Aalborg Universitet

AALBORG UNIVERSITY

Performance Comparison of AR Codebook Training for Speech Processing

Cui, Zihao; Bao, Changchun; Nielsen, Jesper Kjær; Christensen, Mads Græsbøll

Published in:
2020 15th IEEE International Conference on Signal Processing (ICSP)

DOI (link to publication from Publisher):
10.1109/ICSP48669.2020.9320929

Publication date:
2020

Document Version
Early version, also known as pre-print

Link to publication from Aalborg University

Citation for published version (APA):
Cui, Z., Bao, C., Nielsen, J. K., & Christensen, M. G. (2020). Performance Comparison of AR Codebook Training for Speech Processing. In *2020 15th IEEE International Conference on Signal Processing (ICSP)* (pp. 131-135). Article 9320929 IEEE Signal Processing Society. https://doi.org/10.1109/ICSP48669.2020.9320929

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal -

Take down policy
If you believe that this document breaches copyright please contact us at vbn@aub.aau.dk providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from vbn.aau.dk on: December 05, 2025

Performance Comparison of AR Codebook Training for Speech Processing

Zihao Cui[1], Changchun Bao✉[1], Jesper Kjær Nielsen[2] and Mads Græsbøll Christensen[2]
[1] Speech and Audio Signal Processing Lab, Faculty of Information Technology
Beijing University of Technology, Beijing, China
cuizhao@emails.bjut.edu.cn, baochch@bjut.edu.cn
[2] Audio Analysis Lab, CREATE, Aalborg University, Aalborg, Denmark
{jkn,mgc}@create.aau.dk

***Abstract*—In this paper, different ways of training codebook containing autoregressive (AR) parameter vectors are discussed. The fundamental goal of the discussion is to investigate if the classical approach for training AR-codebooks by clustering line spectral frequencies (LSF) can be improved. To do this, we discuss and evaluate the alternatives in terms of the de-correlated AR-parameters and manifold learning. The different training methods are evaluated using different metrics quantifying the distance between actual power spectral density (PSD) and the estimated PSD from the AR-codebook. The experimental results show that the training on the de-correlated features can improve the performance to some degree compared to the traditional LSF training approach in terms of the Itakura-Saito divergence not in terms of the Kullback-Leibler divergence, the log-spectral distortion and speech distortion.**

Keywords—linear prediction, Cramer-Rao bound, manifold learning, (weighted) k-means clustering, AR model

I. INTRODUCTION

The codebook consisting of AR [1] coefficients has been widely used for designing vector quantizers [2], speech enhancement [3], speech coding [4]–[6] and speech recognition [7]. Typically, the AR-codebook is trained by clustering line spectral frequencies (LSF) using a k-means clustering algorithm [2]. This approach has been extremely popular since the LSF coefficients are easily checked for stability and are much less sensitivity to quantization errors [8] than the AR coefficients [9]. However, clustering the LSF parameters directly potentially leads to a suboptimal AR-codebook since the LSF may not meet the assumption of classical AR model [10], namely, the excitation may not be a Gaussian distribution. Therefore, we have to discuss and compare alternative approaches for training AR codebook.

In addition to the classical LSF-based approach using k-means clustering, we here consider a number of alternatives to train an AR-codebook. These are k-means clustering on the raw and de-correlated AR-parameters [10] as well as k-means clustering on the raw and de-correlated features extracted from non-linear manifolds produced by sparse manifold clustering and embedding (SMCE) algorithm [11]. The main idea behind performing de-correlation prior to the clustering or, equivalently, to perform weighted k-means clustering, is that a more efficient representation of the training data is obtained. The de-correlation can be performed efficiently using the Fisher information matrix (FIM) [10]. Unfortunately, this de-correlation is not robust to outliers so that the clustering centers might be quite sensitive to the de-correlation. This problem was mitigated by the application of the SMCE method [11], which is more robust to those outliers. Furthermore, with the increasing of the dataset, the mini-batch k-means algorithm can be utilized to solve memory requirements.

The paper is organized as follows: In Section 2, we describe the computation of different features, which can be used for training AR-codebook. These features include AR-parameters, LSF-parameters as well as manifold vectors. In addition to these features, we also describe how can the weights be computed so that a weighted k-means clustering algorithm implicitly performs this de-correlation. The weighted k-means clustering algorithm is then described in Section 3, and the performance of the different AR-codebook training methods are evaluated in Section 4. Finally, Section 5 concludes the paper.

II. FEATURES AND WEIGHTS

To explain different concepts and the evaluation procedure used in this paper, we will use the illustration in Fig. 1 to show both the training and the testing stages. In the training stage, the features as well as weights are extracted from the training data. Note that the weights in some methods are simple identity matrix. These features and weights are then used as input to a clustering approach that produces M cluster centers form a codebook in the feature domain. If the feature domain is not the AR-parameter domain, the codebook is converted into an AR-codebook in the final step in the training stage. In the testing stage, the entry from the AR-codebook whose spectrum minimizes the Itakura-Saito (IS)-divergence [12] to the periodogram for a testing segment is selected. Using the spectrum corresponding to this selected entry from the codebook, the performance of the AR-codebook is then measured using a number of common metrics used in speech processing shown in Table 1.

In this section, we will describe the different ways of training AR codebook in the testing stage, and we will cover both the traditional approach based on LSF parameters as well as the alternative approaches discussed in the introduction. First, however, we will briefly describe the autoregressive process.

A. Autoregressive Processes

An AR process is a stationary random signal $x(n)$ given by

$$x(n) = -\sum_{i=1}^{p} a_i x(n-i) + e(n) \tag{1}$$

where a_i is the i'th out of p AR parameters and $e(n)$ is a white and Gaussian excitation signal with variance σ^2. The AR-coefficient vector can be expressed as

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

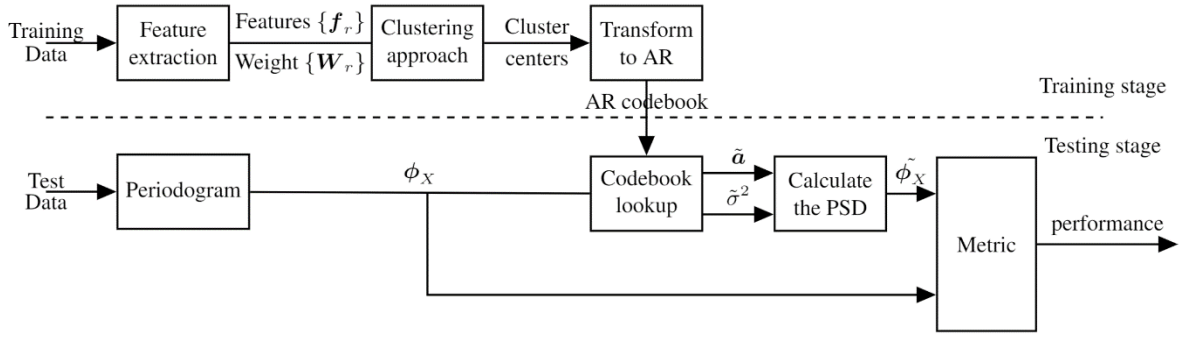

Fig. 1. The training stage and Example of a figure caption.

TABLE I. SOME COMMON METRICS USED IN SPEECH PROCESSING

Metric	Formula
LSD	$\frac{1}{K}\sum_{k=1}^{K}\left[\log_{10}\frac{\phi_X(k)}{\tilde{\phi}_X(k)}\right]^2$
IS	$\frac{1}{K}\sum_{k=1}^{K}\left[\frac{\phi_X(k)}{\tilde{\phi}_X(k)}-\ln\frac{\phi_X(k)}{\tilde{\phi}_X(k)}-1\right]$
KL	$\frac{1}{K}\sum_{k=1}^{K}\left[\phi_X(k)\ln\frac{\phi_X(k)}{\tilde{\phi}_X(k)}-\phi_X(k)+\tilde{\phi}_X(k)\right]$
SD	$\frac{1}{K}\sum_{k=1}^{K}\left[\phi_X(k)-\tilde{\phi}_X(k)\right]^2$

$$\boldsymbol{a}=[a_1 \quad \cdots \quad a_p]^T \tag{2}$$

and the excitation variance σ^2 can be estimated in many different ways, but the classical and computationally efficient approach is to solve the Yule-Walker equations using the Levension-Durbin recursion (LDR) [13]. An AR-codebook consists of M AR-coefficient vectors

$$\boldsymbol{A}=[\boldsymbol{a}_1 \quad \cdots \quad \boldsymbol{a}_M]\ , \tag{3}$$

with each vector being a cluster center representing typical AR coefficients in a neighbourhood around this cluster center. Generally, the spectral shape or power spectral density (PSD) $\boldsymbol{\phi}_X$ of an AR-process is given by

$$\phi_X(k)=\frac{\sigma^2}{\left|A_p(\mathrm{e}^{j\omega_k})\right|^2}\ ,\quad k=0,\ldots,N-1\ . \tag{4}$$

where the prediction error filter $A_p(z)$ is a polynomial given by

$$A_p(z)=1-\sum_{i=1}^{p}a_i z^{-i} \tag{5}$$

Note that the excitation variance is typically not a part of the AR-codebook.

B. LSF Parameters

Clustering the AR coefficients directly is often not a good idea since there is no easy way of ensuring that the cluster centers corresponds to stable AR processes, there is no simple relationship between the AR coefficients and the AR spectrum, and the dynamic range of the parameters is typically quite high. Instead, the line spectral frequencies (LSF) [8] have been used as an alternative representation of the AR coefficients. The LSF parameters are related to the zeros of the two polynomials

$$P_p(z)=A_p(z)+z^{-(p+1)}A_p(z^{-1}) \tag{6}$$

$$Q_p(z)=A_p(z)-z^{-(p+1)}A_p(z^{-1}) \tag{7}$$

where $A_p(z)$ is the prediction error filter given in (5). Since each of these polynomials has $p+1$ zeros, a total of $2p+2$ zeros can be computed. These zeros are all on the unit circle, i.e., $z_i=\mathrm{e}^{j\omega_i}$. They appear alternately and orderly in terms of a complex-conjugate pair, and two of them are always fixed, i.e., $\omega_0=0$ and $\omega_{p+1}=\pi$. Therefore, the p frequencies $\{\omega_i\}$ in the interval $(0,\pi)$ uniquely specifies the $2p+2$ zeros of $P_p(z)$ and $Q_p(z)$ and are called the LSF. Since the LSF parameters are closely related to the frequency response, not sensitive to quantization errors and easily restricted to correspond to a stable prediction error filter, they have historically been the de-facto standard for speech coding [2].

C. De-correlated AR-parameters

A well-known property of the maximum likelihood estimator is that it produces estimates which, asymptotically, are samples from a normal distribution centered on the true value $\boldsymbol{\theta}$ and with a covariance matrix equal to the inverse Fisher information matrix (FIM) $\mathcal{I}^{-1}(\boldsymbol{\theta})$, i.e., [14]

$$\widehat{\boldsymbol{\theta}}\sim\mathcal{N}\left(\boldsymbol{\theta},\mathcal{I}^{-1}(\boldsymbol{\theta})\right). \tag{8}$$

As explained in [10], an AR-codebook can be trained on the de-correlated AR-coefficient vectors where the de-correlation matrix is derived from the FIM. The FIM is in general defined as [14]

$$\mathcal{I}(\boldsymbol{\theta})=E\left\{\frac{\partial\ln p(\boldsymbol{x};\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\frac{\partial\ln p(\boldsymbol{x};\boldsymbol{\theta})^T}{\partial\boldsymbol{\theta}}\right\} \tag{9}$$

where E indicates the mathmatical expectation, $p(\boldsymbol{x};\boldsymbol{\theta})$ indicates probablity density function of $\boldsymbol{x}$ under the condition $\boldsymbol{\theta}$, and $\boldsymbol{x}$ and $\boldsymbol{\theta}$ are a signal segment containing N observations and an unknown parameter vector, respectively. For the case of AR parameters, we can define this parameter vector as

$$\boldsymbol{\theta}=[\boldsymbol{a}^T \quad \sigma^2]^T\ . \tag{10}$$

For the AR parameters, it can be shown that the FIM is approximately block diagonal and given by [15]

$$\mathcal{I}(\boldsymbol{\theta})\approx N\begin{bmatrix}(\boldsymbol{A}_1\boldsymbol{A}_1^T-\boldsymbol{A}_2\boldsymbol{A}_2^T)^{-1} & 0\\ 0 & \dfrac{1}{2\sigma^4}\end{bmatrix} \tag{11}$$

where $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ are lower triangular matrices given by

$$\boldsymbol{A}_1 = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ a_1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{p-1} & a_{p-2} & \cdots & 1 \end{bmatrix} \tag{12}$$

$$\boldsymbol{A}_2 = \begin{bmatrix} a_p & 0 & \cdots & 0 \\ a_{p-1} & a_p & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_1 & a_2 & \cdots & a_p \end{bmatrix}. \tag{13}$$

Since the FIM is block diagonal, we can easily extract the part pertaining to the AR-vector $\boldsymbol{a}$ from the FIM and use it for de-correlating the AR-vector. Specifically, we will use the following weighting matrix

$$\mathbf{W} = (\boldsymbol{A}_1\boldsymbol{A}_1^T - \boldsymbol{A}_2\boldsymbol{A}_2^T)^{-1} \ . \tag{14}$$

D. Modified Features

When clustering the raw or de-correlated AR parameters using a k-means algorithm, we implicitly assume that the feature vectors pertaining to a given cluster are normally distributed with either an isotropic or a general covariance matrix. Since the AR-parameters might in general lie in a non-elliptical manifold, we also perform the clustering directly on such a manifold which is learned using the SMCE algorithm [11]. The goal of the SMCE is to build a sparse similarity matrix $\mathbf{S}$ for the similarity graph describing how similar the various AR training vectors (either raw or de-correlated) are. For the details of similarity matrix computation, please refer to [11]. From this similarity matrix, the diagonal degree matrix $\boldsymbol{D}$ can easily be computed. The r^{th} diagonal element describes the amount that the AR training vectors is connected to the r^{th} training vector on the similarity graph. The Laplacian matrix $\boldsymbol{L}$ is then defined as [16]

$$\mathbf{L} = \mathbf{I} - \mathbf{D}^{-\frac{1}{2}}\mathbf{S}\mathbf{D}^{-\frac{1}{2}} \tag{15}$$

To extract the training vectors (or features) $\{\boldsymbol{f}_r\}_{r=1}^R$ used in the clustering algorithm, we then factorise the Laplacian matrix using the singular value decomposition (SVD) and have

$$\mathbf{L} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T \ . \tag{16}$$

From the matrix $\boldsymbol{V}$, we then extract the last p singular vectors to form new matrix as follows

$$\boldsymbol{Y} = [\boldsymbol{y}_1 \quad \cdots \quad \boldsymbol{y}_R] = [\boldsymbol{v}_R \quad \cdots \quad \boldsymbol{v}_{R-p+1}]^T \ . \tag{17}$$

By normalizing the column of $\boldsymbol{Y}$, we then get the R training vectors as

$$\boldsymbol{f}_r = \frac{\boldsymbol{y}_r}{\| \boldsymbol{y}_r \|_2} \ , \quad r = 1, \ldots, R \tag{18}$$

which are used as the features in the clustering algorithm.

III. Clustering

In the previous section, we have described the feature and weight extraction in Fig. 1, and we now proceed to describe the clustering which will be performed using the weighted k-means clustering. For the discussion, we will assume that we have R feature vectors $\{\boldsymbol{f}_r\}_{r=1}^R$ each having its own weight matrix $\{\boldsymbol{W}_r\}_{r=1}^R$ that might possibly be the identity matrix (e.g., for the case of LSF features).

In the weighted k-means clustering, the objective function

$$J(\boldsymbol{U}, \boldsymbol{S}) = \sum_{r=1}^{R}\sum_{m=1}^{M} s_{rm}(\boldsymbol{f}_r - \boldsymbol{\mu}_m)^T\boldsymbol{W}_r(\boldsymbol{f}_r - \boldsymbol{\mu}_m) \tag{19}$$

is minimized where

$$\boldsymbol{U} = [\boldsymbol{\mu}_1 \quad \cdots \quad \boldsymbol{\mu}_M] \tag{20}$$

and

$$\boldsymbol{S} = \begin{bmatrix} s_{11} & \cdots & s_{1M} \\ \vdots & & \vdots \\ s_{R1} & \cdots & s_{RM} \end{bmatrix} \tag{21}$$

indicate the cluster centers marixes and the binary indicator matrixes describing which feature vectors are assigned to the corresponding clusters, respectively. The problem can easily be solved using an EM-algorithm [17] that estimates the cluster centers and indicator matrixes by the followng iteration

$$\boldsymbol{\mu}_m = \left[\sum_{r=1}^{R} s_{rm}\boldsymbol{W}_r\right]^{-1}\sum_{r=1}^{R} s_{rm}\boldsymbol{W}_r\boldsymbol{f}_r \tag{22}$$

where

$$s_{rm} = \begin{cases} 1 & m = \underset{j \in \{1,\ldots,M\}}{\operatorname{argmin}} \ (\boldsymbol{f}_r - \boldsymbol{\mu}_j)^T\boldsymbol{W}_r(\boldsymbol{f}_r - \boldsymbol{\mu}_j) \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

We use the implementation called k-means++ [18] for obtaining these estimates.

IV. System Evaluation and Comparison

A. Data Set and the Comparison

All experiments were conducted on the TIMIT corpus [19] where speech signals are sampled at 16 kHz. The AR parameters were estimated using the LPC-function in MATLAB from signal segments containing 512 samples, extracted with an overlap of 50% frame and windowed by a sine window [20]. Considering the memory requirements of the SMCE methods, small training database with 150 utterances were used for all methods, and large training database with 700 utterances were used except for the SMCE methods. Different numbers of training data were used for training the codebook consisting of M=8, 16, 32, 64, 128, 256, 1024 separately AR-vectors, each having an order of p=16. Another 100 utterances were used for testing the performance based on the metrics in Table 1.

In total, we evaluated eight different ways of training the AR-codebook. In addition to the 1) classical LSF-approach with small database (LSF) and 2) classical LSF-approach with large database (LSF_l), we evaluated 3) AR-parameters without weighting with small database (AR), 4) AR-parameters without weighting with large database (AR_l), 5) AR-parameters with weighting with small database (W-AR), 6) AR-parameters with weighting with large database (W-AR_l), 7) SMCE without any weighting with small database (SMCE) and 8) SMCE with weighting with small database (W-SMCE). All of the methods were run in MATLAB 2019b. For the SMCE algorithm, the maximum neighbor was set as 500, the

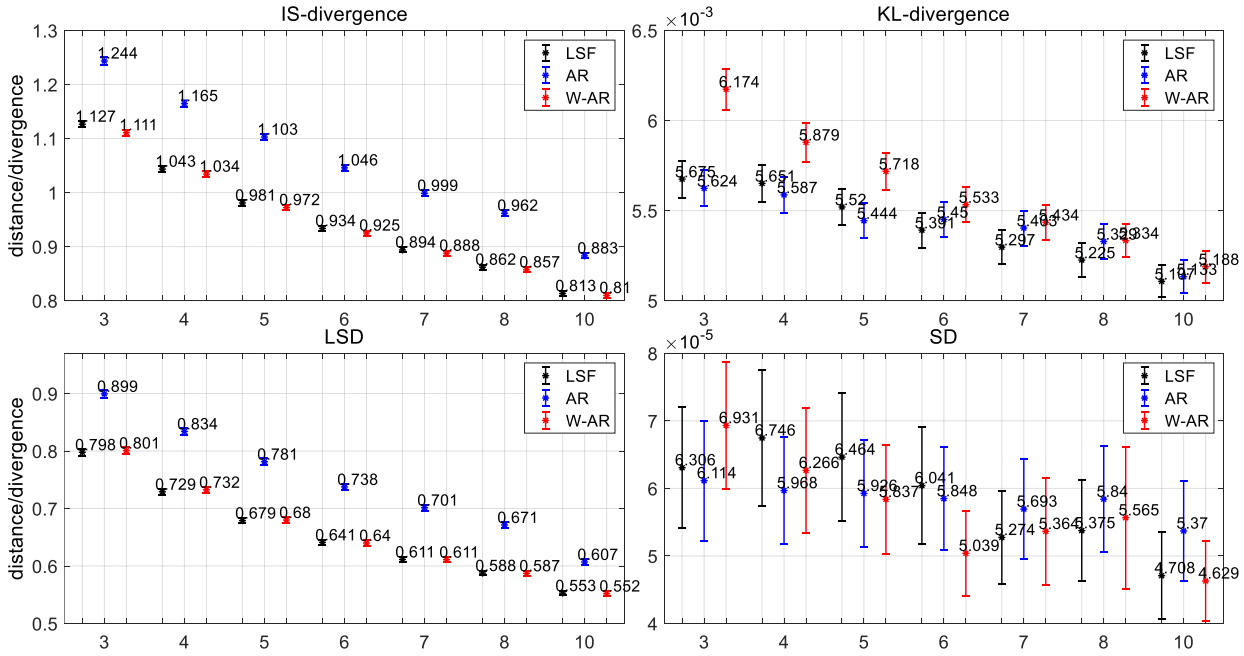

Fig. 3. Mean and 95% confidence intervals of the performances for the AR coefficients estimated by the codebooks with different bit allocation shown in horizontal axis indicates.

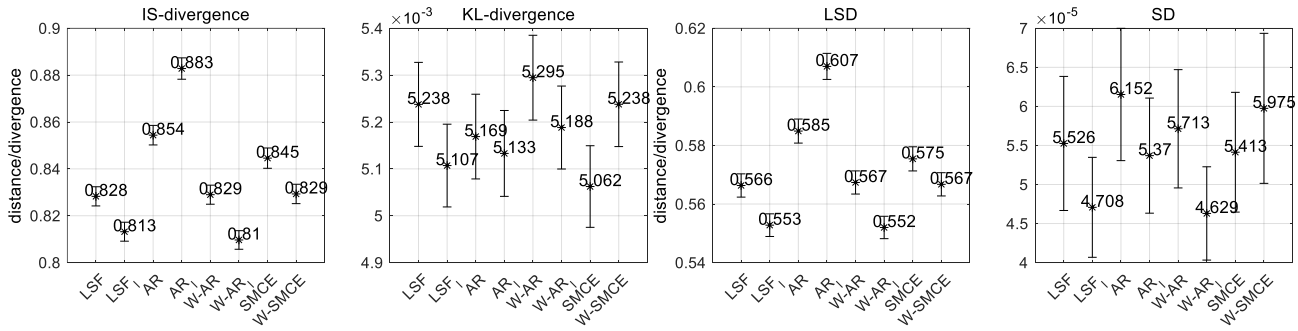

Fig. 2. Mean and 95% confidence intervals of the performances for the AR coefficients estimated by the codebooks.

feature embedding features was set to 12, and λ was set to 0.1 for the trade-off of the sparsity and the rebuilding matrix error.

B. Performance Comparison

Fig. 2 shows the performance comparison of the codebooks evaluated on the test data set using different metrics and different codebook sizes, in which the large database was trained the codebooks. Larger codebook size shows better performance for all codebook training methods. Meanwhile, the difference of the performance between the different approaches is smaller for larger codebook size. Especially the performance using the IS-divergence, the KL-divergence, and the LSD between the LSF-approach and the weighting AR-approach were compared. For the IS-divergence, the weighted AR approach is slightly better than others, because the IS-divergence metric, the weighting AR-approach distance and the LDR are all from the assumption of the Gaussian excitation signal. For the KL-divergence, the LSF approach is marginally better than others for larger codebook size.

Fig. 3 shows the performance comparison of the codebook with 1024 codewords evaluated on the test data set using different metrics, in which the codebook was trained by the small database and large database separately. Large training data gives better performance of the metrics except for the IS divergence and the LSD for the AR approach. Basically, there is no difference for speech distortion measure. The general trend seems that the weighted approaches perform well across the IS-divergence. The SMCE approach improved the performance of the KL-divergence to some degree. The classical LSF-approach seems to be on par with these approaches. The main disadvantage of the SMCE approaches are their computational complexity and memory requirements. However, since the training is often performed offline, the computational complexity is not an issue.

V. Conclusion

In this paper, we have compared a number of ways to train an AR-codebook. In addition to the classical approach of training LSF parameters, we also evaluated the performance of using raw and de-correlated AR parameters and manifold features produced by the SMCE algorithm. The experimental results showed that training an AR-codebook from the de-correlated features can improve the performance in terms of the Itakura-Saito divergence, but not in terms of the Kullback-Leibler divergence, the log-spectral distortion and the speech distortion.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Grant No. 61831019 and No. 61471014).

REFERENCES

[1] J. Makhoul, "Linear prediction: A tutorial review," *Proc. IEEE*, vol. 63, no. 4, pp. 561–580, Apr. 1975, doi: 10.1109/PROC.1975.9792.

[2] J. Jensen and C. H. Taal, "An algorithm for predicting the intelligibility of speech masked by modulated noise maskers," *IEEEACM Trans. Audio Speech Lang. Process.*, vol. 24, no. 11, pp. 2009–2022, 2016.

[3] Y. Ephraim and D. Malah, "Speech enhancement using a minimum mean-square error log-spectral amplitude estimator," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 33, no. 2, pp. 443–445, Apr. 1985, doi: 10.1109/TASSP.1985.1164550.

[4] M. Schroeder and B. S. Atal, "Code-excited linear prediction (CELP): High-quality speech at very low bit rates," in *ICASSP'85. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1985, vol. 10, pp. 937–940.

[5] D. Giacobello, M. G. Christensen, M. N. Murthi, S. H. Jensen, and M. Moonen, "Sparse Linear Prediction and Its Applications to Speech Processing," *IEEE Trans. Audio Speech Lang. Process.*, vol. 20, no. 5, pp. 1644–1657, Jul. 2012, doi: 10.1109/TASL.2012.2186807.

[6] Y. Ji, W.-P. Zhu, and B. Champagne, "Recurrent Neural Network-Based Dictionary Learning for Compressive Speech Sensing," *Circuits Syst. Signal Process.*, Feb. 2019, doi: 10.1007/s00034-019-01058-5.

[7] D. Roe, "Speech recognition with a noise-adapting codebook," in *ICASSP'87. IEEE international conference on acoustics, speech, and signal processing*, 1987, vol. 12, pp. 1139–1142.

[8] F. Itakura, "Line spectrum representation of linear predictor coefficients of speech signals," *J. Acoust. Soc. Am.*, vol. 57, no. S1, pp. S35–S35, Apr. 1975, doi: 10.1121/1.1995189.

[9] P. Kabal and R. P. Ramachandran, "The computation of line spectral frequencies using Chebyshev polynomials," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 34, no. 6, pp. 1419–1426, Dec. 1986, doi: 10.1109/TASSP.1986.1164983.

[10] M. G. Christensen, "Metrics for vector quantization-based parametric speech enhancement and separation," *J. Acoust. Soc. Am.*, vol. 133, no. 5, pp. 3062–3071, 2013.

[11] E. Elhamifar and R. Vidal, "Sparse Manifold Clustering and Embedding," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 55–63.

[12] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Trans. Acoust. Speech Signal Process.*, vol. 23, no. 1, pp. 67–72, 1975.

[13] G. Cybenko, "The numerical stability of the Levinson-Durbin algorithm for Toeplitz systems of equations," *SIAM J. Sci. Stat. Comput.*, vol. 1, no. 3, pp. 303–319, 1980.

[14] S. M. Kay, *Fundamentals of statistical signal processing*. Prentice Hall PTR, 1993.

[15] B. Friedlander and B. Porat, "The exact Cramer-Rao bound for Gaussian autoregressive processes," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 25, no. 1, pp. 3–7, Jan. 1989, doi: 10.1109/7.18656.

[16] U. Von Luxburg, "A tutorial on spectral clustering," *Stat. Comput.*, vol. 17, no. 4, pp. 395–416, 2007.

[17] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. R. Stat. Soc. Ser. B Methodol.*, vol. 39, no. 1, pp. 1–22, 1977.

[18] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, 2007, pp. 1027–1035.

[19] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, "DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1," *NASA STIRecon Tech. Rep. N*, vol. 93, Feb. 1993.

[20] J. Barker, R. Marxer, E. Vincent, and S. Watanabe, "The third 'CHiME'speech separation and recognition challenge: Dataset, task and baselines," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015, pp. 504–511.