation of \hat{y} is the same as the true target \vec{y} .

$$E[\hat{y}] = \vec{y} \quad (13)$$

104 Now, let us obtain the expected value of \vec{z} from (11) and (13)

$$E[\vec{z}] = (1 - \vec{w}) \odot (\vec{x} - \vec{y}) \quad (14)$$

105 Thus, the bias of \vec{z} is given by:

$$\begin{aligned}\text{Bias}_{\vec{z}} &= E[\vec{z}] - \vec{z} \\ &= \vec{w} \odot (\hat{y} - \vec{y})\end{aligned}\quad (15)$$

106 From (11) and (12), the variance of \vec{z} is given by:

$$\text{Var}_{\vec{z}} = \vec{v} \odot w \odot w. \quad (16)$$

107 The mean squared error of the i -th element of the \vec{z} is given by:

$$\begin{aligned}MSE_{\vec{z}_i} &= \text{Bias}_{\vec{z}_i}^2 + \text{Var}_{\vec{z}_i} \\ &= [(\vec{x}_i - \vec{y}_i)^2 + \vec{v}_i^2] \vec{w}_i^2 - 2(\vec{x}_i - \vec{y}_i)^2 \vec{w}_i + (\vec{x}_i - \vec{y}_i)^2\end{aligned}\quad (17)$$

108 Since the above equation (17) is a quadratic equation with respect to \vec{w}_i , we may find that the
109 minimum value of $MSE_{\vec{z}_i}$ is obtained when \vec{w}_i has the following value:

$$\vec{w}_i = \frac{(\vec{x}_i - \vec{y}_i)^2}{(\vec{x}_i - \vec{y}_i)^2 + \vec{v}_i} \quad (18)$$

110 Thus, the final form of the estimated vector $\hat{\vec{z}}$ becomes:

$$\hat{\vec{z}} = \hat{\vec{w}} \odot \hat{y} + (1 - \hat{\vec{w}}) \odot \vec{x} \quad (19)$$

111 where

$$\hat{\vec{w}} = (\vec{x} - \vec{y})^{\odot 2} [(\vec{x} - \vec{y})^{\odot 2} + \vec{v}]^{\odot -1}. \quad (20)$$

112 Sequence enhancement

113 The feature enhancement is to map a sequence of corrupt features \mathbf{X} .

$$\mathbf{X} = (\vec{x}[0], \vec{x}[1], \dots, \vec{x}[M-1]) \quad (21a)$$

$$\mathbf{Y} = (\vec{y}[0], \vec{y}[1], \dots, \vec{y}[M-1]) \quad (21b)$$

114 **Acknowledgments**

115 **References**

116 References follow the acknowledgments. Use unnumbered first-level heading for the references.
117 Any choice of citation style is acceptable as long as you are consistent. It is permissible to reduce
118 the font size to small (9 point) when listing the references. **Remember that you can go over 8**
119 **pages as long as the subsequent ones contain only cited references.**

References

- [1] C. Kim and R. M. Stern. Power-Normalized Cepstral Coefficients (PNCC) for Robust Speech Recognition. *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, pages 1315–1329, July 2016.
- [2] A. Narayanan. Large-scale, sequence-discriminative, joint adaptive training for masking-based robust asr. In *INTERSPEECH-2015*, pages 3571–3575, Sept. 2015.
- [3] J. Allen and D. Berkley. Image method for efficiently simulating small-room acoustics. *J. Acoust. Soc. Am.*, 65(4):943–950, April 1979.
- [4] B. Li, T. Sainath, A. Narayanan, J. Caroselli, M. Bacchiani, A. Misra, I. Shafran, H. Sak, G. Pundak, K. Chin, K-C Sim, R. Weiss, K. Wilson, E. Variiani, C. Kim, O. Siohan, M. Weintraub, E. McDermott, R. Rose, and M. Shannon. Acoustic modeling for Google Home. In *INTERSPEECH-2017*, pages 399–403, Aug. 2017.
- [5] C. Kim, A. Misra, K.K. Chin, T. Hughes, A. Narayanan, T. Sainath, and M. Bacchiani. Generation of simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google Home. In *INTERSPEECH-2017*, pages 379–383, Aug. 2017.
- [6] C. Kim and R. M. Stern. Power function-based power distribution normalization algorithm for robust speech recognition. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pages 188–193, Dec. 2009.
- [7] C. Kim and R. M. Stern. Power-normalized cepstral coefficients (pncc) for robust speech recognition. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, pages 4101–4104, March 2012.
- [8] E. A. Lehmann, A. M. Johansson, and S. Nordholm. Reverberation-time prediction method for room impulse responses simulated with the image-source model. In *2007 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, pages 159–162, Oct. 2007.
- [9] D. Wang W. S. Woods I. Merks K. Han, Y. Wang and T. Zhang. Learning spectral mapping for speech dereverberation and denoising. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(6):982–992, June 2015.
- [10] Y. Wang K. Han and D. Wang. Learning spectral mapping for speech dereverberation. In *IEEE Int. Conf. Acoust., Speech, and Signal Processing*, pages 4628–4632, May 2014.
- [11] Price, P, et al. *Resource Management RM1 2.0 LDC93S3B. DVD*. Linguistic Data Consortium, Philadelphia, PA, 1993.
- [12] S. B. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Trans. Acoust., Speech, and Signal Processing*, 28(4):357–366, Aug. 1980.
- [13] S. G. McGovern. a model for room acoustics.