Deep Neural Network-Based Noise Estimation for Robust ASR in Dual-Microphone Smartphones

Iván López-Espejo*, Antonio M. Peinado*, Angel M. Gomez*, and Juan M. Martín-Doñas†

Dept. of Signal Theory, Telematics and Communications,
University of Granada, Spain
*{iloes,amp,amgg}@ugr.es
†mdjuamart@correo.ugr.es

Abstract. The performance of many noise-robust automatic speech recognition (ASR) methods, such as vector Taylor series (VTS) feature compensation, heavily depends on an estimation of the noise that contaminates speech. Therefore, providing accurate noise estimates for this kind of methods is crucial as well as a challenge. In this paper we investigate the use of deep neural networks (DNNs) to perform noise estimation in dual-microphone smartphones. Thanks to the powerful regression capabilities of DNNs, accurate noise estimates can be obtained by just using simple features as well as exploiting the power level difference (PLD) between the two microphones of the smartphone when employed in close-talk conditions. This is confirmed by our word recognition results on the AURORA2-2C (AURORA2 - 2 Channels - Conversational Position) database by largely outperforming single- and dual-channel noise estimation algorithms from the state-of-the-art when used together with a VTS feature compensation method.

Keywords: Noise estimation, Deep neural network, VTS feature compensation, Automatic speech recognition, Dual-microphone, Smartphone

1 Introduction

Providing robustness against acoustic noise is still a challenge in automatic speech recognition (ASR) [1]. Many techniques devoted to noise-robust ASR such as vector Taylor series (VTS) feature compensation [2] or Wiener filtering [3] might require an explicit estimation of the noise that contaminates speech. The performance of this kind of techniques heavily depends on the accuracy of the given noise estimation. Therefore, accurate noise estimation algorithms are needed and we can find a great variety of them in the literature [4–7].

Particularly nowadays, providing robustness in ASR is a crucial task because of the wide use of mobile devices such as smartphones or tablets which can be employed for ASR purposes in many different acoustic environments. Mobile devices often integrate small microphone arrays which have been successfully

* This work has been supported by the Spanish MINECO TEC2013-46690-P project.

exploited for both speech enhancement [8, 9] and noise-robust ASR [10]. In our previous work [11], noise-robust ASR in a dual-microphone smartphone, a particular case of interest of small microphone array, was addressed. In that work, a missing-data mask estimation method was proposed in order to perform spectral reconstruction. The method proposed consists of a deep neural network (DNN) fed with dual-channel noisy observations which provides a missing-data mask. It was proven that this rather simple and straightforward approach supplies quite accurate missing-data masks by exploiting the power level difference (PLD) between the two microphones of the smartphone. When a dual-microphone smartphone is employed in close-talk conditions (i.e. the loudspeaker of the smartphone is placed on the ear of the user) the primary microphone (which is located at the bottom of the device) captures more speech power than the secondary one (located at the rear of the smartphone) since the latter is placed in an acoustic shadow with respect to the speaker's mouth. Additionally, the noise power observed by both microphones is assumed to be very similar because of the typical existence of a homogeneous noise field [8]. Thus, it is clear that a missing-data mask can be easily derived from a comparison between the noisy speech power present in both channels, where the secondary channel is a good noise reference.

Based on the discussion above, in this work we investigate noise estimation for noise-robust ASR in dual-microphone smartphones by exploiting the PLD between the two sensors of the device. Similarly to our previous work [11], a DNN is used to find a mapping function between the dual-channel noisy observation and the noise that contaminates speech at the primary channel. While DNNs have been employed for many different tasks from noise-robust ASR such as missing-data mask estimation [12, 11], surprisingly they have not yet been applied to directly estimate noise. DNN-based noise estimates will be used by a VTS feature compensation method to perform noise-robust ASR. The resulting robust ASR system gathers the advantages of two different approaches. Thus, it combines a traditional signal processing technique for feature compensation with a novel DNN-based approach for noise estimation, which undoubtedly is a difficult task with the classical signal processing tools. It is expected that this kind of hybrid architectures will be extensively explored in the near future [13]. Our experimental evaluation on the AURORA2-2C (AURORA2 - 2 Channels - Conversational Position) database [10] shows the effectiveness of our proposal in terms of word accuracy by achieving the best performance among several single- and dual-channel noise estimation algorithms from the state-of-the-art.

The rest of the paper has been organized as follows. In Section 2 a system overview is presented along with the proposed DNN-based noise estimation method. Both the experimental framework and results are shown in Section 3. Finally, in Section 4 our conclusions and future work are drawn.

2 Proposed Method

The noise-robust ASR framework considered in this paper is depicted in Figure 1. The noisy speech signal captured by the primary microphone of the smartphone is denoted as $y_1(m)$, where m is the sampling time index. Similarly, $y_2(m)$

refers to the noisy speech signal recorded by the secondary microphone of the device. As presented in Section 1, the noise components in $y_1(m)$ and $y_2(m)$ are assumed to be quite similar while speech is very attenuated at the secondary sensor with respect to the primary one since the former is placed in an acoustic shadow regarding the speaker's mouth. Then, log-Mel spectral features $\mathbf{y}_i$ are extracted from the noisy signals $y_i(m)$, $i = 1, 2$, which are employed by a DNN-based stage in order to provide a noise estimate of the primary channel, $\hat{\mathbf{n}}_1$. To obtain the clean speech log-Mel features at the primary channel, $\hat{\mathbf{x}}_1$, this noise estimate is used along with $\mathbf{y}_1$ by a VTS feature compensation method. Finally, $\hat{\mathbf{x}}_1$ is transformed into the cepstral domain by application of the discrete cosine transform (DCT) prior to be used by the speech recognizer.

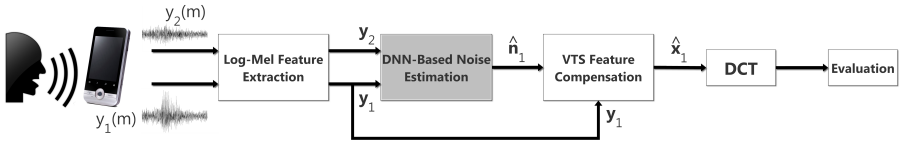

Fig. 1. Block diagram of the noise-robust ASR framework considered in this work.

Subsection 2.1 is devoted to show the fundamentals of the DNN-based noise estimation stage marked in gray in Figure 1, while a noise-aware training (NAT)-based extension to it, intended to increase the awareness of the DNN about the noise that contaminates speech, is presented in Subsection 2.2.

2.1 Dual-Channel Noise Estimation Based on DNN

A DNN (i.e. a feed-forward neural network with multiple hidden layers) is considered in this work to find a non-linear mapping function between dual-channel noisy speech and noise log-Mel features at the primary channel of the smartphone. This DNN-based method exploits the PLD between the two microphones of the device when employed in close-talk conditions to effectively provide accurate noise estimates. An illustration on how the DNN is used to this end can be seen in Figure 2.

First, let

$$\mathbf{y}_i(t) = (y_i(0,t), y_i(1,t), ..., y_i(\mathcal{M}-1,t))^{\mathrm{T}}, \quad i = 1, 2, \tag{1}$$

and

$$\mathbf{n}_1(t) = (n_1(0,t), n_1(1,t), ..., n_1(\mathcal{M}-1,t))^{\mathrm{T}} \tag{2}$$

respectively be noisy speech and noise log-Mel feature vectors at time frame t. These vectors are comprised of $\mathcal{M}$ frequency components where $\mathcal{M}$ is the total number of filterbank channels. Moreover, the subscript indicates the channel to which each vector belongs. Our DNN works on a frame-by-frame basis so that it gives a noise frame estimate at each time t from an input consisting of the dual-channel noisy speech observation at time t along with its temporal context. In particular, if we define

$$\mathbf{y}(t) = \begin{pmatrix} \mathbf{y}_1(t) \\ \mathbf{y}_2(t) \end{pmatrix}, \tag{3}$$

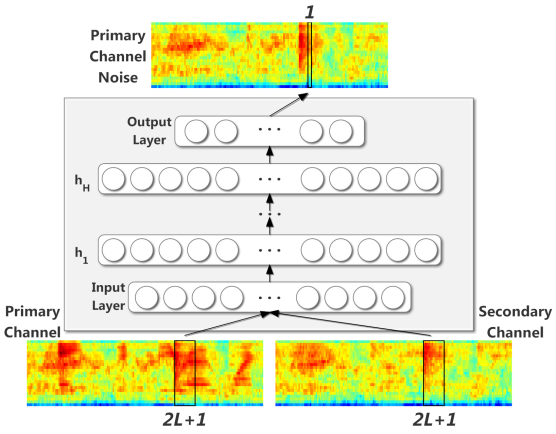

Fig. 2. An outline of the DNN as used for noise estimation purposes.

the DNN input vector becomes

$$\boldsymbol{\mathcal{Y}}(t) = \left(\mathbf{y}^{\mathrm{T}}(t-L)\ ,\ \ldots\ ,\ \mathbf{y}^{\mathrm{T}}(t+L)\right)^{\mathrm{T}}, \tag{4}$$

where the variable L determines the size of the temporal context considered (i.e. the size of the temporal window used is $2L+1$). Therefore, the dimension of the input vector is $\dim(\boldsymbol{\mathcal{Y}}(t)) = 2\mathcal{M}(2L+1)$. Additionally, as expected, the corresponding $\mathcal{M}$-dimensional target vector is that of Eq. (2).

The DNN training consists of an unsupervised generative pre-training, where it is considered each pair of layers as restricted Boltzmann machines (RBMs) [14], followed by a supervised fine-tuning step. The goal of this pre-training is to avoid getting stuck in plateaus or local minima during the fine-tuning phase because of the complex error surface as a result of the deep architecture [15]. Thus, the input and first hidden layers form a Gaussian-Bernoulli RBM (i.e. a visible layer of Gaussian variables connected to binary units in a hidden layer) since the input vector is real-valued. The successive pairs of layers form Bernoulli-Bernoulli RBMs (i.e. two layers with connections between their binary units). Input data are used to train the Gaussian-Bernoulli RBM and the inferred states of its hidden units are employed to train the following Bernoulli-Bernoulli RBM, and so on. The parameters resulting from this generative model consisting of the stack of RBMs are used to initialize the DNN, which is then fine-tuned by performing a supervised training by means of the backpropagation algorithm. For backpropagation learning the minimum mean square error (MMSE) criterion was chosen. Furthermore, the activation function type considered for the hidden layers is sigmoid while that is linear for the output layer, as could be expected for regression purposes.

The values chosen for the DNN hyperparameters and the rest of details about the DNN setup can be found in Subsection 3.2.

2.2 Noise-Aware Training

DNN noise-aware training (NAT) is a method first appeared in [16] to strengthen the DNN-based acoustic modeling for ASR. It basically consists of appending a noise estimate to the network's input vector containing the noisy speech features to improve word recognition rates when employing multi-style acoustic modeling. Since then, NAT has been successfully applied to different tasks such as, for instance, DNN-based speech enhancement [17]. In this work, we want to explore if the DNN-based noise estimation approach of Subsection 2.1 can be improved by increasing the awareness of the DNN about the noise that contaminates speech in each case.

A simple noise estimator, which has demonstrated to be quite accurate [18], consists of the linear interpolation between the averages of the first and last M frames of an utterance in the log-Mel domain. It should be pointed out that this method assumes that the first and last M frames in each utterance contain only noise energy. To imitate this method, and assuming that an utterance is T frames long, the initial input vector $\boldsymbol{\mathcal{Y}}(t)$ is augmented by appending the aforementioned averages,

$$\bar{\mathbf{n}}_1^{(0)} = \frac{1}{M}\sum_{t=0}^{M-1}\mathbf{y}_1^{\mathrm{T}}(t); \qquad \bar{\mathbf{n}}_1^{(1)} = \frac{1}{M}\sum_{t=T-M}^{T-1}\mathbf{y}_1^{\mathrm{T}}(t), \tag{5}$$

as well as a time index to indicate the frame's relative position within the utterance:

$$\tau(t) = t/(T-1). \tag{6}$$

Additionally, we decided to also use noise variance information by computing and appending the following sample quantities:

$$\boldsymbol{\sigma}_1^{(0)} = \frac{1}{M-1}\sum_{t=0}^{M-1}\left(\mathbf{y}_1^{\mathrm{T}}(t) - \bar{\mathbf{n}}_1^{(0)}\right)^2; \qquad \boldsymbol{\sigma}_1^{(1)} = \frac{1}{M-1}\sum_{t=T-M}^{T-1}\left(\mathbf{y}_1^{\mathrm{T}}(t) - \bar{\mathbf{n}}_1^{(1)}\right)^2, \tag{7}$$

where $(\cdot)^2$ is applied element-wise. Thus, the final DNN input vector is

$$\boldsymbol{\mathcal{Y}}_{NAT}(t) = \left(\boldsymbol{\mathcal{Y}}^{\mathrm{T}}(t),\ \bar{\mathbf{n}}_1^{(0)},\ \bar{\mathbf{n}}_1^{(1)},\ \boldsymbol{\sigma}_1^{(0)},\ \boldsymbol{\sigma}_1^{(1)},\ \tau(t)\right)^{\mathrm{T}}, \tag{8}$$

with dimension $\dim(\boldsymbol{\mathcal{Y}}_{NAT}(t)) = \dim(\boldsymbol{\mathcal{Y}}(t)) + 4\mathcal{M} + 1 = 4\mathcal{M}\left(L + \frac{3}{2}\right) + 1$.

3 Experimental Evaluation

The performance of the proposed dual-channel DNN-based noise estimator is evaluated in terms of word accuracy (WAcc) when used together with a VTS feature compensation method as depicted in Figure 1. Subsections 3.1 and 3.2 describe the recognition framework considered and how the DNN has been set up for experiments, respectively. Then, our DNN-based noise estimation results and a comparison with other noise estimation algorithms when they are also used together with a VTS feature compensation method are presented in Subsections 3.3 and 3.4, respectively.

3.1 Recognition Framework

The AURORA2-2C (AURORA2 - 2 Channels - Conversational Position) database first reported in [10] is employed for ASR experiments. The AURORA2-2C database is an extension to the well-known Aurora-2 database [19] that comprises the acquisition of noisy speech with a dual-microphone smartphone used in close-talk conditions. Two test sets, A and B, are defined in AURORA2-2C with different noise types in each one. The types of noise used in test set A are bus, babble, car and pedestrian street, while test set B considers the noises café, street, bus station and train station.

For VTS feature compensation we employ the first-order implementation reported in [18]. The only difference with respect to [18] consists of the use of different noise estimation algorithms. VTS is performed using a 256-component clean speech Gaussian mixture model (GMM) with diagonal covariance matrices. This GMM was obtained by performing the expectation-maximization (EM) algorithm on the same dataset as that used for clean acoustic model training.

To extract acoustic features from the speech signals, the European Telecommunication Standards Institute front-end (ETSI FE, ES 201 108) is used [20]. Log-Mel feature vectors employed by both the DNN and VTS are composed of $\mathcal{M} = 23$ frequency bins. Twelve Mel-frequency cepstral coefficients (MFCCs) along with the 0th order coefficient are obtained by application of the DCT to the enhanced log-Mel features. Then, their velocity and acceleration coefficients are appended to them to form the 39-dimensional feature vector employed by the recognizer. To strengthen the speech recognizer against channel mismatches cepstral mean normalization (CMN) is also applied.

Regarding the speech recognizer, both clean and multi-style acoustic models are considered for evaluation. The latter models are trained with distorted speech features to strengthen the ASR system against noisy conditions. In AURORA2-2C, the multi-style training dataset is created from the training clean utterances of Aurora-2 and consists of dual-channel utterances contaminated with the types of noise in test set A at the signal-to-noise ratios (SNRs) of 5 dB, 10 dB, 15 dB and 20 dB, along with the clean condition. Noisy utterances are compensated with VTS using the corresponding noise estimation algorithm prior to training the multi-style acoustic models. To model each digit, left to right continuous density hidden Markov models (HMMs) with 16 states and 3 Gaussians per state are used. Silences and short pauses are modeled by HMMs with 3 and 1 states, respectively, and 6 Gaussians per state [19].

3.2 DNN Setup

Taking into account the speech recognition task as well as the different noise conditions considered in this paper, for the sake of efficiency and to avoid data redundancy, our DNN was trained using 25600 sample pairs of input-target vectors. Training input data consisted of a mixture of samples contaminated with the noises of test set A at the SNRs of -5 dB, 0 dB, 5 dB, 10 dB, 15 dB and 20 dB. Thus, the noise types of test set B are useful to test the generalization capability of the DNN to unseen noise conditions during training. On the

one hand, for the unsupervised pre-training stage the number of epochs in each RBM was 40 and the learning rate was set to 0.0005. On the other hand, for the fine-tuning step the number of epochs was 100 and a learning rate of 0.1 was employed. The momentum rate used was 0.9. Following the tips from the Hinton's report in [21], the mini-batch size was 10 sample pairs. To improve the generalization capability of the network, early-stopping was adopted as a regularization strategy to avoid overfitting during training. Moreover, since the task addressed in [11] is similar to that in this paper, and assuming that noise has weak temporal correlations, L was set to 2 as in [11]. Since $\mathcal{M} = 23$, the input layer has $\dim(\boldsymbol{\mathcal{Y}}(t)) = 230$ and $\dim(\boldsymbol{\mathcal{Y}}_{NAT}(t)) = 323$ neurons for the DNN of Subsection 2.1 (without NAT) and 2.2 (with NAT), respectively. For both DNN configurations the output layer has $\mathcal{M} = 23$ neurons and five hidden layers were set up, according to preliminary recognition experiments, with 512 neurons each. For NAT, $M = 20$ was considered. Finally, the implementation of the DNN was done using Python along with the library Theano [22].

3.3 DNN-Based Noise Estimation Results

Tables 1 and 2 present a comparison in terms of WAcc between our DNN without NAT (DNN_2) and other techniques when employing clean and multi-style acoustic models, respectively. These reference techniques are a single-channel DNN-based noise estimator (DNN_1) as well as our previous approach reported in [11] (TGI+DNN), which shares similarities with this work. It should be noticed that the only difference between DNN_1 and DNN_2 is that Eq. (3) is redefined as $\mathbf{y}(t) = \mathbf{y}_1(t)$ for the former one. Baseline results are obtained by directly using the noisy speech features from the primary channel with no compensation. For both types of acoustic models, the best results are achieved by DNN_2, which makes it a better choice than TGI+DNN to provide robustness for ASR in dual-microphone smartphones. Also by a large margin (11.13% and 9.06% on average under clean and multi-style acoustic modeling, respectively) DNN_2 is clearly superior to DNN_1 as it exploits the information from the secondary channel, which is a good noise reference since speech is very attenuated in it as previously discussed. As expected, better WAcc results are generally obtained by employing multi-style instead of clean acoustic models, since the mismatch between training and test data is lower. In addition, test set B baseline results are substantially worse than those of test set A. Nevertheless, DNN_2 exhibits some generalization capabilities to noise conditions not seen during training. Finally, it is worth mentioning that word recognition results for both TGI+DNN and baseline are slightly different from those reported in [11]. This is because in this work the AURORA2-2C database was generated considering an anechoic chamber instead of a semi-anechoic environment for the acoustic path estimation, as in [11].

Table 3 shows the WAcc results achieved by both DNN_1 and DNN_2 when integrating the NAT approach of Subsection 2.2, DNN_1^{NAT} and DNN_2^{NAT}. On the one hand, DNN_1 has experienced an average relative improvement of 3.74% and 2.27% in terms of WAcc when employing clean and multi-style acoustic models,

respectively, by incorporating a noise reference by means of NAT. On the other hand, NAT degrades the performance of DNN_2. This could be explained because the secondary channel is a better noise reference itself than the information considered in our NAT-based approach, which introduces a greater uncertainty.

	Baseline			TGI+DNN			DNN_1			DNN_2		
SNR (dB)	Test *A*	Test *B*	Avg.	Test *A*	Test *B*	Avg.	Test *A*	Test *B*	Avg.	Test *A*	Test *B*	Avg.
-5	21.14	15.15	18.15	54.80	32.29	43.55	41.69	20.72	31.21	63.02	40.87	51.95
0	38.19	25.50	31.85	79.42	60.24	69.83	67.21	45.81	56.51	85.74	71.09	78.42
5	64.60	47.61	56.11	92.67	84.12	88.40	85.56	73.57	79.57	94.53	90.20	92.37
10	87.71	77.84	82.78	97.08	94.38	95.73	93.09	88.56	90.83	97.72	96.21	96.97
15	95.99	93.44	94.72	98.45	97.54	98.00	96.28	94.36	95.32	98.59	98.17	98.38
20	98.13	97.38	97.76	98.93	98.38	98.66	96.92	96.50	96.71	98.94	98.77	98.86
Clean	99.13	99.13	99.13	99.13	99.13	99.13	97.59	97.59	97.59	99.02	99.02	99.02
Avg. (-5 to 20)	67.63	59.49	63.56	86.89	77.83	82.36	80.13	69.92	75.03	**89.76**	**82.55**	**86.16**

Table 1. WAcc results (%) for DNN_2 and other comparison techniques when clean acoustic models are employed. Results are averaged across all types of noise in each test set.

	Baseline			TGI+DNN			DNN_1			DNN_2		
SNR (dB)	Test *A*	Test *B*	Avg.	Test *A*	Test *B*	Avg.	Test *A*	Test *B*	Avg.	Test *A*	Test *B*	Avg.
-5	47.64	26.22	36.93	57.89	35.75	46.82	48.14	24.10	36.12	67.05	44.37	55.71
0	76.99	56.39	66.69	82.18	65.88	74.03	72.91	51.93	62.42	87.95	75.02	81.49
5	92.36	85.33	88.85	94.12	87.15	90.64	89.29	78.73	84.01	95.39	91.57	93.48
10	96.94	94.52	95.73	97.50	95.02	96.26	95.19	91.96	93.58	97.93	96.84	97.39
15	97.98	97.14	97.56	98.46	97.41	97.94	97.59	96.04	96.82	98.60	98.34	98.47
20	98.49	98.12	98.31	98.85	98.02	98.44	98.16	97.68	97.92	98.79	98.65	98.72
Clean	98.77	98.77	98.77	98.61	98.61	98.61	98.49	98.49	98.49	98.99	98.99	98.99
Avg. (-5 to 20)	85.07	76.29	80.68	88.17	79.87	84.02	83.55	73.41	78.48	**90.95**	**84.13**	**87.54**

Table 2. WAcc results (%) for DNN_2 and other comparison techniques when multi-style acoustic models are employed. Results are averaged across all types of noise in each test set.

	Clean models						*Multi-style models*					
	DNN_1^{NAT}			DNN_2^{NAT}			DNN_1^{NAT}			DNN_2^{NAT}		
SNR (dB)	Test *A*	Test *B*	Avg.	Test *A*	Test *B*	Avg.	Test *A*	Test *B*	Avg.	Test *A*	Test *B*	Avg.
-5	41.93	22.70	32.32	52.85	33.04	42.95	46.73	26.40	36.57	58.21	38.17	48.19
0	71.02	54.18	62.60	78.76	63.17	70.97	75.01	60.24	67.63	82.77	69.69	76.23
5	90.03	82.11	86.07	92.06	86.74	89.40	91.91	85.33	88.62	93.80	88.99	91.40
10	96.62	93.61	95.12	96.70	94.51	95.61	96.73	94.32	95.53	97.44	95.36	96.40
15	98.32	97.37	97.85	98.23	97.34	97.79	98.24	97.15	97.70	98.30	97.56	97.93
20	98.82	98.49	98.66	98.84	98.40	98.62	98.63	98.29	98.46	98.72	98.31	98.52
Clean	98.83	98.83	98.83	98.60	98.60	98.60	98.89	98.89	98.89	98.74	98.74	98.74
Avg. (-5 to 20)	82.79	74.74	78.77	86.24	78.87	82.56	84.54	76.96	80.75	**88.21**	**81.35**	**84.78**

Table 3. WAcc results (%) for NAT when both clean and multi-style acoustic models are employed. Results are averaged across all types of noise in each test set.

3.4 A Comparison with other Noise Estimation Algorithms

To conclude our experimental evaluation, DNN_2, which has exhibited the best performance so far, is compared with different single-channel noise estimation algorithms when applied on the primary channel: Rangachari's algorithm (RANG) [4], improved minima controlled recursive averaging (IMCRA) [5], minimum statistics (MS) [6], MMSE-based noise estimation (MMSE) [7] and linear interpolation (INT) as described in Subsection 2.2 with $M = 20$. Furthermore, power level difference noise estimation (PLDNE) [8], which is a dual-channel noise estimation algorithm based on recursive averaging, is also tested. PLDNE is especially interesting since it is intended for dual-microphone smartphones

employed in close-talk conditions by assuming both a homogeneous diffuse noise field and that clean speech at the secondary channel is very attenuated with respect to the primary one. The corresponding WAcc results obtained when clean and multi-style acoustic models are used can be seen in Tables 4 and 5, respectively. As can be observed, on average and for all the SNRs considered but the clean case, DNN_2 shows the best performance among the noise estimation algorithms evaluated. In particular, thanks to the powerful regression capabilities of DNNs, DNN_2 is able to achieve a greater performance than PLDNE with no other assumptions than just exploiting the PLD between the two channels of the device.

Method / SNR (dB)	-5	0	5	10	15	20	Clean	Avg. (-5 to 20)
Baseline	18.15	31.85	56.11	82.78	94.72	97.76	**99.13**	63.56
RANG	38.15	64.35	83.30	92.22	95.42	96.40	96.48	78.31
IMCRA	35.07	63.38	83.60	93.04	96.99	98.21	99.01	78.38
MS	35.51	63.79	83.81	92.98	96.86	98.27	98.90	78.54
MMSE	38.87	66.27	85.57	93.65	97.14	98.33	99.08	79.97
INT	44.25	72.75	89.69	95.44	97.71	98.49	99.09	83.06
PLDNE	40.57	69.32	87.05	94.22	97.07	98.08	98.95	81.05
DNN_2	**51.95**	**78.42**	**92.37**	**96.97**	**98.38**	**98.86**	99.02	**86.16**

Table 4. Comparison between several noise estimation algorithms in terms of WAcc (%) when clean acoustic models are employed. Results are averaged across all types of noise in test sets *A* and *B*.

Method / SNR (dB)	-5	0	5	10	15	20	Clean	Avg. (-5 to 20)
Baseline	36.93	66.69	88.85	95.73	97.56	98.31	98.77	80.68
RANG	45.76	73.26	89.72	95.62	97.44	98.22	98.40	83.34
IMCRA	41.78	71.49	88.80	95.32	97.69	98.47	98.88	82.26
MS	49.65	71.71	88.96	95.49	97.75	98.53	98.89	83.68
MMSE	44.99	72.85	89.22	95.38	97.58	98.49	**99.01**	83.09
INT	47.98	76.07	91.22	96.00	97.91	98.49	98.79	84.61
PLDNE	48.11	77.70	92.68	96.65	98.05	98.53	98.71	85.29
DNN_2	**55.71**	**81.49**	**93.48**	**97.39**	**98.47**	**98.72**	98.99	**87.54**

Table 5. Comparison between several noise estimation algorithms in terms of WAcc (%) when multi-style acoustic models are employed. Results are averaged across all types of noise in test sets *A* and *B*.

4 Conclusions

In this paper we have presented a novel noise estimation method for noise-robust ASR in dual-microphone smartphones. In particular, a DNN has successfully been used to find a non-linear mapping function between dual-channel noisy speech and noise log-Mel features at the primary channel of the smartphone. Thanks to the powerful regression capabilities of DNNs, very high word recognition results have been obtained by just using simple features as well as exploiting the PLD between the two channels of the device when employed in close-talk conditions. As future work, we would like to explore the performance of this approach under different mobile devices with different small array configurations (e.g. with more than two microphones) as well as employed in arbitrary positions and not only in close-talk conditions.

References

1. Li, J., Deng, L., Gong, Y., Haeb-Umbach, R.: "An Overview of Noise-Robust Automatic Speech Recognition". *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 22, no. 4 (2014)
2. Moreno, P. J., et al.: "A Vector Taylor Series Approach for Environment-Independent Speech Recognition". *In: ICASSP*, pp. 733–736. Atlanta, USA (1996)
3. Wu, J., Droppo, J., Deng, L., Acero, A.: "A Noise-Robust ASR Front-End Using Wiener Filter Constructed from MMSE Estimation of Clean Speech and Noise". *In: ASRU*, pp. 321–326. Virgin Islands (2003)
4. Rangachari, S., Loizou, P. C.: "A Noise-Estimation Algorithm for Highly Non-Stationary Environments". *Speech Communication*, vol. 48 (2006)
5. Cohen, I.: "Noise Spectrum Estimation in Adverse Environments: IMCRA". *IEEE Trans. on Speech, and Audio Proc.*, vol. 11 (2003)
6. Martin, R.: "Noise Power Spectral Density Estimation Based on Optimal Smoothing and Minimum Statistics". *IEEE Trans. on Speech, and Audio Proc.*, vol. 9 (2001)
7. Hendriks, R. C., Heusdens, R., Jensen, J.: "MMSE Based Noise PSD Tracking with Low Complexity". *In: ICASSP*. Dallas, USA (2010)
8. Jeub, M., et al.: "Noise Reduction for Dual-Microphone Mobile Phones Exploiting Power Level Differences". *In: ICASSP*, pp. 1693–1696. Kyoto, Japan (2012)
9. Zhang, J., et al.: "A Fast Two-Microphone Noise Reduction Algorithm Based on Power Level Ratio for Mobile Phone". *In: ISCSLP*, pp. 206–209. Hong-Kong (2012)
10. López-Espejo, I., et al.: "Feature Enhancement for Robust Speech Recognition on Smartphones with Dual-Microphone". *In: EUSIPCO*. Lisbon, Portugal (2014)
11. López-Espejo, I., et al.: "A Deep Neural Network Approach for Missing-Data Mask Estimation on Dual-Microphone Smartphones: Application to Noise-Robust Speech Recognition". *Lecture Notes in Computer Science*, vol. 8854 (2014)
12. Wang, Y., Wang, D. L.: "Towards Scaling Up Classification-Based Speech Separation". *IEEE Trans. on Audio, Speech, and Language Proc.*, vol. 21, no. 7 (2013)
13. Vincent, E.: "Is Audio Signal Processing Still Useful in the Era of Machine Learning?". *In: WASPAA*. New York, USA (2015)
14. Hinton, G. E., Osindero, S., Teh, Y. W.: "A Fast Learning Algorithm for Deep Belief Nets". *Neural Computation*, vol. 18 (2006)
15. Hinton, G. E., Salakhutdinov, R.: "Reducing the Dimensionality of Data with Neural Networks". *Science*, vol. 313, no. 5786 (2006)
16. Seltzer, M. L., et al.: "An Investigation of Deep Neural Networks for Noise Robust Speech Recognition". *In: ICASSP*, pp. 7398–7402. Vancouver, Canada (2013)
17. Xu, Y., Du, J., Dai, L. R, Lee, C. H.: "A Regression Approach to Speech Enhancement Based on Deep Neural Networks". *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 23, no. 1 (2015)
18. Segura, J. C., et al.: "Model-Based Compensation of the Additive Noise for Continuous Speech Recognition. Experiments Using the AURORA II Database and Tasks". *In: EUROSPEECH*. Aalborg, Denmark (2001)
19. Pearce, D., Hirsch, H. G.: "The Aurora Experimental Framework for the Performance Evaluation of Speech Recognition Systems Under Noisy Conditions". *In: ICSLP*. Beijing, China (2000)
20. ETSI ES 201 108 - *Distributed Speech Recognition; Front-End Feature Extraction Algorithm; Compression Algorithms*
21. Hinton, G. E.: "A Practical Guide to Training Restricted Boltzmann Machines". *UTML TR 2010-003* (2010)
22. Theano Library - *http://deeplearning.net/software/theano/*