BUT ASR SYSTEM FOR BABEL SURPRISE EVALUATION 2014

Martin Karafiát, Karel Veselý, Igor Szoke, Lukáš Burget, František Grézl, Mirko Hannemann, Jan Černocký

Brno University of Technology, Speech@FIT and IT4I Center of Excellence, Brno, Czech Republic

karafiat,grezl,vesely,ihannema,szoke,cernocky,burget@fit.vutbr.cz

ABSTRACT

The paper describes Brno University of Technology (BUT) ASR system for 2014 BABEL Surprise language evaluation (Tamil). While being largely based on our previous work, two original contributions were brought: (1) speaker-adapted bottle-neck neural network (BN) features were investigated as an input to DNN recognizer and semi-supervised training was found effective. (2) Adding of noise to training data outperformed a classical de-noising technique while dealing with noisy test data was found beneficial, and the performance of this approach was verified on a relatively clean training/test data setup from a different language. All results are reported on BABEL 2014 Tamil data.

Index Terms— speech recognition, discriminative training, bottle-neck neural networks, deep neural networks, adaptation of neural networks, noisy speech

1. INTRODUCTION

This paper presents our recent effort to build an Automatic Speech Recognition (ASR) system for Tamil OpenKWS 2014 Babel evaluations.

In the 2nd year of the BABEL project, teams had one year to build systems for five languages (Assamese, Bengali, Haitian Creole, Zulu and Lao) and this effort culminated by one-month evaluation. Immediately after, a "surprise" language was released and all participating teams had *three weeks* to build a system. The surprise language was Tamil.

Three weeks are barely enough to run all the system training and produce the evaluation results. Therefore, our ASR is our usual combination of Gaussian-Mixture Model (GMM) and Deep Neural Network (DNN) ASR systems with features based on a hierarchy of NNs with bottle-neck (BN) layers. Our previous papers [1, 2, 3, 4, 5, 6] provide full account of this development. In this paper, section 2 summarizes the baseline system, section 3 outlines our stacked-bottle-neck (SBN) feature extraction and 4 describes the full GMM system including its training.

The original contributions of our work towards Tamil ASR were two: first, we improved our DNN system by working with adapted features (see section 5). Another important piece of work targeted the noise present in the evaluation data that was new for Tamil. We found that augmenting the training data by versions of this same data corrupted by artificially generated noises led to a significant increase in robustness. This approach was analyzed also on clean data from a different language where an improvement of accuracy was shown too (section 6).

LLP training hours	23h + 157h untrans. 11.9h + 68.7h after VAD
FLP training hours	138h + 42h untrans. 69.3h + 20h after VAD
LLP LM training words FLP LM training words	79688 474722
LLP dictionary size FLP dictionary size	16271 58039
LLP LM - OOV rate [%] FLP LM - OOV rate [%]	16.8% 9.6%
dev hours	20h (10h VAD)

Table 1. Tamil data statistics.

2. DATA AND SYSTEM

2.1. Data

The IARPA Babel Program data[1] simulates a case of what one could collect in limited time from a completely new language. For each language, two main conditions are defined:

1. Full Language Pack (FullLP or FLP) - about 70h of transcribed speech plus 20h of un-transcribed speech.

Supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense US Army Research Laboratory contract number W911NF-12-C-0013. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government. The work was also partly supported by, Technology Agency of the Czech Republic grant No. TA01011328 and by IT4Innovations Centre of Excellence CZ.1.05/1.1.00/02.0070. M. Karafiat was supported by Grant Agency of the Czech Republic post-doctoral project No. P202/12/P604.

[1] Collected by Appen http://www.appenbutlerhill.com

2. Limited Language Pack (LimitedLP or LLP) - A subset of FullLP data (about 10h of transcribed clean speech). The remaining "unseen" part of the FLP data can be used for unsupervised training.

LLP is Babel's primary condition, therefore, all results in this paper are reported with systems build on LLP data (extended later by unsupervised data).

The data consists of three parts: scripted (speakers read text through telephone channel) and spontaneous. The spontaneous part was recorded mainly from telephone conversations and partly from wide-band far-field microphones. The *dev* data contains conversational speech from both channels. Table 1 summarizes the statistics of available data. A pronunciation dictionary and Language Model (LM) were also defined with respect to the Language Pack.

2.2. Baseline ASR system and features

Our baseline speech recognition system is HMM-based on cross-word tied-states triphones; it is trained from scratch using standard maximum likelihood training. The final word transcriptions are decoded using 3-gram Language Model (LM) trained only on the transcriptions of the training data.[2]

Our features are Mel-PLPs augmented with deltas, double- and triple-deltas, so that the feature vector has a dimensionality of 52. Cepstral mean and variance normalization is applied with the means and variances estimated per conversation side. HLDA is estimated with Gaussian components as classes to reduce the dimensionality to 39.

3. STACKED BOTTLE-NECK NN FEATURE EXTRACTION

The architecture used to generate our advanced features was Stacked Bottle-Neck (SBN) NN, which was found to outperform standard Bottle-Neck features [1]. The scheme of SBN feature extraction is given in figure 1. It contains two NNs: the BN outputs from the first one are adapted [6], stacked over 21 frames, down-sampled and taken as an input vector for the second NN. This second NN has again a BN layer, of which the outputs are taken as input features for a GMM-HMM recognition system.

The NN input features are 24 log Mel Filter band energies concatenated with fundamental frequency (F0) features produced by three different estimators: BUT F0 detector produces 2 coefficients (F0 and probability of voicing), Snack F0 gives just a single F0 and Kaldi F0 estimator outputs 3 coefficients (Normalized F0 across sliding window, probability of voicing and delta). Fundamental frequency variation (FFV) estimator [7] produces a 7-dimensional vector. Therefore, the whole feature vector has 24+2+1+3+7=37 coefficients. The

[2]This is coherent to BABEL rules, where *the provided data only* can be used for system training in the primary condition.

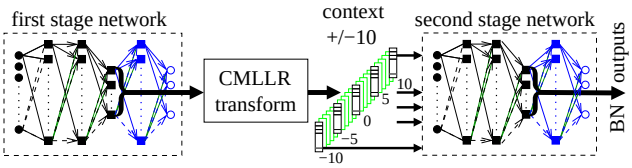

Fig. 1. Stacked Bottle-Neck Neural Network feature extraction.

positive effect of using more F0 estimators is described in our previous work [6].

Conversation-side based mean subtraction is applied on the speaker basis and 11 frames are stacked together. Hamming window followed by DCT consisting of 0th to 5th base are applied on the time trajectory of each parameter resulting in 37×6=222 coefficients at the first-stage NN input (see fig. 1).

The first-stage NN has four hidden layers with 1500 units each except the BN layer. The BN layer is the third hidden layer and its size is 80 neurons. Its outputs are stacked over 21 frames (+/-10) and down-sampled (every 5th is taken) before entering the second-stage NN. The second-stage NN has the same structure and sizes of hidden layers as the first one. The size of BN layer is 30 neurons and its outputs are the final BN features for the recognition system. Neurons in both BN layers have linear activation functions as they were reported to provide better performance [8] than classical sigmoids. The NN targets are triphone states obtained by forced alignment of training data. To train the system on Bottle-Neck features, the BN outputs are transformed by Maximum Likelihood Linear Transform (MLLT), which considers HMM states as classes.

4. FULL GMM SYSTEM

The final system is based on feature-level fusion by Region Dependent Transform (RDT) [9]. Three feature streams — PLP-HLDA (39 dimensions), SBN features (30 dim.) and BUT F0 with delta and acceleration coefficients (3 dim.) — are concatenated which results in 72-dimensional feature stream (called PLP-NN-F0 in the following text). Then, new models are trained by single-pass retraining from the basic PLP system. 12 Gaussian components per state were found to be sufficient for these features. The resulting models and PLP-NN-F0 features serve as a starting point for RDT training.

In RDT framework, an ensemble of linear transformations is trained with the discriminative Minimum Phone Error (MPE) criterion. Each transformation corresponds to one region in feature space partitioned by a GMM.

Our RDT settings performs dimensionality reduction from 72 to 69 which gives us better convergence than RDT without dimensionality reduction. The final GMM system was trained using MPE [10] on top of RDT features. This

system is denoted later in text as MPE-RDT or MPE-SAT-RDT, depending on used adaptation technique.

The whole SBN system training can be described as follows:

1. Initial ML PLP models are used to estimate PLP-HLDA transform and to generate triphone state targets for NN training.

2. SBN Neural Net and the whole MPE-RDT system is trained and used as a First-Pass system for speaker adaptation purposes.

3. The SBN NN is cut after the First Stage NN and this 80-dimensional feature stream is adapted by speaker-based Constrained Maximum Likelihood Regression CMLLR (further denoted as BN1-CMLLR features). Consequently, the Second Stage NN is re-trained in SAT fashion [5].

4. The new adapted SBN features generate new PLP-SATNN-F0 feature stream which is further speaker-adapted by CMLLR. Further, RDT system is also trained in SAT fashion.[3]

This system is used to generate unsupervised transcripts of un-transcribed data (the rest of FLP set) and also to rebuild triphone state NN targets on training data. This procedure was found effective especially for tonal languages because the initial targets are generated with plain PLP only [5]. Therefore, further system building runs with additional data and also with more accurate NN targets.

4.1. Semi-supervised training (SST)

SST is performed on LLP's un-transcribed data. Transcriptions are generated automatically. This again simulates a real scenario of a user that managed to collect data but transcribe only its small portion.

The automatic transcriptions are naturally erroneous due to many reasons (imperfect acoustic model, OOVs or poor language model). Thus, it is important to select sentences with reasonable transcription. We use utterance-level confidence defined as a weighted average of non-silence word confidences in the segment: $C_{utt} = \frac{1}{T}\sum_{w=1}^{W} t^w C^w_{max}$, where W is number of words, C^w_{max} is word confidence measure [11], t^w is length of the word in frames and T is length of all non-silence words.

The $(C_{max} > 0.5)$ rule was used in our experiments as it was found [4] to be a safe value for the NN training. With 0.5 threshold, about 70% of un-transcribed data is retained for the training.

[3]Note, that our previous experiments showed a marginal effect of VTLN on the PLP feature stream if CMLLR was used, therefore, VTLN was not applied for simplicity.

Then, the SBN architecture was trained on concatenated (transcribed + automatically transcribed and retained) data. It was further "fine-tuned" into transcribed data only. Learning rate for the "fine-tuning" was set to one tenth of its original value. Only the second stage NN is tuned to keep the training process simple and fast [6].

5. DNN SYSTEM

Besides the GMM-HMM system described in the previous sections, we have also built a hybrid DNN-HMM baseline system. The DNN training followed recipe described in [12] including semi-supervised training (SST). Baseline DNN input features were based on PLPs augmented with KaldiF0. These features were mean/variance normalized, spliced by +/- 4 frames next to the central frame and projected down to 40 dimensions using linear discriminant analysis (LDA) and MLLT. Moreover, speaker adaptive training (SAT) was done using a single feature-space MLLR transform estimated per speaker. These features are noted later in the text as a PLP-CMLLR. These PLP-CMLLR features were spliced using context of 11 (+/- 5) frames and globally mean and variance normalized.

For all the experiments, we used the same DNN topology: 440 inputs (11x40) and 6 hidden layers where each hidden layer has 2048 neurons with sigmoids. The hidden layers of DNN were initialized with stacked Restricted Boltzmann Machines (RBMs) that were pre-trained in a greedy layer-wise fashion [13]. After pre-training, we added the output layer with random weights and we performed frame-classification training (we classify frames into triphone tied-states). We used mini-batch Stochastic Gradient Descent (SGD) to minimize per-frame cross-entropy between the labels and network outputs. Finally, the seed network was re-trained by sequence-discriminative training by optimizing sMBR objective. This aimed to maximize expected frame accuracy of being in the correct state. The expectation was calculated over possible state sequences represented by lattices. The reference sequences were obtained by force-alignment to transcription.

This DNN was used as a seed system for further semi-supervised training. Unlabeled training data were automatically transcribed and per-frame confidences were produced. The confidences were based on re-scaled lattice posteriors that were selected by the best-paths' state-sequence from the lattice [12].

The semi-supervised training is applied to per-frame Cross-Entropy training only, for the sMBR training, the transcribed LLP data-set was used. Large part of the improvement from the per-frame Cross-Entropy semi-supervised training was preserved also after the sMBR training.

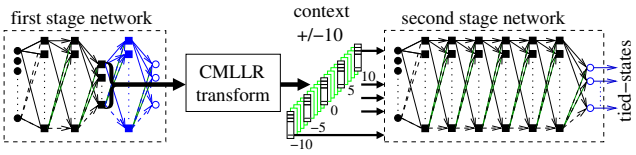

Fig. 2. DNN system architecture.

DNN features:	DNN training			
	10h		SST	
	Xent	sMBR	Xent	sMBR
PLP-CMLLR	74.9	72.1	-	-
BN1-CMLLR	73.8	71.9	72.1	70.4
BN1-CMLLR,SST	71.7	**69.9**	69.9	**68.3**

Table 2. Semi-supervised training of DNN.

5.1. BN-CMLLR features in DNN system

Finally, we extended DNN system by proposed first stage BN features with speaker adaptation (BN1-CMLLR). These features were stacked in context of 21 (+/-10) frames, downsampled by factor 5 (like in SBN) and fed into DNN system. Figure 2 presents this architecture.

Our previous work [6] presented excellent gains by using this architecture. Table 2 shows that baseline PLP-CMLLR features are giving close performance to BN1-CMLLR features if these are not SST trained. When the SST training is applied to input features, the gains are about 2% absolute regardless whether SST training is applied to the final DNN classifier or not — gains from SST training of DNN are additive to SST training of input features, which is a very positive finding.

6. DEALING WITH NOISE

The Tamil data were found to contain lots of noise mainly in wide-band channels which led to significantly worse performance than on clean data. Two main approaches were investigated: noise suppression and adding noise into the training data.

Many noise suppression techniques have been described, such as Spectral Subtraction (SS) [14], Wiener filtering [15], subspace techniques [16] and others. Wiener filtering was chosen for our experiments as it is the most popular in speech enhancement. The basic principle is to obtain a clean signal back from that corrupted by additive noise. It is required to estimate an optimal filter for the noisy input speech by minimizing Mean Square Error between desired signal s and estimated signal $\hat{s}$ (see [15, 17] for details).

The performance of noise suppression methods under real conditions with highly non-stationary or unpredictable noises may be limited especially in cases of limited amounts of training. Therefore, we experimented with cloning of training data and corrupting its versions by noise. Using one exact type of noise known from a few samples of data can lead to decrease of robustness on unknown distortions. Therefore, we artificially generated more types of noises (mostly by passing a white noise through various kinds of filters), hoping that it will lead to more general solution. Eight filters (see figure 3 for frequency characteristics) were defined:

- **v1** - **v4**: low pass filters; **v1** is close to the noise characteristics found in the dev data.
- **v5**, **v6**: one band-pass filter.
- **v7**, **v8**: several narrow band-pass filters.

Noises **v9** and **v10** are 100Hz and 50Hz sinusoidal hums. The noise was added to the original clean data on 3 amplitude levels:

- **L1** 0.35 of Root Mean Square (RMS) of the original signal (SNR 9 dB).
- **L2** 1.00 of RMS of the original signal (SNR 0 dB).
- **L3** 3.5 of RMS of the original signal (SNR -11 dB).

The amount of training data was multiplied by factor 4 (original data + 3 levels of noise). The noise type was selected randomly for each particular pair of utterance and noise level.

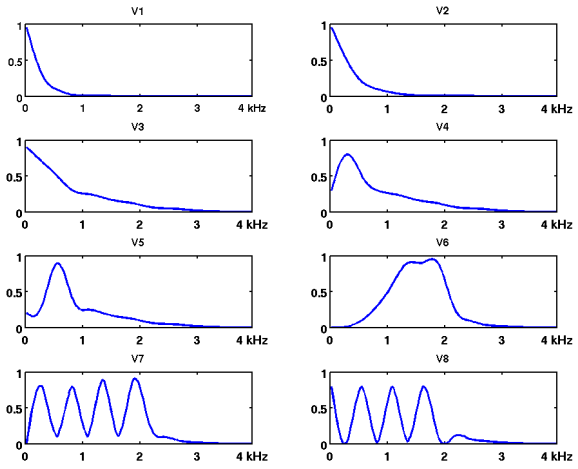

Fig. 3. Frequency characteristics of noise filters.

6.1. Analysis on simple SBN system

All features (PLP, FBANK, F0) were re-generated on noised waveforms but the NN training targets were cloned from clean-data alignments. The baseline system for these experiments was a simple ML-trained Stacked Bottle-Neck tandem system built on clean data only to see solely the effect of SBN

System	%WER
Baseline	76.0
Wiener filter on train/test	76.1
Additive noise in training data	**75.4**

Table 3. Noise suppression vs. noising of training data.

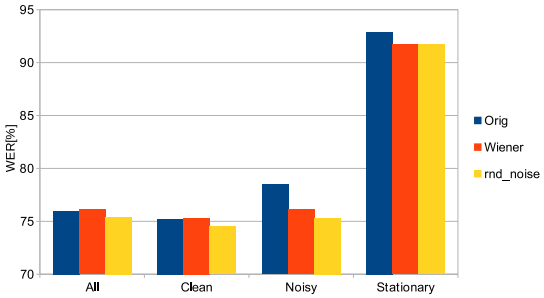

Fig. 4. Noise suppression vs. noising – details.

training. Table 3 presents slight degradation when Wiener filter was used. It could be caused by dominance of clean data in the test — the noise suppression techniques are know to degrade the performance on clean data due to adding distortion and artifacts. On contrary, adding noises into the data for SBN training is giving 0.6% absolute improvement against baseline clean condition.

For more detailed analysis, the test set was split into four categories:

- Clean - Signal to Noise Ratio (SNR) higher than 30 (1147 minutes of audio).
- Noisy - SNR bellow 30 (10 minutes of audio). Most of the recordings contain telephone speech with background noise like another speaker, television/radio, etc.
- Stationary - most corrupted data coming from "Far-field" microphones (49 minutes of audio).

Figure 4 shows that Wiener filter is effective on noise data only whereas adding noise to the training data is effective on both clean/noise data.

Adding the noise is an ad-hoc technique, therefore we were interested in analyzing the importance of each particular noise type. Ten noise-specific NNs were trained on clean plus three levels of single type of noise. Figure 5 shows that **v1** and **v2** are performing the best, probably due to noise characteristics close to the actual data, but none of the NNs shows an erratic behavior. Even hums, **v9, v10**, which are quite improbable to be in the data, are performing better than the baseline. It seems that the main effect of cloning and noising the data is increasing system robustness to various distortions.

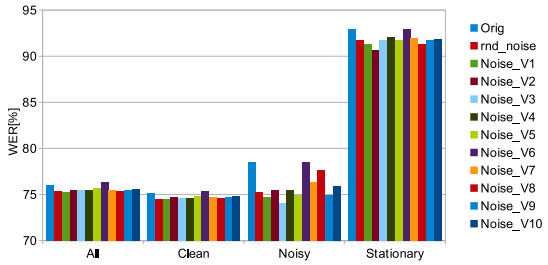

Fig. 5. Noise-specific NNs.

6.2. Final results

Table 4 presents the final ASR results obtained from the SST GMM system. It is obvious that the gain coming from noises can be doubled by re-training not only the feature extractor but also the whole system on noised data.

System	%WER
Baseline	72.1
Added noises to NN only	71.7
Added noises also to RDT-GMM	**71.2**

Table 4. Final GMM system results.

Similar outcome can be also seen on the final DNN-HMM system (table 5). It was semi-supervised trained on adapted PLP + BN1-CMLLR features with sMBR criterion. The main effect is coming from retraining DNN feature extractor on noised data (1.3% absolute gain); retraining of the whole DNN system also on noise data had 0.4% additive gain only.

System	%WER
BN1-CMLLR on clean data, DNN on clean only	69.4
BN1-CMLLR on noised data, DNN on clean only	68.1
BN1-CMLLR on noised data, DNN on noised data	**67.7**

Table 5. Final DNN system results.

6.3. Re-visiting 1st year languages - Cantonese

Adding noises to the training seems to increase the robustness of ASR against various distortions. We believed it should be also helpful on standard telephone speech recorded in clean conditions. The first BABEL language, Cantonese, was revisited for this purpose. The details about the data can be found in [2], the system was rebuilt on LLP data according to our current recipe (this paper) with and without noise. Table 6 shows a small 0.2% absolute gain obtained by training simple ML GMM on top of SST SBN NN on noised features (similar experiment as in Table 3). Next, we SST trained full MPE-RDT GMM system where 0.6% absolute gain was observed by adding a noised data (similar experiment as in Table 4).

System	ML SBN %WER	MPE-RDT system %WER
Baseline	54.5	48.2
+ noise	**54.3**	**47.6**

Table 6. Cantonese ML SBN and complete GMM system trained on noised data.

7. CONCLUSION

This paper presented a summary of heroic three weeks' work of our team on Tamil OpenKWS Surprise evaluation. We investigated speaker-adapted BN features as an input to DNN recognizer where semi-supervised training was found effective. Adding noise into training data was found superior to de-noising the data in NN-based feature extraction. It does not degrade performance for clean speech and it performs better on noisy speech. Moreover, further retraining of the whole ASR system on noised data led to significant increase of final system performance. All these partial improvements led to a very successful system in this evaluation.

The effect of adding noise to training data was verified on relatively clean training/test data setup from a different language. Data noising had positive effect also in this condition.

8. REFERENCES

[1] Frantisek Grezl, Martin Karafiat, and Lukas Burget, "Investigation into bottle-neck features for meeting speech recognition," in *Proc. Interspeech 2009*, 2009, number 9, pp. 2947–2950.

[2] Martin Karafiát, František Grézl, Mirko Hannemann, Karel Veselý, and Jan "Honza" Černocký, "BUT BABEL System for Spontaneous Cantonese," in *Proceedings of Interspeech 2013*, 2013, pp. 2589–2593.

[3] Karel Veselý, Mirko Hannemann, and Lukáš Burget, "Semi-supervised training of deep neural networks," in *Proc. of ASRU 2013*, Dec 2013.

[4] Grezl F., Karafiat M., and Vesely K., "Adaptation of neural network feature extractor for new language," in *in Proceedings of ASRU 2013*, Olomouc, Czech Republic, 2013.

[5] Martin Karafiát, František Grézl, Mirko Hannemann, and Jan "Honza" Černocký, "BUT neural network features for spontaneous vietnamese in BABEL," in *Proceedings of ICASSP 2014*, Florence, Italy, May 2014, IEEE.

[6] Martin Karafiát, František Grézl, Mirko Hannemann, Karel Veselý, Igor Szoke, and Jan "Honza" Černocký, "BUT 2014 Babel system: Analysis of adaptation in NN based systems," in *Proceedings of Interspeech 2014*, Singapure, September 2014, IEEE.

[7] Kornel Laskowski, Mattias Heldner, and Jens Edlund, "The fundamental frequency variation spectrum," in *in Proc. FONETIK*, 2008.

[8] Karel Veselý, Martin Karafiát, and František Grézl, "Convolutive bottleneck network features for LVCSR," in *Proceedings of ASRU 2011*, 2011, pp. 42–47.

[9] Bing Zhang, Spyros Matsoukas, and Richard Schwartz, "Recent progress on the discriminative region-dependent transform for speech feature extraction," in *Proc. of Interspeech 2006*, Pittsburgh, PA, USA, Sep 2006, pp. 2977–2980.

[10] D. Povey, *Discriminative training for large vocabulary speech recognition*, Ph.D. thesis, University of Cambridge, 2003.

[11] Frank Wessel, Ralf Schlüter, Klaus Macherey, and Hermann Ney, "Confidence measures for large vocabulary continuous speech recognition," *IEEE Transactions on Speech and Audio Processing*, vol. 9, pp. 288–298, 2001.

[12] Karel Veselý, Mirko Hannemann, and Lukáš Burget, "Semi-supervised training of deep neural networks," in *Proceedings of ASRU 2013*, 2013, pp. 267–272.

[13] G E Hinton, S Osindero, and Y Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, pp. 1527–1554, 2006.

[14] Bagher BabaAli, Hossein Sameti, and Mehran Safayani, "Likelihood-maximizing-based multiband spectral subtraction for robust speech recognition," *EURASIP J. Adv. Signal Process*, vol. 2009, pp. 12:1–12:15, Jan. 2009.

[15] Lim J.S. and Oppenheim A.V., *Enhancement and Bandwidth Compression of Noisy Speech*, Technical note. Lincoln Laboratory, M.I.T., 1979.

[16] Yi Hu and Philipos C. Loizou, "A subspace approach for enhancing speech corrupted by colored noise.," in *ICASSP*, 2002, pp. 573–576.

[17] Marwa A. Abd El-Fattah, Moawad I. Dessouky, Alaa M. Abbas, Salaheldin M. Diab, El-Sayed M. El-Rabaie, Waleed Al-Nuaimy, Saleh A. Alshebeili, and Fathi E. Abd El-Samie, "Speech enhancement with an adaptive Wiener filter," *Int. J. Speech Technol.*, vol. 17, no. 1, pp. 53–64, Mar. 2014.