OBJECTIVE EVALUATION OF A DEEP NEURAL NETWORK APPROACH FOR SINGLE-CHANNEL SPEECH INTELLIGIBILITY ENHANCEMENT

Dongfu Li and Martin Bouchard

School of Electrical Engineering and Computer Science,
University of Ottawa, Ottawa, Canada
Lidongfu1983@gmail.com, martin.bouchard@uottawa.ca

ABSTRACT

Single-channel speech intelligibility enhancement is much more difficult than multi-channel intelligibility enhancement. It has recently been reported that machine learning training-based single-channel speech intelligibility enhancement algorithms perform better than traditional algorithms. In this paper, the performance of a deep neural network method using a multi-resolution cochlea-gram feature set recently proposed to perform single-channel speech intelligibility enhancement processing is evaluated. Various conditions such as different speakers for training and testing as well as different noise conditions are tested. Simulations and objective test results show that the method performs better than another deep neural networks setup recently proposed for the same task, and leads to a more robust convergence compared to a recently proposed Gaussian mixture model approach.

KEYWORDS

Single-channel speech intelligibility enhancement processing, Deep Neural Networks (DNN), Multi-Resolution CochleaGram (MRCG), Gaussian Mixture Models (GMM)

1. INTRODUCTION

Single-channel speech intelligibility enhancement is more challenging than multi-channel speech intelligibility enhancement, because information about the spatial sound propagation is not available. In most cases, it is hard for a single-channel noise-reduction algorithm to know how and to what extent to modify a specific parameter to improve speech intelligibility [1].

A lot of previous work has been designed on the prior knowledge or estimation of noise, such as the a priori Signal-to-Noise Ratio (SNR) algorithm, the Minimum Mean-Square Error (MMSE) approach, the log-MMSE approach, the Wiener filter, and so on. They have all been shown to be ineffective for intelligibility enhancement [1].

Two machine learning training-based methods have recently been proposed for intelligibility improvement: the Gaussian Mixture Models (GMM)-based approach [2] and the Deep Neural

David C. Wyld et al. (Eds) : CSEN, SIPR, NCWC - 2016
pp. 111–123, 2016. © CS & IT-CSCP 2016 DOI : 10.5121/csit.2016.61010

Networks (DNN)-based approach [3]. In 2006, an efficient way to train a multilayer neural network was proposed [4] and a new area of machine learning emerged, which is called deep learning, deep hierarchical learning or DNN [5],[6]. Key aspects of machine learning and artificial intelligence have been widened by the techniques developed from deep learning [7]-[9], and there are several active researchers in this area [10]. For speech intelligibility processing using DNNs, in 2013 a DNN using 85 features as inputs was proposed for a speech recognition task with hearing-impaired listeners [11]. In 2014, it was reported that the Multi-Resolution CochleaGram (MRCG) feature set produced a better result in a multilayer perceptron neural network, which is a simpler type of neural network [3].

Under some conditions, it has been reported that the DNN approach, which better represents the state of the art in machine learning, generalizes better and better processes previously unseen data patterns compared to the GMM method. The main goal of this paper is to evaluate the performance of the DNN method proposed in [3] using objective measures under different conditions (different types and levels of noise, mismatch between training set and testing set, etc.) and to compare the performance with the GMM approach previously proposed for the same task [2].

2. DEEP NEURAL NETWORKS

2.1 Pretraining DNN with a Restricted Boltzmann Machine

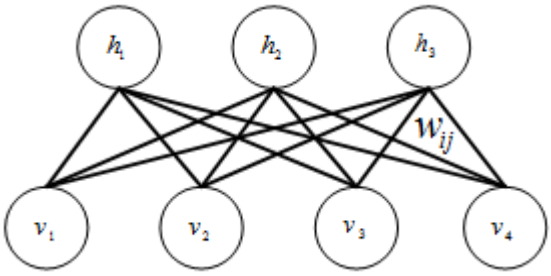

Figure 1. RBM with 4 visible units and 3 hidden units.

Training deep neural networks is challenging, because training can easily get stuck in undesired local optima which prevent the deeper layers from learning useful features. This problem can be partially circumvented by pretraining, i.e., performing a step of unsupervised training before the supervised learning step. The Restricted Boltzmann Machine (RBM) method is a useful way to conduct DNN training. A RBM is a simplified kind of a Boltzmann Machine with no visible-visible units connections and hidden-hidden units connections. In a Restricted Boltzmann Machine, a visible unit only has connections to hidden units, and reversely a hidden unit only has connections to visible units. This special kind of structure provides the advantage that when a visible unit is learning its optimal weights corresponding to a set of hidden units, the learning is independent to other visible units. This advantage also applies to the training of hidden units. Then the whole network can be trained in parallel. With the great progress made in graphics processing unit processors (GPU), i.e., in parallel computing, this can become a great advantage. Figure 1 shows a simple Restricted Boltzmann Machine with 4 visible units and 3 hidden units.

The "energy" of a Restricted Boltzmann Machine can be written as below:

$$E(\mathbf{s}) = -\sum_i s_i b_i - \sum_{i<j} s_i s_j w_{ij} \tag{1}$$

where $\boldsymbol{s}$ is the state vector $\boldsymbol{s} = \{ v_1, v_2, v_3, \ldots, v_n, h_1, h_2, \ldots, h_m \}$, n is the number of visible units, m is the number of hidden units, and s_i is a component of $\mathbf{s}$ that can be a visible unit or a hidden unit. If s_j is the unit connected to s_i, then w_{ij} is the weight between s_i and s_j

2.2 Restricted Boltzmann Machine Learning

Let unit i be a unit to update its binary state. The total input z_i for this unit i is the sum of its bias b_i and the weighted products from connections to other units:

$$z_i = b_i + \sum_j s_j w_{ij} \tag{2}$$

The probability for this unit to turn on or off is given by a logistic function:

$$prob(s_i = 1) = \frac{1}{1 + e^{-z_i}} \tag{3}$$

As mentioned earlier, the visible units only connect to hidden units. For a given state vector $\boldsymbol{s}$, no matter if the network is updated in any order, the network will eventually reach a stationary distribution (equilibrium). Then for all possible binary state vectors $\boldsymbol{u}$, the probability of vector $\boldsymbol{s}$ can be given by the energy:

$$P(\mathbf{s}) = e^{E(s)} \Big/ \sum_{\mathbf{u}} e^{E(\mathbf{u})} \tag{4}$$

Given a training set of state vectors (data), the goal of the learning is to find the optimal weights and biases to make the state vectors maximize the product of the probabilities that the Boltzmann machine assigns to the binary vectors in the training set. By differentiating (4) using $\partial E(\mathbf{s}) / \partial w_{ij} = -s_i s_j$, it can be shown that:

$$\sum_{\mathbf{s}} \frac{\partial \log P(\mathbf{s})}{\partial w_{ij}} = \left\langle s_i s_j \right\rangle_{data} - \left\langle s_i s_j \right\rangle_{model} \tag{5}$$

where $\left\langle s_i s_j \right\rangle_{data}$ is the state value of the data distribution and $\left\langle s_i s_j \right\rangle_{model}$ is the state value when the Boltzmann machine is sampling state vectors from its equilibrium distribution. Then the gradient ascent is surprisingly simple, because the differentiation $\sum_{\mathbf{s}} \frac{\partial \log P(\mathbf{s})}{\partial w_{ij}}$ only depends on the states:

$$\Delta w_{ij} \quad \propto \quad \left\langle s_i s_j \right\rangle_{data} - \left\langle s_i s_j \right\rangle_{model} \tag{6}$$

The update value Δw_{ij} is the product of $\langle s_i s_j \rangle_{data} - \langle s_i s_j \rangle_{model}$ and the learning rate. The learning rate can be constant or it can vary during the training steps to satisfy different situations. To get the Boltzmann equilibrium distribution, we can follow the steps below:

1) Starting with a data vector on visible units, update all of the hidden units in parallel;

2) Update all of the visible units in parallel to get a "reconstruction" of the visible units;

3) Update all of the hidden units again until it is the equilibrium distribution.

This algorithm may take a long time to get to the equilibrium. It can be stopped at step 3) or continue iteratively to update 1) and 2), this is called "contrastive divergence" and has been found to work well in practice [12].

3. MULTI-RESOLUTION COCHLEAGRAM FEATURE

The multi-resolution cochleagram (MRCG) feature was originally proposed in [3]. The MRCG feature is a multi-resolution power distribution of an acoustic signal in the time-frequency representation. The cochleagram represents the excitation pattern on the basilar membrane in the inner ear as a function of time. Four cochleagrams at different frequency resolutions are combined to form the MRCG feature, including one high resolution cochleagram and three low resolution cochleagrams.

The cochleagram is calculated in two steps. The input signal is first filtered by a gammatone filter bank:

$$g_{f_c}(t) = t^{N-1}\exp[-2\pi t\, b(f_c)]\cos(2\pi f_c t)u(t) \tag{7}$$

where f_c , the center frequencies, are uniformly spaced on the equivalent rectangular bandwidth (ERB) scale, N is the order of the filter, $u(t)$ is the step function, and $b(f_c)$ is the bandwidth related to f_c :

$$b(f_c) = 1.019 * ERB(f_c) = 1.019 * 21.4 * \log(1 + 0.00437 * f_c) \tag{8}$$

$b(f_c)$ increases as f_c increases, which means that the frequency resolution decreases as the frequency increases. The signal is then divided into frames. A cochleagram is the power of each time frame and in each gammatone bank channel.

The MRCG feature can then be described as below:

1) Given an input, compute a 64-channel cochleagram (CG1) using 20 ms frames with 10 ms frame shifts, and apply a log operation on the output in each time-frequency (T-F) unit;

2) CG2 is similar as CG1, but with 200 ms frames and 10 ms frame shifts;

3) CG3 is derived by averaging CG1 across a square window of 11 frequency channels and 11 time frames centered at a given T-F unit (zero padding is applied at the edges of CG1 when units outside CG1 are needed in the averaging process);

4) CG4 is similar as CG3, but with a 23*23 square window;

5) Concatenate CG1-CG4 to obtain the MRCG feature.

Delta and double-delta features (i.e., computing the differences of feature values at consecutive times, and computing the differences of those differences) are widely used in speech processing to capture temporal dynamics. Delta and double-delta features were thus also used on the MRCG features generated above. Then the final MRCG feature set is obtained. Each time frame becomes an MRCG feature set of dimension 64 by 12.

4. MRCG-DNN ALGORITHM

The MRCG-DNN algorithm is a kind of channel-selection algorithm. A channel-selection based algorithm will retain the T-F bins where speech is dominant, and discard the T-F bins where noise is dominant. The algorithm makes the retain/discard decision based on the SNR of each T-F bin, compared to a threshold called the local SNR criterion (LC). In an ideal case, where separate access to clean speech and noise-only signals is possible and the exact SNR can be computed, using a channel-selection algorithm we can get the ideal binary mask (IDBM):

$$B(k,t)=\begin{cases}1 & SNR > LC\\ 0 & otherwise\end{cases} \tag{9}$$

where $B(k,t)$ is the IDBM mask at channel (frequency) k and time t.

The IDBM has been shown to improve speech intelligibility at any input SNR level [1] (even as low as -40 dB) . The IDBM is an unrealistic condition that can never occur in real life, but for several reasons this result is important. In particular, the outcome of the IDBM can provide an upper bound, so that the IDBM can be a criterion to estimate the performance of a practical algorithm.

The MRCG-DNN algorithm uses a relatively large corpus of speech and noise sources, together with the calculated IDBM as the input of a system to be trained. An overview of MRCG-DNN algorithm is shown in Figure 2.

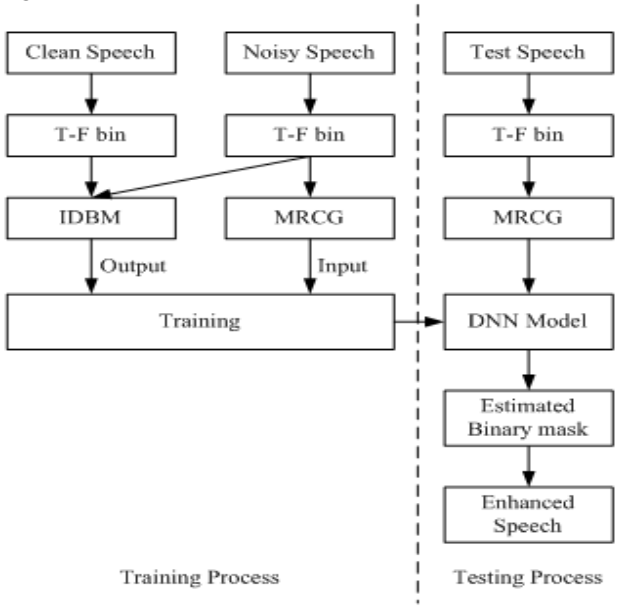

Figure 2. Overview of the MRCG-DNN algorithm

5. RESULTS

To test how the MRCG-DNN algorithm performs in different situations, we did some tests using the HINT speech database [13], the TIMIT speech database (LDC93S1) [14], and noise from the Aurora 2 database [15]. The HINT speech files have a 2 to 3 seconds length for each file, all from the same male speaker. The TIMIT is a corpus of phonemically and lexically transcribed speech of American English speakers of different genders and dialects. Each sentence is about 3 to 5 seconds long. The Aurora noise files contain different situations of noise, including babble, airport, restaurant, and street. In the tests we have used these four kinds of noise.

In this paper, the Matlab™ code is based on two Deep Learning toolboxes: DeepLearnToolBox [16] and DeepNeuralNetwork [17]. The bone structure of the first toolbox, a few active functions of the second toolbox, and some improvements were combined together to perform the simulations of this work.

In the following tests, 200 clean speech files are used for training (approx. 5-6 minutes of recordings), and 70 clean speech files are used for testing (approx. 2-3 minutes of recordings). If the training data is too short (i.e., less than 1 min) the result was found to be poor. If the training data is too long, it can either cause the computer to run out of memory or to take too much processing time. Using 5-6 minutes of recordings for training was found to produce a good result with a relatively fast processing speed. As a comparison, in [18] 390 sentences were used for training and in [3] 100 sentences were used. The sampling rate was set to 16000 Hz. The noisy files were produced by randomly selecting a noise segment that has the same length as the clean speech files and adding them together with the desired SNR. The tests were operated using Matlab™ on a Windows 7™ (64-bit) system, with an Intel Core™ i5-4310M CPU, and 8 GB memory.

To evaluate the estimated mask compared to the IDBM, a method called the HIT - FA (HIT minus FA) metric is used, which has been shown to correlate well with human intelligibility [1]. The HIT is the probability of correct detection (the percentage of target-dominant T-F units correctly classified), while the FA is the probability of false alarm (the percentage of noise-dominant units incorrectly classified). Both the HIT and FA need to be taken into account when evaluating the performance of the binary mask estimate, and the HIT-FA is therefore a simple difference metric that can be used to quantify the performance of the estimate.

5.1 Tests for simple noises

These first tests used the HINT database, which is a single speaker database. White, purple and pink noises were used. The model was trained with -5dB, 0 dB and 5dB input SNR, and tests were performed with -5dB SNR. For each noise that was tested, the DNN model was trained with speech from the training set and additive noise consisting of only one noise type. The testing phase was performed with speech from the test set and again additive noise consisting of only the same noise type. Table 1 shows the HIT-FA results as well as URL links to the clean test male speech sound file, the noisy sound files and the processed sound files.

Table 1. Results for simple noises

Clean test speech					
Noise type	**HIT**	**FA**	**HIT-FA**	**noisy**	**processed**
white	84.2%	2.6%	81.6%	link	link
purple	92.3%	3.1%	89.2%	link	link
pink	78.5%	20.2%	58.3%	link	link

From Table 1 we can see that the DNN model performs very well for white noise, and even better for purple noise. The HIT rate is very high, while the FA rate is very low. The processed sound files are also quite clear. On the other hand, the DNN model produces a much higher FA for pink noise, and the processed file is highly distorted. The reason for this is that purple noise energy is mainly in high frequency bands, while pink noise is mainly in low frequency bands, and human voice is also mostly distributed in low frequencies. In the purple noise case, the DNN model can easily separate them, it behaves as a kind of high-end low pass filter. In the pink noise condition, the separation is more difficult for the DNN because of the increased frequency overlap between speech and noise.

5.2 Same speaker for training and testing, same noise type for training and testing

In this test we used the HINT database which has a single speaker, with different sentences for training and testing. We used four kinds of realistic noise: airport, babble, restaurant, and street. We trained the DNN model with -5dB, 0 dB and 5dB input SNR, and then performed the testing with -5dB SNR. For each type of noise we trained the model and then used the model to test different sentences corrupted by the same kind of noise as the one used for training (but of course using different noise segments). Table 2 shows the HIT-FA results and URL links to the clean test male speech sound file, the noisy sound files and the processed sound files.

Table 2. Results with same speaker for training and testing, same noise type for training and testing.

Clean test speech					
Noise type	**HIT**	**FA**	**HIT-FA**	**noisy**	**processed**
babble	76.2%	7.3%	68.9%	link	link
airport	80.3%	11.0%	69.3%	link	link
restaurant	77.5%	9.9%	67.6%	link	link
street	78.2%	7.5%	70.7%	link	link

From Table 2 we can see that the HIT rate of the estimated mask is relatively high and the FA rate is low, regardless of the noise type. The resulting HIT-FA rate are fairly good and are comparable with previous results from the literature [18]. When listening to the processed output files, it is debatable if the intelligibility is really improved or not compared to the original noisy files, i.e., subjective listening tests would be required to fully determine this. In some cases (e.g. babble noise) the intelligibility of the processed files appears to be better than for other cases (e.g. street noise), so this indicates that the performance can be noise dependent. It should be noted that the noises used for testing were challenging noises, but they are more likely to correspond to real noise conditions in practice.

5.3 Different speakers for training and testing, same noise type for training and testing

In this section, we used the TIMIT dataset with several different speakers for training and another speaker was used for testing (but for the same female gender as the ones used in training). Four

kinds of noise were used: airport, babble, restaurant, and street. We trained the DNN model with -5dB, 0 dB and 5dB SNR, and did testing at -5dB SNR. For each type of noise we trained the model and performed the testing using the same type of noise (but different noise segments). Table 3 shows the HIT-FA results and URL links to the clean test female speech sound file, the noisy sound files and the processed sound files.

Table 3. Results for different speakers for training and testing, same noise type for training and testing.

Clean test speech					
Noise type	**HIT**	**FA**	**HIT-FA**	**noisy**	**processed**
airport	83.8%	8.2%	75.6%	link	link
babble	80.6%	6.9%	73.7%	link	link
restaurant	82.2%	13.2%	69.0%	link	link
street	84.4%	6.0%	78.4%	link	link

FromTable 3 we can see that the HIT rate of the estimated mask is relatively high and the FA rate is low, regardless of the noise type. This indicates that in terms of HIT-FA rate the performance of the MRCG-DNN method can be robust to different speakers, i.e., when the speakers used for training differ from the speakers used in testing. Although it is possible to train simultaneously for both male and female speech, our experience has been that better results are obtained when the same gender is used for training and testing. In terms of intelligibility, as in the results of Table 2, it is debatable if the intelligibility is improved in the processed sound files of Table 3, and the performance seems to be better for some noise types (e.g. airport and street).

5.4 Training with 3 types of noise and testing with a fourth type of noise

For this test, we have used either the HINT dataset (single speaker) or the TIMIT dataset (different speakers), as well as airport, babble, restaurant, and street noise. We trained the DNN model with airport, babble and restaurant noise, then we tested the model with street noise. For the HINT database the training and testing sentences were different but from the same speaker. For the TIMIT database, training and testing were made with different speakers (of the same gender) and different sentences. Table 4 shows the HIT-FA results and URL links to the clean test speech sound file, the noisy sound files and the processed sound files.

Table 4. Results for training with 3 types of noise and testing with a fourth type of noise.

Clean test speech					
Database	**HIT**	**FA**	**HIT-FA**	**noisy**	**processed**
HINT database	76.2%	11.4%	64.8%	link	link
Clean test speech					
Database	**HIT**	**FA**	**HIT-FA**	**noisy**	**processed**
TIMIT database	84.6%	18.9%	65.7%	link	link

From Table 4, we can see that the HIT rate of the estimated mask is relatively high and the FA rate is low. This indicates that in terms of HIT-FA rate the performance of the MRCG-DNN method can in principle be robust to conditions where we have both different speakers and different noise types between training and testing, as long as training is done with appropriate data (i.e., the training speech should have some similarity with the test speech, the training noise should have

some similarity with the test noise). But as observed in the previous results of Table 2 and Table 3, for the results of Table 4 it is highly debatable if the intelligibility is improved.

5.5 Training with different levels of noise, testing with the same noise type at a specific level

Here we used the HINT dataset (same speaker for training and testing, but different sentences), and babble noise for training and testing. We trained the DNN model with -5dB, 0 dB and 5dB SNR, then tested with -5dB SNR. We trained the DNN model with -7dB, -5 dB and 0dB SNR, then tested with -5dB SNR. Finally, we trained the DNN model with -10dB, -7 dB and -5dB SNR, and tested with -5dB SNR. **Error! Reference source not found.** shows the HIT-FA results and URL links to the clean test speech sound file, the noisy sound files and the processed sound files.

Table 5. Results for training with different levels of noise, testing with the same noise type at a specific level.

Clean test speech					
SNR	**HIT**	**FA**	**HIT-FA**	**noisy**	**processed**
-5dB, 0dB, 5dB	76.2%	7.3%	68.9%	link	link
-7dB,-5dB, 0dB	74.8%	6.6%	68.1%	link	link
-10dB,-7dB,-5dB	70.3%	5.3%	64.9%	link	link

From Table 5 we can see that the HIT rate as well as the FA and HIT-FA rates of the estimated mask all slightly drop from the top row to the bottom row. This indicates that training with data having the same and higher SNR than the testing data may lead to a slightly higher HIT-FA rate. Informally, the intelligibility in the sound files also seems to follow that trend.

5.6 Training with different local SNR criterion (i.e., thresholds used for binary mask decision)

For this test, we used the HINT dataset (same speaker for training and testing, but different sentences), and babble noise for training and testing. We trained the DNN model with -5dB, 0 dB and 5dB SNR, and tested with -5dB SNR. We did this procedure 3 times, and the difference between the 3 cases is that different local SNR criterion values were used (thresholds used for the binary mask decision): 0dB, -5dB, and -10dB. Table 6 shows the HIT-FA results and URL links to the clean test speech sound file, the noisy sound files and the processed sound files.

Table 6 Results of training with different local SNR criterion (thresholds)

Clean test speech					
local SNR criterion	**HIT**	**FA**	**HIT-FA**	**noisy**	**processed**
0dB	74.3%	8.8%	65.5%	link	link
-5dB	76.2%	7.3%	68.9%	link	link
-10dB	79.9%	11.7%	68.3%	link	link

From Table 6 we can see that the HIT rate of the estimated mask increases as the local SNR criterion (threshold) decreases, but the HIT-FA rate doesn't change too much. Listening to the resulting processed files, we note (informally) that the intelligibility improves from the top row to the bottom row. A higher HIT rate means that a higher percentage of target-dominant T-F bins are kept, so based on these results we adopted a dynamic adjustment method during training in the software program, to make sure that the HIT rate is at least 75%. If the condition is not fulfilled

then we dynamically lower the local SNR criterion used for training. But overall, as in previous tables, it is debatable if the processed files provide some intelligibility improvement over the original noisy files.

5.7 MRCG-DNN compared with other approaches

In this section we use two other types of speech intelligibility enhancement approaches and compare them with the MRCG-DNN approach: an DNN using 85 input features proposed in [11] and an Amplitude Modulation Spectrogram (AMS)-GMM proposed in [18]. We used our own implementation of the 85 input features DNN and the original code from [18] for the AMS-GMM. The database and noise conditions were the same as the ones originally used in the 85 input features DNN and the AMS-GMM. The IEEE speech database is also used as clean speech data in this section [19].

For the DNN with 85 input features, the input data is passed through a 64 band gammatone filter. Each subband is divided to 20 ms frames with 10 ms overlap. Each T-F unit extracts 85 features: 15 AMS features, 13 Relative Spectral Transform - Perceptual Linear Prediction (RASTA-PLP) features, 31 Mel-Frequency Cepstral Coefficients (MFCC) features, and 13 delta features for the RASTA-PLP features. This method uses a DNN model for each subband, for a total of 64 DNN models. The MRCG-DNN on the other hand only uses one DNN model for all subbands, so the MRCG-DNN training takes much less time than the 85 input features DNN.

The results for the 85 input features DNN and the MRCG-DNN are compared in Table 7. They both use the same DNN structure, and were trained for 100 iterations. “n6 noise” is a babble noise produced by adding several TIMIT sentences together. We can see from Table 7 that the HIT-FA for the MRCG-DNN is always significantly higher than for the 85 input features DNN, indicating a better performance in terms of HIT-FA rate.

Table 7. Results for 85 input features DNN and MRCG-DNN.

		85 input features DNN			**MRCG-DNN**		
database	**noise**	**HIT**	**FA**	**HIT-FA**	**HIT**	**FA**	**HIT-FA**
HINT	n6 (babble)	62.9%	7.4%	55.5%	72.3%	6.4%	65.9%
IEEE	factory	46.2%	3.4%	42.8%	66.5%	3.8%	62.7%

The other method used for comparison is the AMS-GMM [18]. More specifically, in this approach 4 sub-GMMs are used in order to make the decision to set the TF binary mask to 1 or 0. The results for the AMS-GMM and the MRCG-DNN are compared in Table 8. The HIT-FA of the MRCG-DNN is slightly less (worse) than for the AMS-GMM method, but with a much lower FA (better). Overall it can be said that the results of these two methods are fairly comparable for the considered setup, in terms of HIT-FA rate. However, we also found that the AMS-GMM method fails to find a clustering solution in some conditions (e.g., IEEE database speech with factory noise), while the MRCG-DNN method always converges to a solution, so it was found to be much more robust.

Table 8. Results for AMS-GMM and MRCG-DNN.

		AMS-GMM			MRCG-DNN		
database	**noise**	**HIT**	**FA**	**HIT-FA**	**HIT**	**FA**	**HIT-FA**
IEEE	speech shaped-noise	93.4%	12.3%	81.12%	80.5%	3%	77.4%

In conclusion, the MRCG-DNN has a better HIT-FA than the DNN with other features (85 input features DNN), and it is more robust than the AMS-GMM method, with a similar performance. In addtion, the MRCG-DNN method is faster for training than the 85 input features DNN and the AMS-GMM methods: the MRCG-DNN takes about half an hour of training time, while the 85 input features DNN and AMS-GMM both need more than three hours of training time. Therefore, compared to other available methods, the MRCG-DNN was confirmed as a good choice for attempting to improve speech intelligibility through maximizing the HIT-FA rate.

6. CONCLUSIONS

The MRCG-DNN approach was found to outperform other approaches that have appeared in the literature for single-channel speech intelligibility enhancement processing, either in terms of objective measures (HIT-FA rate) or in terms of robustness to converge to a solution. Using the DNN method with "easier" noises (white noise, purple noise) lead to a better HIT-FA rate performance because of the stationarity of the noise and the reduced overlap with the speech content. For cases with more challenging noise conditions, the HIT-FA rate performance of the different approaches was not as good and the true intelligibility improvement was more debatable. Training with more data (clean speech data, noise data, or both) could be an option to attempt to improve the performance, although that option may not always be feasible in real-life scenarios.

ACKNOWLEDGEMENTS

Financial support for this work was provided by the Correctional Service Canada and by a NSERC Discovery grant.

REFERENCES

[1] Loizou, P. C. (2013) Speech enhancement: theory and practice. CRC press, 2nd ed.

[2] Kim, G., and Loizou, P. C. (2010) "Improving speech intelligibility in noise using environment-optimized algorithms." IEEE Transactions on Audio, Speech, and Language Processing, Vol. 18, No.8, pp.2080-2090.

[3] Chen, J., Wang, Y., and Wang, D. (2014) "A feature study for classification-based speech separation at very low signal-to-noise ratio." Proceedings of 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp.7039-7043.

[4] Hinton, G. E., and Salakhutdinov, R. R. (2006) "Reducing the dimensionality of data with neural networks." Science, Vol. 313, No. 5786, pp.504-507.

[5] Bengio, Y. (2009) "Learning deep architectures for AI." Foundations and trends in machine learning, Vol. 2, No.1, pp.1-127.

[6] Hinton, G. E., Osindero, S., and Teh, Y. W. (2006) "A fast learning algorithm for deep belief nets." Neural computation, Vol.18, No.7, pp.1527-1554.

[7] Arel, I., Rose, D. C., and Karnowski, T. P. (2010) "Deep machine learning-a new frontier in artificial intelligence research." IEEE Computational Intelligence Magazine, Vol. 5, No. 4, pp.13-18.

[8] Deng, L. (2011) "An overview of deep-structured learning for information processing" Proceedings of Asian-Pacific Signal & Information Processing Annual Summit and Conference (APSIPA-ASC).

[9] Yu, D., and Deng, L. (2011) "Deep learning and its applications to signal and information processing." Signal Processing Magazine, Vol.28, No.1, pp.145-154.

[10] Deng, L., and Yu, D. (2014) "Deep learning: methods and applications." Foundations and Trends in Signal Processing, Vol. 7, No.3–4, pp.197-387.

[11] Healy, E. W., Yoho, S. E., Wang, Y., and Wang, D. (2013) "An algorithm to improve speech recognition in noise for hearing-impaired listeners." The Journal of the Acoustical Society of America, Vol. 134, No. 4, pp.3029-3038.

[12] Carreira-Perpinan, M. A., and Hinton, G. E. (2005) "On contrastive divergence learning." Proceedings of the 10th international workshop on artificial intelligence and statistics, pp.33-40.

[13] Nilsson, M., Soli, S. D., and Sullivan, J. A. (1994) "Development of the Hearing in Noise Test for the measurement of speech reception thresholds in quiet and in noise." The Journal of the Acoustical Society of America, Vol. 95, No.2, pp.1085-1099.

[14] TIMIT Acoustic-Phonetic Continuous Speech Corpus https://catalog.ldc.upenn.edu/LDC93S1

[15] AURORA Project Database 2.0 - Evaluation Package, http://catalog.elra.info/product_info.php?products_id=693

[16] DeepLearnToolbox, https://github.com/rasmusbergpalm/DeepLearnToolbox

[17] Deep Neural Network toolbox, http://www.mathworks.com/matlabcentral/fileexchange/42853-deep-neural-network

[18] Kim, G., Lu, Y., Hu, Y., and Loizou, P. C. (2009) "An algorithm that improves speech intelligibility in noise for normal-hearing listeners." The Journal of the Acoustical Society of America, Vol.126, No. 3, pp.1486-1494.

[19] Rothauser, E. H., Chapman, W. D., Guttman, N., Nordby, K. S., Silbiger, H. R., Urbanek, G. E., and Weinstock, M. (1969) "IEEE recommended practice for speech quality measurements." IEEE Trans. Audio Electroacoust., Vol. 17, No.3, pp.225-246.

AUTHORS

Dongfu Li received the M.A.Sc. degree in Electrical and Computer Engineering from the University of Ottawa in 2016, a M. Eng. degree from Chongqing University of Posts and Telecommunications in 2010, and a B. Eng. from Chongqing University of Posts and Telecommunications in 2007. He has completed internships at Chongqing Jiuzhou Starnav System Co. in 2010-2011, at Chongqing University of Posts and Telecommunications in 2008-2009, and at Chongqing Aerospace New Century Satellite Application Technology Co. in 2007-2008.

Martin Bouchard received the B.Eng., M.App.Sc. and Ph.D. degrees in Electrical Engineering from Université de Sherbrooke (Sherbrooke, Qc., Canada) in 1993, 1995 and 1997 respectively. In January 1998, he joined the School of Electrical Engineering and Computer Science (formerly known as the School of Information Technology and Engineering) at the University of Ottawa (Ottawa, Canada). Over the years Professor Bouchard has conducted research activities and consulting activities with several private and governmental partners. He has served as a member of the Speech and Language Technical Committee (SLTC, IEEE Signal Processing Society, 2009-2011), as an Associate Editor for the EURASIP Journal on Audio, Speech and Music Processing (2006-2011), and an Associate Editor for the IEEE Transactions on Neural Networks (2008-2009). His current research interests are signal processing methods in general, with an emphasis on speech, audio, acoustics and hearing aids applications.