Lightweight LPCNet-based Neural Vocoder with Tensor Decomposition

Hiroki Kanagawa and Yusuke Ijima

NTT Corporation, Japan

hiroki.kanagawa.wk@hco.ntt.co.jp

Abstract

This paper proposes a lightweight neural vocoder based on LPCNet. The recently proposed LPCNet exploits linear predictive coding to represent vocal tract characteristics, and can rapidly synthesize high-quality waveforms with fewer parameters than WaveRNN. For even greater speeds, it is necessary to reduce the time-heavy two GRUs and the DualFC. Although the original work only pruned the first GRU weight, there is room for improvements in the other GRU and DualFC. Accordingly, we use tensor decomposition to reduce these remaining parameters by more than 80%. For the proposed method we demonstrate that 1) it is 1.26 times faster on a CPU, and 2) it matched naturalness of the original LPCNet for acoustic features extracted from natural speech and for those predicted by TTS.

Index Terms: neural vocoder, LPCNet, lightweight neural waveform generation, speech synthesis

1. Introduction

Text-to-speech (TTS) advanced dramatically with the advent of the neural network-based vocoder (neural vocoder). The well-known model called WaveNet [1], which directly models the speech sample distribution by dilated causal convolution, successfully synthesizes more natural speech than conventional signal-processing based vocoder [2, 3, 4]. Nevertheless, it offers only serial processing and its inference is slow due to the autoregressive (AR) model. One approach to speed up neural vocoders is to use flow-based generation models [5, 6, 7]. They realize the simultaneous generation of multiple waveform samples by introducing knowledge distillation from AR-WaveNet into non-AR models. However, since parallel processors such as GPUs are required for fast inferencing, the number of applicable devices is limited.

Fortunately, several methods using simple models with low computation complexity have been proposed to achieve CPU inferencing. WaveRNN [8] replaces the dilated causal convolutions in WaveNet with simple GRUs for sequence modeling. They also examined sparse GRU and simultaneous multisample generation, and realized fast vocoding on a CPU. A similar approach was proposed for SqueezeWave [9], which replaces 1D convolution of WaveGlow [7] with depth-wise convolution and simplifies the conditioning network. All of the neural vocoders described above yield very natural speech, even though they do not use classical speech synthesis techniques. In addition, combining the classical source-filter model with the neural vocoder provides better results, even with a small number of parameters [10, 11, 12]. The source-filter model makes strong assumptions about the independence of source signals and vocal tract characteristics. This allows us to reasonably approximate the problem to be solved, without increasing neural vocoder's size. One of these methods, LPCNet [10], reduces the number of model parameters to about 30% compared to WaveRNN by using linear predictive coding (LPC) for the speech generation process. An extension of this [13], makes the inference 1.5 times faster by generating two excitation samples simultaneously. However, these LPCNet-based approaches only prune one of the two GRUs, and thus the other modules offer room for enhancement. Furthermore, in TTS vocoder applications, feature mismatch tends to occur between training and generation because the input acoustic features are more smoothed than those extracted from natural speech. In this case, parameter reduction may adversely affect the quality, and so should be validated.

In this paper, we propose a lightweight LPCNet-based neural vocoder with tensor decomposition for parameter reduction. The two GRUs and DualFC, which are used in the sample rate network of LPCNet, account for more than 90% of the total processing time. As mentioned above, only the first GRU (GRU$_A$) was simplified in the original LPCNet, while the other GRU (GRU$_B$) and DualFC were implemented as is. Since GRU$_B$ and DualFC occupy about half the processing time in the sample rate network, further parameter reduction is necessary to accelerate inferencing. Although knowledge distillation can be used as a parameter reduction approach, we adopt low-rank approximation as it allows us to use the original weight parameters as initial values. Tensor-train decomposition [14, 15], which was shown to be effective in [16, 17], is used for parameter reduction of GRU$_B$. While DualFC is the weighted sum of projections by two weight matrices, and it is possible to approximate each weight matrix with low rank equivalents by singular value decomposition (SVD), we treat the two weight matrices as third-order tensors and apply higher-order SVD (HOSVD) [18]. This enables a low-rank approximation that incorporates the relationship between the weight matrices, as discussed in [19]. We investigate our proposal's robustness under multiple conditions of parameter reductions by vocoding, not only for the acoustic features extracted from natural speech but also those predicted by TTS. The naturalness of synthetic speech is maintained by our proposed model even though the number of DualFC and GRU$_B$ parameters is reduced by more than 80%. This model also achieved 1.26 times faster CPU inferencing than the original LPCNet.

2. Original LPCNet

Here, we briefly explain the original LPCNet [10] as it is the basis of this work.

2.1. Architecture and algorithm

Figure 1 shows the LPCNet network structure. LPCNet mainly consists of frame rate processing (encoder) and sample rate processing (decoder). The encoder predicts intermediate representations and LPC coefficients from input bark-scale cepstrum and 2-dimensional pitch-related features. The decoder predicts the speech samples from the intermediate representation obtained

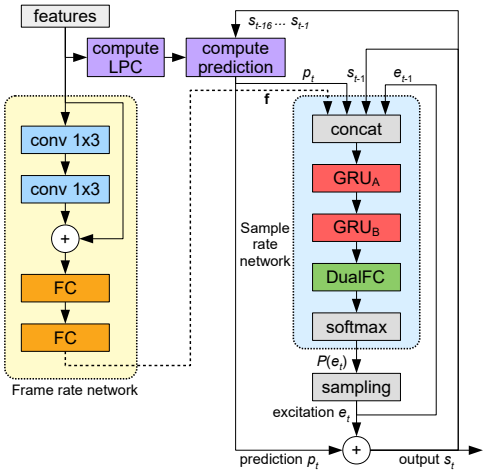

Figure 1: *Original LPCNet architectures [10]. It has two types of network. One is the encoder (on the left) and the other is the decoder (on the right).*

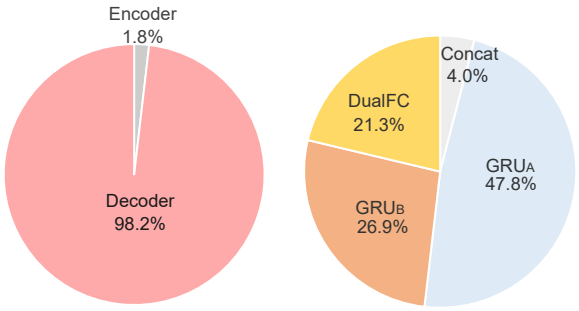

Figure 2: *The inference time ratios of encoder (on the left) and decoder (on the right).*

from the encoder and LPC coefficients. Unlike general neural vocoders, which directly predict speech samples, LPCNet only predicts excitation e_t by a non-linear 8-bit inverse μ-law algorithm. The noise present in the high-frequency bands in e_t is reduced by applying an inverse pre-emphasis filter. The speech sample s_t at time T is predicted by using the LPC coefficient, previous samples, and the obtained excitation e_t as follows:

$$s_t = p_t + e_t, \tag{1}$$

$$p_t = \sum_{m=1}^{M} a_m s_m, \tag{2}$$

where M, m, a_m are the LPC order, its index and LPC coefficient, respectively.

2.2. Inferencing time

To prepare for further speed enhancement, we investigated the inference time ratios of each module of LPCNet. Figure 2 (on the left) shows the processing time ratios of encoder and decoder when processing acoustic features with 24 kHz sampling frequency and 10 ms frame shift. From the figure, the decoder needs to be reduced for faster vocoder processing because its ratio is quite large. Figure 2 (on the right) shows the inference time ratio of the decoder. Although GRUA shows the largest ratio, this paper does not focus on it because the original LPCNet has already pruned its parameters. While GRUB and DualFC occupy 48.2% of the inference time, parameter size reduction has not been attempted for these modules.

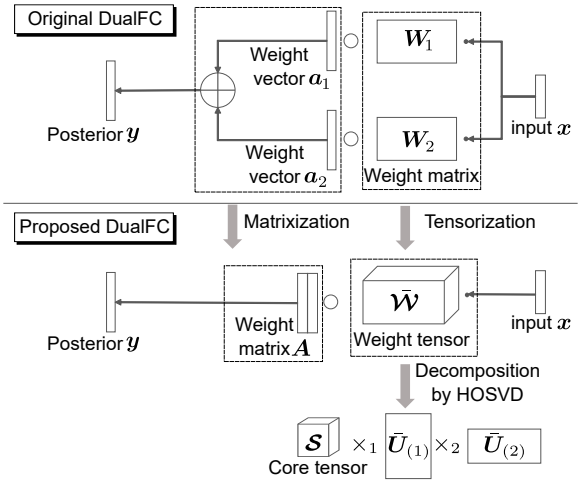

Figure 3: *DualFC tensorization and its decomposition by HOSVD. Biases are omitted for clarity.*

3. Proposed lightweight LPCNet using tensor decomposition

We propose a lightweight LPCNet-based neural vocoder using tensor decomposition for higher inferencing speeds without any significantly quality degradation. As mentioned in Section 2.2, our proposed method aims to reduce the parameters of the time-consuming DualFC and GRUB.

3.1. DualFC tensorization and its decomposition via higher-order SVD (HOSVD DualFC)

Before describing the tensor decomposition of DualFC, we introduce its mathematical definition. DualFC predicts the posterior probability $\boldsymbol{y}$ of 8-bit excitation class by processing GRUB's output $\boldsymbol{x}$. The definition is given by:

$$\boldsymbol{y} = \mathrm{DualFC}\left(\boldsymbol{x}\right) = \sum_{i=1}^{2} \boldsymbol{a}_i \circ \tanh\left(\boldsymbol{W}_i \boldsymbol{x} + \boldsymbol{b}_i\right), \tag{3}$$

where M, N, i, $\boldsymbol{W}_i \in \mathbb{R}^{N \times M}$, $\boldsymbol{b}_i \in \mathbb{R}^{N}$, and $\boldsymbol{a}_i \in \mathbb{R}^{N}$ are $\boldsymbol{x}$'s dimension, $\boldsymbol{y}$'s dimension, index, weight matrix, bias vector, and weight vector, respectively. $\circ$ denotes Hadamard product operation.

Figure 3 overviews our HOSVD approach. To apply tensor decomposition, we combine the weight matrices as a tensor and vectors as a matrix, and reformulate DualFC as follows:

$$\boldsymbol{y} = \mathrm{DualFC}\left(\boldsymbol{x}\right) = \boldsymbol{A} \circ \tanh\left(\boldsymbol{\mathcal{W}} \boldsymbol{x} + \boldsymbol{B}\right), \tag{4}$$

where $\boldsymbol{A} \in \mathbb{R}^{N \times 2}$, $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{N \times M \times 2}$, $\boldsymbol{B} \in \mathbb{R}^{N \times 2}$ are $\boldsymbol{A} = [\boldsymbol{a}_1\ \boldsymbol{a}_2]$, $\boldsymbol{\mathcal{W}} = [\boldsymbol{W}_1\ \boldsymbol{W}_2]$, and $\boldsymbol{B} = [\boldsymbol{b}_1\ \boldsymbol{b}_2]$, respectively. We employ higher-order singular value decomposition (HOSVD) [18] for $\boldsymbol{\mathcal{W}}$ decomposition. Mode-1 and mode-2 flattening are applied to $\boldsymbol{\mathcal{W}}$. Flattened matrices $\boldsymbol{W}_{(1)} \in \mathbb{R}^{N \times 2M}$ and $\boldsymbol{W}_{(2)} \in \mathbb{R}^{M \times 2N}$ are decomposed by SVD:

$$\boldsymbol{W}_{(n)} = \boldsymbol{U}_{(n)} \boldsymbol{\Sigma}_{(n)} \boldsymbol{V}_{(n)}^{\top}, \tag{5}$$

where n is the mode index. $\boldsymbol{U}_{(n)}$ and $\boldsymbol{V}_{(n)}$ are unitary matrices. $\boldsymbol{\Sigma}_{(n)}$ is the diagonal matrix with $\boldsymbol{W}_{(n)}$ singular values on the diagonal in descending order. In order to create a low-rate approximation of $\boldsymbol{U}_{(1)} \in \mathbb{R}^{N \times N}$ and $\boldsymbol{U}_{(2)} \in \mathbb{R}^{M \times M}$, we extract the matrices $\bar{\boldsymbol{U}}_{(1)} \in \mathbb{R}^{N \times N'}$ and $\bar{\boldsymbol{U}}_{(2)} \in \mathbb{R}^{M \times M'}$ from corresponding singular values up to the rank of N' and M',

respectively. This allows DualFC's weight tensor $\boldsymbol{\mathcal{W}}$ to be reconstructed as:

$$\bar{\boldsymbol{\mathcal{W}}} \approx \boldsymbol{\mathcal{S}} \times_1 \bar{\boldsymbol{U}}_{(1)} \times_2 \bar{\boldsymbol{U}}_{(2)}, \tag{6}$$

where $\times_n$ denotes mode-n product operation and $\boldsymbol{\mathcal{S}}$ is the core tensor given by:

$$\boldsymbol{\mathcal{S}} = \boldsymbol{\mathcal{W}} \times_1 \bar{\boldsymbol{U}}_{(1)}^{\top} \times_2 \bar{\boldsymbol{U}}_{(2)}^{\top}. \tag{7}$$

By tensor $\bar{\boldsymbol{\mathcal{W}}}$ reconstruction, the number of parameters of DualFC's weight matrices can be reduced from $2MN$ to $2M'N' + MN' + NM'$.

3.2. Tensor-train decomposition for GRUB (TTGRUB)

We decompose LPCNet's GRUB using tensor-train (TT) decomposition [14], which is one of the low-rank approximations. Before decomposing GRU, we first define the TT decomposition of the feed-forward layer given by

$$\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}, \tag{8}$$

where $\boldsymbol{x} \in \mathbb{R}^M$, $\boldsymbol{y} \in \mathbb{R}^N$, $\boldsymbol{W} \in \mathbb{R}^{N \times M}$, $\boldsymbol{b} \in \mathbb{R}^N$ are input vector, output one, weight matrix, and bias, respectively. M and N are factorized into d integer arrays such as:

$$M = \prod_{k=1}^{d} m_k, \tag{9}$$

$$N = \prod_{k=1}^{d} n_k. \tag{10}$$

Then, the input vector $\boldsymbol{x}$ and the output one $\boldsymbol{y}$ can be reformulated into tensors with factorized dimensions, and Eq. (8) is redefined as:

$$\begin{aligned}\boldsymbol{\mathcal{Y}}(j_1, \ldots, j_d) = \sum_{i_1=1}^{m_1} \cdots \sum_{i_d=1}^{m_d} \boldsymbol{\mathcal{W}}((i_1, j_1), \ldots, (i_d, j_d)) \\ \cdot \boldsymbol{\mathcal{X}}(i_1, \ldots, i_d) + \boldsymbol{\mathcal{B}}(j_1, \ldots, j_d),\end{aligned} \tag{11}$$

where $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{m_1 \times \cdots \times m_d}$, $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{m_1 n_1 \times \cdots \times m_d n_d}$, $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ are tensor representations of $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{W}$ and $\boldsymbol{b}$, respectively. The weight tensor $\boldsymbol{\mathcal{W}}$ is decomposed by using low-rank core tensors $\boldsymbol{\mathcal{G}}_k$ as follows.

$$\begin{aligned}&\boldsymbol{\mathcal{W}}((i_1, j_1), \ldots, (i_d, j_d)) \\ &\quad \approx \boldsymbol{\mathcal{G}}_1(i_1, j_1) \ldots \boldsymbol{\mathcal{G}}_i(i_d, j_d),\end{aligned} \tag{12}$$

$$\boldsymbol{\mathcal{G}}_k \in \mathbb{R}^{m_k \times n_k \times r_{k-1} \times r_k} \ \forall k \in [1, d], \tag{13}$$

where $r_k \forall k \in [1, d]$, called TT-rank, is the hyperparameter required for TT decomposition, and $r_0 = r_d = 1$. Thus TT layer (TTL) approximates Eq. (8) with Eq. (12) as follows:

$$\bar{\boldsymbol{y}} \approx \mathrm{TTL}(\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{x}), \tag{14}$$

where $\bar{\boldsymbol{y}}$ is the output vector obtained by the low-ranked approximation $\boldsymbol{W}$.

TTGRUB is achieved by replacing all linear transformations regarding the input vector $\boldsymbol{x}$ with TTLs for GRUB's update gate, reset gate, and output. Therefore, the number of weight matrix parameters, for the mapping from input to hidden layer, can be reduced from $3MN$ to $\sum_{k=1}^{d} m_k n_k r_{k-1} r_k + 2 m_1 n_1$.

4. Experiments

4.1. Setup

We used speech data uttered by a Japanese professional female speaker. The sampling frequency is 24 kHz. Twenty utterances were extracted as evaluation data (1.1 minutes), and the others were used for training (11.6 hours) and validation (1.9 hours).

LPCNet-based neural vocoders were trained under the same conditions as [10], using a 20-dimensional vector consisting of 18-dimensional bark-scale cepstrum coefficients, pitch period, and pitch correlation. The analysis frame shift was 10 ms. For comparison with a conventional signal-processing based vocoder, 40-dimensional mel-cepstrum, logarithmic F0, voiced unvoiced binary flag, and five-dimensional aperiodicity with 5 ms frame shift were extracted by STRAIGHT [3].

Regarding the tensor decomposition's hyperparameters, we used (2,4,2) for DualFC's core tensor shape. The input and output dimensions of TTGRUB were set to (16,32) and (4,4) so as to satisfy Eq. (9) and Eq. (10), respectively. Two types of TTGRUBs (v1,v2) with different TT-ranks (1,8,1 and 1,4,1) were trained to investigate their speed and subjective evaluation performance when the number of parameters were reduced. These hyper parameters were chosen from our preliminary experiments as they can reduce the weight parameters by more than 80%. Table 1 and Table 2 show the number of parameters and reduction rates for DualFC and TTGRUB for these settings. In all experiments, decomposed tensor's weights were retrained with the other frozen weights.

Table 1: *DualFC parameter size comparison. The reduction rate was obtained by dividing the number of parameters of the decomposed model by that of the original DualFC.*

DualFC	Core shape	# of parameters	Reduction rate [%]
Original	-	9216	-
HOSVD DualFC	2,4,2	1616	82.5

Table 2: GRUB *parameter size comparison. The reduction rate was obtained by dividing the number of parameters of the decomposed model by that of the original* GRUB.

GRUB	TT-rank	# of parameters	Reduction rate [%]
Original	-	25440	-
TTGRUB (v1)	1,8,1	3376	86.7
TTGRUB (v2)	1,4,1	2096	91.8

Table 3: *Average RTFs obtained from all evaluation data. "Speed enhancement" denotes the improvement over original LPCNet.*

Method	Average RTF	Speed enhancement
Original LPCNet	0.35	-
HOSVD DualFC	0.33	1.08x
TTGRUB (v1)	0.30	1.16x
HOSVD DualFC+TTGRUB (v1)	0.28	1.26x
HOSVD DualFC+TTGRUB (v2)	0.27	1.29x

4.2. Comparing inference speeds for waveform generation

The real-time factors (RTFs) were calculated to measure the inference speeds of the original LPCNet and our proposed vocoder. The RTF definition is given by:

$$\mathrm{RTF} = T_{\mathrm{inference}} / T_{\mathrm{data}}, \tag{15}$$

where T_{data} and $T_{\mathrm{inference}}$ are speech length and inference time measured on an Intel Core i7-8750H CPU 2.20 GHz, respectively. Table 3 shows method, averaged RTFs obtained by all evaluation data, and RTF improvements.

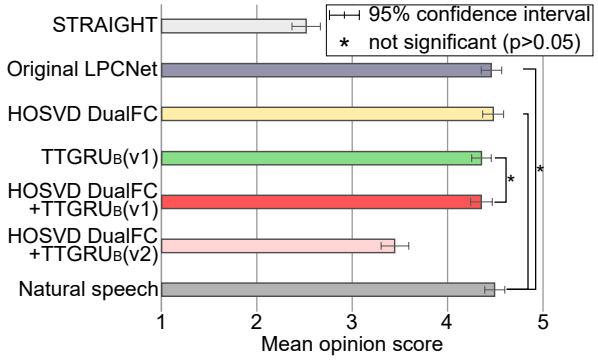

Figure 4: *Mean opinion scores of naturalness of synthetic speech. Acoustic features for vocoding were extracted from natural speech.*

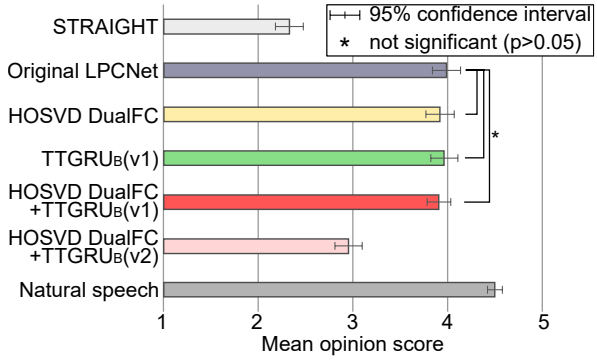

Figure 5: *Mean opinion scores in terms of naturalness. Acoustic features for vocoding were predicted by TTS models.*

Applying HOSVD to DualFC increases the overall speed 1.08 times over the original (1.5 times for DualFC alone). This speed improvement was reasonable given the parameter values used in downsizing. By introducing TTGRUB (v1), the overall speed improvement was 1.16 times (2.0 times for GRUB alone). We expected a dramatic improvement with the parameter size reduction, but the improvement was not as significant as expected. The TTGRUB's forward-propagation includes matrix transposition after reshaping, and we found that it consumed too much time. To further enhance the speed, we will explore the block-term decomposition [20] in the future that can avoid this operation and has fewer parameters [21]. When TTGRUB (v1) and TTGRUB (v2) were used with HOSVD DualFC, their respective parameter size reductions contributed to 1.26 and 1.29 times speed enahncements, respectively.

4.3. Subjective evaluations

We subjectively evaluated naturalness of synthetic speech by using mean opinion score (MOS) on a five-point scale ranging from 5: very natural to 1: very unnatural. Four of all evaluation utterances were randomly chosen for each method. Sixty listeners participated in the test via crowdsourcing, each of them giving scores for synthetic speech.

4.3.1. Vocoding for extracted acoustic features from natural speech

Figure 4 shows the subjective evaluation results of vocoding with acoustic features extracted from natural speech. Original LPCNet and our proposed HOSVD DualFC obviously outperformed STRAIGHT, and they were comparable to natural speech. Among the vocoders that introduced TTGRU, TTGRUB (v1) was slightly inferior to these two methods. The score of the method using both TTGRUB(v1) and HOSVD DualFC was comparable to that of TTGRUB(v1). Since DualFC created no degradation, TTGRUB may be responsible for the deterioration. This is confirmed by the fact that TTGRUB (V2) + HOSVD DualFC yielded strong degradation. These results found that 1) HOSVD DualFC offered robust performance, 2) TTGRUB(v1) created gave slight degradation in vocoding based on acoustic features extracted from natural speech.

4.3.2. Vocoding based on acoustic features predicted by TTS

To investigate robustness against degraded acoustic features, TTS models were also trained with the same data as the neural vocoders. The TTS models consisted of four feed-forwards and two uni-directional LSTMs with 256 units. We fed a 307-dimensional vector with 305-dimensional linguistic features, UV flags, and logarithmic F0s to the TTS models. As the acoustic features of TTS model for the LPCNet-based vocoder, we used a 19-dimensional vector containing cepstrum coefficients and pitch correlations. The TTS model for the STRAIGHT vocoder used a 45-dimensional vector combining mel-cepstrum coefficients and aperiodicity as acoustic features. Furthermore, we applied CMVN [22] to these acoustic features, in combination with their Δ and $\Delta\Delta$ coefficients. The TTS models were optimized via a minimum mean squared error criterion with 50 iterations by Adam [23]. The mini-batch size and the learning rate were 32 and 10^{-4}, respectively. At the time of testing, we applied MLPG [24] to predict smooth static features. Variance scaling [25] were also used to compensate spectral variances. To evaluate the robustness to degraded spectrum features, it would be desirable to exclude the effects of prosodic features (i.e., F0 and phoneme durations). For this purpose, we used these features extracted from the natural speech.

Figure 5 shows the subjective evaluation results of vocoding with the acoustic features predicted by TTS. The obvious superiority of Original LPCNet over STRAIGHT confirms that LPCNet is also useful as a vocoder for TTS, as reported in [26]. The original LPCNet was inferior to the natural speech because the predicted acoustic features were degraded more than the extracted ones, and there was a mismatch between vocoder training and inference. The proposed TTGRUB(v1) and HOSVD DualFC, and their combined methods, were comparable to the original LPCNet. As in Section 4.3.1, a even lower rank approximation, GRUB, TTGRUB(v2), was clearly degraded. These results demonstrate that our methods can robustly vocode acoustic features predicted by TTS without degrading naturalness unless the parameters are reduced as much as TTGRUB(v2).

5. Conclusions

In this work, we proposed lightweight LPCNet that uses tensor decomposition. Our proposed vocoder improved inference speed 1.26 times by reducing DualFC's and GRUB's parameters by 82.5% and 86.7%, respectively. With this setting, it also achieved robust vocoding for not only the acoustic features extracted from natural speeches but also those predicted by TTS. Thus, we believe that our vocoder is a promising approach to introducing neural vocoders to more low-resource devices. Furthermore, we can also combine our approach with prior works such as multisample generation [13].

6. References

[1] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *CoRR abs/1609.03499*, 2016.

[2] S. Imai, "Cepstral analysis synthesis on the mel frequency scale," *Proc. ICASSP*, pp. 93–96, 1983.

[3] H. Kawahara, I. Masuda-Katsuse, and A. de Cheveigne, "Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds," *Speech Communication*, vol. 27, no. 3, pp. 187–207, 1999.

[4] M. Morise, F. Yokomori, and K. Ozawa, "WORLD: A vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Trans. on Information and Systems*, vol. E99-D, no. 7, pp. 1877–1884, 2016.

[5] A. van den Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. Van Den Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, N. Casagrande, D. Grewe, S. Noury, S. Dieleman, E. Elsen, N. Kalchbrenner, H. Zen, A. Graves, H. King, T. Walters, D. Belov, and D. Hassabis, "Parallel WaveNet: Fast high-fidelity speech synthesis," *Proc. ICML*, vol. 9, 2018.

[6] W. Ping, K. Peng, and J. Chen, "Clarinet: Parallel wave generation in end-to-end text-to-speech," *Proc ICLR*, 2019.

[7] B. C. Ryan Prenger, Rafael Valle, "WaveGlow: A flow-based generative network for speech synthesis," *Proc. ICASSP*, pp. 3617–3621, 2019.

[8] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. Van Den Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient Neural Audio Synthesis," *Proc. PMLR*, pp. 2410–2419, 2018.

[9] B. Zhai, T. Gao, F. Xue, D. Rothchild, B. Wu, J. Gonzalez, and K. Keutzer, "SqueezeWave: Extremely lightweight vocoders for on-device speech synthesis," *arXiv:2001.05685*, 2020.

[10] J.-M. Valin and J. Skoglund, "LPCNet: Improving neural speech synthesis through linear prediction," *Proc. ICASSP*, pp. 5891—-5895, 2019.

[11] E. Song, K. Byun, and H.-G. Kang, "ExcitNet vocoder: A neural excitation model for parametric speech synthesis systems," *Proc. EUSIPCO*, pp. 1–5, 2019.

[12] M.-J. Hwang, F. Soong, E. Song, X. Wang, H. Kang, and H.-G. Kang, "LP-WaveNet: Linear prediction-based wavenet speech synthesis," *arXiv:1811.11913*, 2018.

[13] V. Popov, M. Kudinov, and T. Sadekova, "Gaussian LPCNet for multisample speech synthesis," *Proc. ICASSP*, pp. 6204–6208, 2020.

[14] I. Oseledets, "Tensor-train decomposition," *SIAM Journal Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.

[15] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov, "Tensorizing neural networks," *Proc. NIPS*, pp. 442–450, 2015.

[16] A. Tjandra, S. Sakti, and S. Nakamura, "Compressing recurrent neural network with tensor train," *Proc. IJCNN*, vol. 2017-May, pp. 4451–4458, 2017.

[17] Y. Yang, D. Krompass, and V. Tresp, "Tensor-train recurrent neural networks for video classification," *Proc. ICML*, vol. 8, pp. 5929–5938, 2017.

[18] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.

[19] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[20] L. Lathauwer, "Decompositions of a higher-order tensor in block terms—part II: Definitions and uniqueness," *SIAM Journal Matrix Analysis Applications*, vol. 30, no. 3, pp. 1033–1066, 2008.

[21] J. Ye, L. Wang, G. Li, D. Chen, S. Zhe, X. Chu, and Z. Xu, "Learning compact recurrent neural networks with block-term tensor decomposition," *Proc. CVPR*, pp. 9378–9387, 2018.

[22] O. Viikki and K. Laurila, "Cepstral domain segmental feature vector normalization for noise robust speech recognition," *Speech Communication*, vol. 25, pp. 133–147, 1998.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[24] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," *Proc. ICASSP*, pp. 1315–1318, 2000.

[25] H. Silén, E. Helander, J. Nurminen, and M. Gabbouj, "Ways to implement global variance in statistical speech synthesis," *Proc. INTERSPEECH*, pp. 1436–1439, 2012.

[26] Z. Kons, S. Shechtman, A. Sorin, C. Rabinovitz, and R. Hoory, "High quality, lightweight and adaptable TTS using LPCNet," *Proc. INTERSPEECH*, pp. 176–180, 2019.