INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Concatenative Resynthesis using twin networks

Soumi Maiti[1], *Michael I Mandel*[1,2]

[1]The Graduate Center, CUNY, USA
[2]Brooklyn College, CUNY, USA

smaiti@gradcenter.cuny.edu, mim@sci.brooklyn.cuny.edu

Abstract

Traditional noise reduction systems modify a noisy signal to make it more like the original clean signal. For speech, these methods suffer from two main problems: under-suppression of noise and over-suppression of target speech. Instead, synthesizing clean speech based on the noisy signal could produce outputs that are both noise-free and high quality. Our previous work introduced such a system using concatenative synthesis, but it required processing the clean speech at run time, which was slow and not scalable. In order to make such a system scalable, we propose here learning a similarity metric using two separate networks, one network processing the clean segments offline and another processing the noisy segments at run time. This system incorporates a ranking loss to optimize for the retrieval of appropriate clean speech segments. This model is compared against our original on the CHiME2-GRID corpus, measuring ranking performance and subjective listening tests of resyntheses.
Index Terms: noise suppression, siamese network

1. Introduction

Noise is very disruptive to speech communication technologies, such as hearing aids, automatic speech recognition, and mobile communication. State of the art single-channel noise suppression techniques (e.g. [1]) modify the noisy signal to make it more like the original clean signal. In doing so, they both include some noise in the output and reduce the speech quality. In contrast, we use here the recently introduced concatenative resynthesis approach [2, 3], which replaces segments of noisy speech with the best-matching segments of clean speech. Such an approach can theoretically remove all of the noise and generate high quality speech signals.

The key component of the concatenative resynthesis system is a learned similarity metric between noisy and clean speech segments. Proposed in our previous work [2], this similarity metric is a deep neural network that takes as input both a clean and a noisy segment and outputs a scalar similarity score. The similarity network processes all pairs of clean and noisy segments at run time, making its runtime increase linearly with the size of the dictionary. This performance makes it difficult to scale the system to large dictionaries of clean speech which are required for large-vocabulary tasks. Here, we propose splitting the similarity function into two "twin" networks, one that processes the clean speech and another that processes the noisy speech. Both networks output embedding codes into a shared low-dimensional embedding space, which is learned so that matching clean and noisy segments are assigned similar embeddings. The clean segments can then be preprocessed offline and only the noisy segments would need to be processed by the network at runtime, with an efficient search of the embedding space performed using an approximate nearest neighbor algorithm, e.g.,[4, 5].

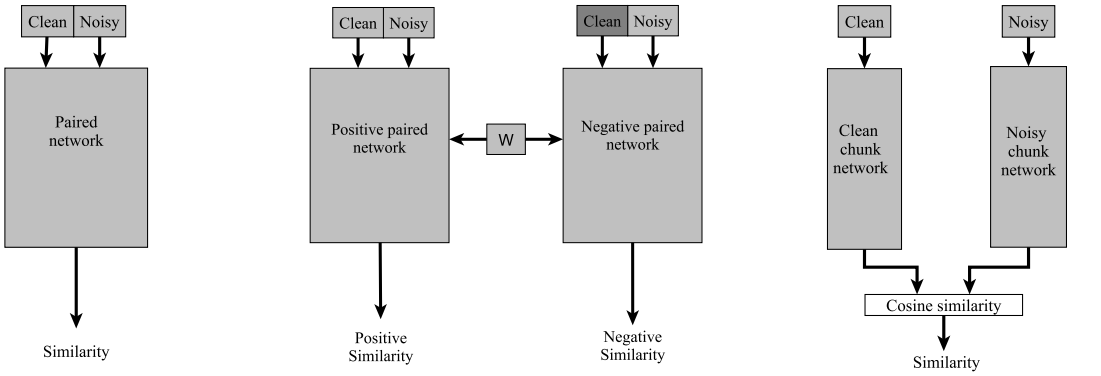
In this paper we examine the effect of splitting the original similarity network in this way. First, we utilize a siamese architecture[6] to duplicate our previous network[2] and incorporate a ranking loss. The new ranking similarity model fetches the correct dictionary element 97% of the time when it is present and provides better performance that our previous system in intelligibility and comparable results in speech quality. Second, we split the similarity network into two twin networks, one processing clean and another processing noisy speech segments. The twin network achieves higher speech quality with a minor loss in intelligibility, but reduces the runtime of the most expensive part of the code by several orders of magnitude, potentially enabling real-time and large-vocabulary applications.

2. Related Work

The siamese network architecture was first introduced for signature verification in 1993[6]. Several siamese deep neural networks have been used for face recognition/ face verification applications [7, 8, 9, 10] in recent years. The name siamese network is inspired by siamese twins or conjoined twins. A siamese network consists of two identical sub-networks with tied weights that are conjoined in their loss function [6]. The network is trained using pairs of inputs that should be recognized as matching and pairs that should not, but avoids the need for a direct supervision signal dictating the desired output of each network individually. Without pre-defining an output class, this architecture can be used to identify matching instances of classes that were not seen at training time. The original work [6] applied it to discriminating real versus forged signatures. A recent approach using a similar idea in audio is Deep Clustering [11], which learns embeddings of neighborhoods of spectrogram points for source separation without needing to pre-define the number of sources. Previous uses of siamese networks have typically treated the two paths symmetrically, assuming that the same type of input is being processed by networks with the same weights. Here, instead, the two inputs are of qualitatively different character, one clean and the other noisy, and so must be processed by two different networks. It is just the joint embedding space that they share.

To learn the relative similarity of images using genuine and impostor examples, some authors use pairwise ranking [12, 13]. In [13], the authors implement a triplet DNN network with ranking loss, defining a loss that counts violations in relative ranking. Hence by minimizing this loss, the model learns the expected ranking order. Hadsell et al[14] proposed a contrastive loss function to reduce dimensionality for set of similar and dissimilar points so that similar points are mapped nearer in a manifold. This contrastive loss was also used successfully for image recognition siamese networks [9]. We utilize both of these losses with our model to improve performance.

http://dx.doi.org/10.21437/Interspeech.2017-1653



(a) *Paired-input network* (b) *Ranking paired-input network* (c) *Twin networks*

Figure 1: *Different network structures of similarity functions. In (b) the two networks share weights W, both networks received the same noisy input, for which the unshaded clean input matches and the shaded clean input does not.*

3. Technical Overview

Noisy and clean utterances are divided into temporally overlapped smaller chunks. From the clean utterances we build a clean chunk dictionary and from the noisy mixtures we build a noisy observation chunk set. Let us assume we have a clean chunk dictionary $\{z_j\}_{j=1}^{J}$ and an observed noisy chunk set $\{x_i\}_{i=1}^{I}$. We also assume each noisy chunk is constructed by combining one clean dictionary chunk with noise. Each of these chunks consists of a 192 ms of audio or 11 spectrogram frames at a hop size of 16 ms.

3.1. Training dataset construction

If x_i is created by adding noise to z_j, then define (z_j, x_i) as a matching clean-noisy pair, if it is created from a different clean chunk, then this is a non-matching pair. Matching and non-matching clean-noisy pairs are also referred to as positive and negative pairs in this paper.

To construct the training dataset, for each noisy chunk, x_i, we select the corresponding clean chunk, z_i^+, as a matching pair (z_i^+, x_i). Then we randomly select any clean dictionary chunk excluding the correct one, $z_i^- \in \{z_j\}_{j=1}^{J} \setminus z_i^+$, as a non matching pair (z_i^-, x_i). Matching and non-matching samples are defined as,

$$\begin{aligned} &\{(z_i^+, x_i)\}_{i=1}^{I} && \text{s.t. } x_i \text{ is created from } z_i^+ \\ &\{(z_i^-, x_i)\}_{i=1}^{I} && \text{s.t. } z_i^- \in \{z_j\}_{j=1}^{J} \setminus z_i^+. \end{aligned}$$

This sampled data consists of one matching and one non-matching pair of chunks for each noisy observation chunk. We use this data for training the similarity models.

3.2. Paired-input network

In our previous work, we learn the similarity function using a DNN as $g_1(z, x)$. The network takes as input a pair of clean and noisy speech chunks (z, x) and generates a similarity score. The desired output of the similarity function is 1 for matching pairs and 0 for non-matching pairs. The structure of the network is shown in Figure 1a. The network consists of 4 hidden layers of 1024 Rectified Linear Units(ReLU) each and a single sigmoid output unit. They are initialized with random weights and trained using back-propagation with dropout of 20% and ADA-Grad stochastic gradient descent. The training minimizes the cross entropy between the predicted similarity and the desired output. In this paper we refer to this network as concatPaired.

3.3. Ranking paired-input network

In order to train the paired network for retrieval of clean chunks, we use a ranking loss. One copy of the network processes a positive pairs (z^+, x) and another copy of the network processes a negative pair (z^-, x). The model learns a similarity function $g_2(z, x)$ so that the positive pair has a higher similarity score than the negative pair, i.e.

$$g_2({z_i}^+, x_i) > g_2({z_i}^-, x_i), \forall i = \{1, 2, ...I\}. \quad (1)$$

The structure of the ranking paired network is shown in Figure 1b. Each network is identical to the concatPaired structure (Figure 1a), i.e., has 4 hidden layers of 1024 rectified linear units (ReLU), and a single sigmoid output unit. The left network is fed a matching clean-noisy pair and the right a non-matching clean-noisy pair.

We use a triplet ranking loss [12, 13] to enforce the ranking order of positive and negative pairs. A triplet ranking loss is defined between a triplet of a query, x, a positive, z^+, and a negative example, z^-, as,

$$\mathcal{L}_{\text{rank}}(x, z^+, z^-) = \max\{0, y^- - y^+\} \quad (2)$$

where $y^+ = g_2(z^+, x)$ and $y^- = g_2(z^-, x)$. The ranking loss measures the violation of the ranking order and is combined with the cross-entropy loss to create the final loss

$$\mathcal{L}(x, z^+, z^-) = \log(1 - y^-) + \log(y^+) + \mathcal{L}_{\text{rank}}(x, z^+, z^-). \quad (3)$$

This model is referred to as concatPairedRank subsequently.

3.4. Twin networks

To scale the resynthesizer to a large dictionary, we break the processing of clean and noisy chunks into separate parallel networks. We use two networks with the same architecture, but different weights, although there is no reason their architectures couldn't differ as well. Thus they are not siamese networks, and we instead call them twin networks. One network learns to map clean speech chunks $g_3(z)$ and another network learns to map noisy chunks $g_4(x)$ to a shared embedding space. The cosine similarity between the clean and noisy embeddings is used as

Table 1: *Retrieval performance: Precision-at-1: higher is better. Average rank of the correct chunk: lower is better.*

	Euclidean	**concatPaired**	**concatPairedRank**	**concatTwin**
Number of dictionary chunks	2899	2899	2899	2899
Precision-at-1	31%	96%	97%	96%
Average Rank of the correct chunk	450	6	3	12

a joining layer between the outputs of twin networks. Cosine similarity, $\cos(u, v)$, measures the cosine of the angle between two vectors, u, v, and is defined as, $\cos(u, v) = \frac{u \cdot v}{\|\mathbf{u}\|_2 \cdot \|\mathbf{v}\|_2}$.

Figure 1c shows the structure of the twin networks. Each network has 4 hidden layers of 512 rectified linear units. They are initialized with random weights and trained using back-propagation with dropout of 20% and ADA-Grad stochastic gradient descent. The desired output, y, of the model is 1 for positive pairs, (z^+, x) and 0 for negative pairs, (z^+, x).

To learn the similarity with the twin network we use the contrastive loss as defined by [14]

$$L(y, \hat{y}) = (1 - y)\frac{1}{2}\hat{y}^2 + y\frac{1}{2}\{\max(0, m - \hat{y})\}^2 \quad (4)$$

where $\hat{y}$ is the predicted similarity score of the model and m is a margin parameter. By minimizing this loss, the similarity score, $\hat{y}$ is increased for positive pairs and decreased for negative pairs.

4. Experiments

For the experiments we use the CHiME2-GRID small vocabulary data set [15]. This dataset contains read speech simulated in a living room environment. Audio files are mixed with recorded household noises in six different signal-to-noise ratios (-6 dB to 9 dB). The noises are mainly the speech of women and children, music, and various household activities. Each utterance consists of a six word sentence of the form

⟨command⟩ ⟨color⟩ ⟨preposition⟩ ⟨letter⟩ ⟨digit⟩ ⟨adverb⟩.

For example, one such sentence is "Place blue at A 9 again". Each sentence is approximately 2 seconds long. We use clean spatialized speech from the "reverberant" condition for the clean speech dictionary and speech+noise mixtures from the "isolated" condition as noisy speech. Although all signals are provided in stereo, we average the two channels together.

Our experiments are speaker dependent, i.e. we train and test on different utterances and noises with the same speaker. We selected speaker 3 for our experiments, as he had longest speech duration. We use the official training set of 500 utterances for both training and tuning purposes, 490 utterances for training and 10 for tuning. For testing we use a subset of the official development set of 24 utterances, with each mixture at a single randomly selected signal-to-noise ratio instead of all of them. There were no repeated sentences in the dataset. In total, we have approximately 16 minutes of speech from the target speaker. From these utterances which are sampled at 16kHz, we calculate log mel spectrograms and extract 11 frame chunks that overlap with their neighbors by 10 frames. This leads to 67,040 clean dictionary chunks and 124,080 clean-noisy pairs, half matching and half non-matching. We learn the three similarity models (i.e., concatPair, concatPairedRank, and concatTwin) using same training and testing utterances.

5. Evaluation

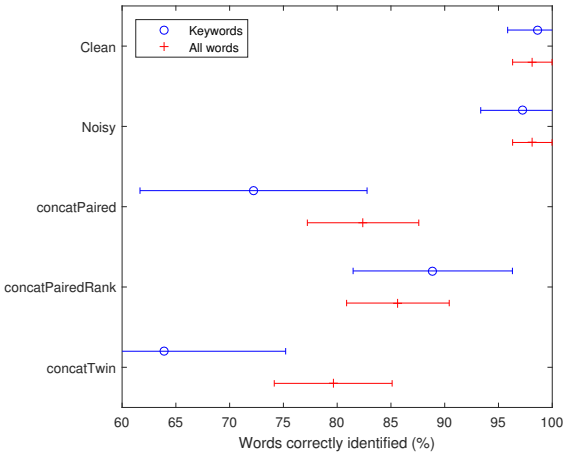
We compare the performance of the three networks in two ways. In the first test, we measure directly the retrieval performance of the similarity predictions when the exact clean segment that was used to construct a noisy segment is present in the dictionary. In the second test, we use the concatenative resynthesizer to resynthesize test files using the three different similarity models, with a dictionary constructed from a separate set of utterances, forcing the models to generalize to unseen noisy segments. These resyntheses are evaluated in terms of subjective quality and intelligibility through a listening test. Processing a single 2 second mixture of noisy speech with the concatPaired network takes approximate 169 CPU-seconds on a Intel-Xeon processor(E5-2650Lv3). With the concatTwin network, the same mixture takes approximately 69 CPU-seconds, a speedup of 2.45×. The concatTwin time estimate could still be significantly sped up by switching from a full Viterbi decode to a beam search.

5.1. Ranking of dictionary elements

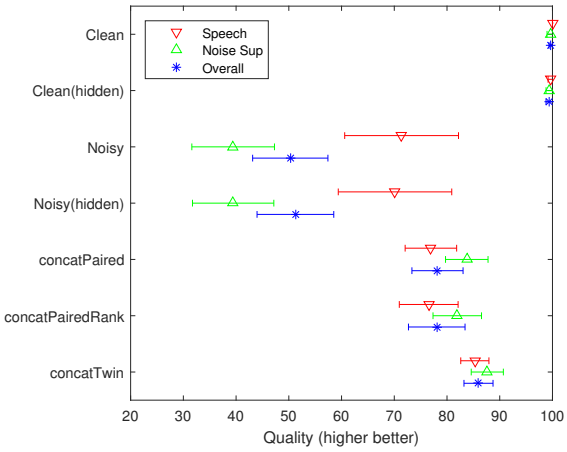
We compare the performance of the networks directly by measuring their ranking performance on dictionary elements when the correct one is present. This is not a realistic situation, but allows for the performance of the systems to be quantified directly. For this experiment, we randomly select 500 noisy chunks from the noisy observation set. We use a dictionary where for each noisy speech chunk there is exactly one matching clean chunk. We know which clean speech chunk matches each noisy chunk and this mapping serves as our ground truth. The baseline for this task is the euclidean distance between the log mel spectrum of clean and noisy chunks. We then predict the similarity of all of the clean speech dictionary elements for each noisy input and rank the clean chunks by their similarity.

Ranking performance is quantified by measuring the average precision-at-1: the percentage of queries where the top ranked dictionary element is the actual matching clean element. If the correct dictionary chunk is the top ranked element more of the time, we get higher precision-at-1. We also measure the average rank of the correct dictionary chunk for each model. We want the correct chunk to be closer to the top of the list (indicated by a lower number).

Table 1 shows the ranking performance results. The euclidean baseline gives 31% precision-at-1 and the average rank of the correct chunk is 450. Precision-at-1 is highest for the concatPairedRank model at 97%. For both concatPaired and concatTwin we get 96% precision-at-1. For concatPairedRank, the average rank of the correct chunk is 3. For concatPaired and concatTwin, the average ranks are 6 and 12, respectively. Thus it is clear that all of the concat metrics perform well on these tasks, with a slight advantage to concatPairedRank. This makes sense, as compared to concatPaired, concatPairedRank is trained to directly optimize ranking performance, and compared to concatTwin, it has a greater expressive power.



(a) *Intelligibility test results*



(b) *Quality test results*

Figure 2: *Listening test results*

5.2. Listening Tests

Speech enhancement systems are typically evaluated by two measures: intelligibility and quality of their outputs [16]. Quality measures how realistic, clear and natural the speech sounds. The ITU-T standard defines three types of speech quality that should be assessed in listening tests of noise-suppressed speech: speech quality, noise suppression quality, and overall quality [17]. Intelligibility, on the other hand, measures how accurately a listener can hear what is being said. Therefore, we evaluate the concatenative resynthesis systems by resynthesizing speech from noisy recordings and measuring its subjective quality and intelligibility. We randomly selected 12 test utterances from the randomly selected SNR levels as noisy mixtures. For each noisy mixtures, five different versions were compared: clean speech, noisy mixture, and resynthesized speech using three concat systems: concatPaired, concatPairedRank and concatTwin. Three users participated in the listening tests, 2 of them are native English speakers and 1 is a non-native English speaker.

The intelligibility test measures whether a listener's transcription of an utterance matches the original sentence. Three subjects listened to the 60 files in a randomized order. They were asked to transcribe the sentence as best they could. They were given the sample GRID grammar as an example but were also told that they did not have to adhere to the exact grammar. The results of the intelligibility test for each system averaged over all files are presented in Figure 2a. In the result, we present both the accuracy on all words and the accuracy on the letter and number words within the sentences, which in the task were designed as the keywords. Speech intelligibility for both clean and noisy utterances is very high. For the concat systems, concatPairedRank model achieves 86% accuracy on all words, concatPaired achieves 82%, and concatTwin achieves 80%.

The speech quality test compares the same five systems under a Multiple Stimuli with Hidden Reference and Anchor (MUSHRA) paradigm [18]. For each noisy file, the listener was first presented with the reference clean and noisy speech and then the outputs from the five systems unlabeled and in a randomized order. The comparisons were also presented in random order to listeners. They were asked to rate speech quality, noise suppression quality, and overall speech quality for each processed mixtures on a scale from 0 (poor) to 100 (excellent). Speech quality was defined to the users as suitability to play in living-room stereo systems or suitability of being included in a radio broadcast. Listeners were also instructed that speech intelligibility should not effect speech quality, a less-intelligible sentence could very well be high-quality sentence. The results for speech quality are presented in Figure 2 averaged over all listeners and all files. As expected, all three speech quality measures are highest for both the clean speech reference and the hidden clean speech. concatTwin performs slightly better than other two models (speech quality = 85, noise suppression = 88 and overall speech quality = 87), although all three achieve very similar ratings.

These results show that the triplet ranking loss employed by the concatPairedRank network provides better performance in both intelligibility and quality than the original paired binary loss. It also shows that the concatTwin network is able to achieve similar or slightly higher quality than the two paired networks, with slightly lower intelligibility, while providing significantly better asymptotic runtime performance.

6. Conclusions

In this work we introduced two new ways of training concatenative resynthesis similarity functions with different performance characteristics. Using a ranking loss with paired model gives better intelligibility and quality. Twin networks give much faster processing with only a minor loss in output performance. As twin networks reduces the runtime of the most expensive part of the code by several orders of magnitude, it enables us to now design real-time and large-vocabulary applications. In our future work we want to test the scalability limits of the twin networks. We also want to utilize efficient decoding techniques for large vocabulary systems.

7. Acknowledgements

Thanks to Ali Raza Syed for his implementation of the baseline system. This material is based upon work supported by the National Science Foundation under Grant No. IIS-1618061. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

8. References

[1] A. Narayanan and D. Wang, "Investigation of speech separation as a front-end for noise robust speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 826–835, Apr. 2014.

[2] M. I. Mandel, Y.-S. Cho, and Y. Wang, "Learning a concatenative resynthesis system for noise suppression," in *Proceedings of the IEEE GlobalSIP conference*, 2014.

[3] M. I. Mandel and Y. S. Cho, "Audio super-resolution using concatenative resynthesis," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2015.

[4] M. Muja and D. G. Lowe, "Scalable nearest neighbor algorithms for high dimensional data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2227–2240, nov 2014.

[5] "ANNOY," https://github.com/spotify/annoy, accessed: 2017-03-21.

[6] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," *IJPRAI*, vol. 7, no. 4, pp. 669–688, 1993.

[7] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 539–546.

[8] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708.

[9] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1891–1898.

[10] J. Hu, J. Lu, and Y.-P. Tan, "Discriminative deep metric learning for face verification in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1875–1882.

[11] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 31–35.

[12] G. Chechik, V. Sharma, U. Shalit, and S. Bengio, "Large scale online learning of image similarity through ranking," *Journal of Machine Learning Research*, vol. 11, no. Mar, pp. 1109–1135, 2010.

[13] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1386–1393.

[14] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, vol. 2. IEEE, 2006, pp. 1735–1742.

[15] E. Vincent, J. Barker, S. Watanabe, J. Le Roux, F. Nesta, and M. Matassoni, "The second 'CHiME' speech separation and recognition challenge: Datasets, tasks and baselines," May 2013, pp. 126–130.

[16] P. C. Loizou, *Speech enhancement: theory and practice*, 2nd ed. CRC press, 2013.

[17] "Subjective test methodology for evaluating speech communication systems that include noise suppression algorithm," International Telecommunication Union Telecommunication Standardization Sector, Tech. Rep. P.835, Nov. 2003.

[18] "Method for the subjective assessment of intermediate quality level of audio systems," International Telecommunication Union Radiocommunication Standardization Sector, Tech. Rep. BS.1534-3, Oct. 2015.