tinyVocos: Neural Vocoders on MCUs

Stefano Ciapponi[1,2], Francesco Paissan[1], Alberto Ancilotto[1,2], Elisabetta Farella[1]
Energy Efficient Embedded Digital Architectures
Fondazione Bruno Kessler[1], University of Trento[2]
Trento, Italy
{sciapponi, fpaissan, aancilotto, efarella}@fbk.eu

***Abstract*—Neural Vocoders convert time-frequency representations, such as mel-spectrograms, into corresponding time representations. Vocoders are essential for generative applications in audio (e.g. text-to-speech and text-to-audio). This paper presents a scalable vocoder architecture for small-footprint edge devices, inspired by Vocos and adapted with XiNets and PhiNets. We test the developed model capabilities qualitatively and quantitatively on single-speaker and multi-speaker datasets and benchmark inference speed and memory consumption on four microcontrollers. Additionally, we study the power consumption on an ARM Cortex-M7-powered board. Our results demonstrate the feasibility of deploying neural vocoders on resource-constrained edge devices, potentially enabling new applications in Internet of Sounds (IoS) and Embedded Audio scenarios. Our best-performing model achieves a MOS score of 3.95/5 while utilizing 1.5MiB of FLASH and 517KiB of RAM and consuming 252 mW for a 1s audio clip inference.**

***Index Terms*—neural vocoders, tinyML, embedded AI, microcontrollers**

I. Introduction

The Internet of Sounds (IoS), a growing field within the Internet of Things (IoT) ecosystem, introduces the concept of ”Sound Things” [1]. These are networked computing devices capable of acquiring, processing, exchanging, or generating sound-related information. While research into sound analysis within the Embedded Audio domain has seen substantial progress [2]–[7], implementing Generative Deep Learning technologies on embedded systems for sound-based applications presents some unique challenges, especially targeting low-power microcontrollers (MCUs).

The rapid integration of generative technologies on IoT devices enables significant advancements in several application areas, ranging from anonymization [8], [9] to smart human-machine interfaces [10]. Nonetheless, this implies addressing significant challenges imposed by generative networks' energy-hungry and cloud-centric nature. Addressing the challenges of generative model deployment at the edge has become a pivotal milestone in enabling applications like LLM-powered smart assistants at the edge. A recent research stream has focused on the feasibility of running GAN-based approaches on microcontroller units (MCUs), focusing specifically on the image domain [11]. Nonetheless, the audio and speech domains are lagging behind, possibly due to the extra challenges of temporal consistency. By leveraging time-frequency representations (i.e. spectrograms), we can unlock generative capabilities in the audio processing domain that are on par with those in the image domain. This common pratice [12]–[15] opens up exciting possibilities such as GAN-based speech separation [16], text-to-speech [17], text-to-audio [13]–[15], and audio editing [18].

One major challenge still needs to be addressed: converting the generated spectrogram into a waveform. For linear magnitude spectrograms, a standard solution to estimate the phase of the generated signal is the Griffin-Lim algorithm, or its fast variant Fast Griffin-Lim algorithm [19] and then proceed to compute the inverse Fourier transform. However, generating audio as its Mel-spectrogram representation is the de-facto standard for state-of-the-art audio generation pipelines. Mel-spectrograms are not invertible, and thus, we can only approximate the linear spectrogram, which usually introduces artifacts. Neural Vocoders (NV) emerged as a reliable solution for converting Mel-spectrograms into waveforms [20]–[24]. NV are typically deep networks whose computational requirements are not compatible with the constraints of consumer-grade edge devices. Additionally, NV are the final stage of generative pipelines (e.g. text-to-speech) and thus cannot use all the available resources.

To address the limitations discussed above, and the gap in the literature regarding vocoders suitable for MCU devices, we propose an efficient pipeline that enables real-time inference on edge IoS nodes. Our contributions can be summarized as follows:

- We propose an efficient and scalable vocoder to convert Mel-spectrograms into waveforms inspired by Vocos and adapted with XiNets [25] and PhiNets [26], two edge-oriented neural architectures. The code is available on GitHub[1];
- We benchmark our vocoder on two datasets, measuring the single-speaker and multi-speaker performance.
- We deploy our solution on four Arm Cortex M7-based microcontrollers, assessing power consumption, latency, and RAM usage.

The paper structure is as follows: Sec. II presents the literature relevant to our work, analyzing various neural vocoding approaches aimed at enhancing model efficiency and identifying literature gaps. Sec. III introduces the methods proposed in this manuscript, briefly describing the Vocos architecture and the optimization choices that contributed to a more efficient vocoder. Sec. IV details the experimental setup to validate

[1] https://github.com/sciapponi/tiny-vocos

the optimized vocoder against different computational budgets, showcasing both qualitative and quantitative metrics. Finally, Sec. V describes the findings of the paper demonstrating the feasibility of audio synthesis on tiny devices and, thus, for IoS.

II. Related Works

In this section, we present an overview of Neural Vocoder models and discuss various research efforts to enhance their efficiency.

A. Neural Vocoders

Neural vocoders can be broadly classified into two families: autoregressive and GAN-based models. Autoregressive vocoders generate one waveform sample at a time, modelling the value of each timestep on previously generated ones. Although these models, such as WaveNet [27] and SampleRNN [28], can produce high-quality audio, they are computationally intensive, resulting in slow inference and high memory requirements. WaveGlow [29] represents a hybrid approach that uses Inverse Autoregressive Flows [30] to model sample distributions in a non-autoregressive manner, transforming a noise latent space into a target speech distribution. On the other hand, GAN-based vocoders, including MelGAN [31] or HiFi-GAN [20], use a generator to produce entire audio segments at once and a discriminator to enhance audio quality. While this variety of vocoders may produce less consistent audio quality than autoregressive models, they offer faster inference and lower computational requirements, making them more suitable for real-time applications and less powerful hardware platforms. The improved efficiency of GAN-based models is primarily due to their reduced parameter count, lack of complex long-range dependency modelling, and use of hardware-friendly operations. Most GAN-based vocoders generate waveforms by upsampling the spectrogram temporal axis. This approach creates a common computational bottleneck consisting of transposed convolutions and Multi Receptive Field [32] modules stacking dilated convolutions. These operations, however, can have significant memory requirements, limiting their applicability on resource-constrained devices like MCUs.

B. Efficient Vocoders

Research on efficient neural vocoders aims to develop models with faster inference, real-time capabilities, and low computational requirements. Several approaches have been proposed to address these challenges. *Basis-Mel GAN* [33] represents audio signals with learned basis and associated weights, and it is inspired by TASNet [34]. This method models the waveform as a non-negative weighted sum of N basis signals, resulting in a more compact and efficient representation that mitigates the Mel-GAN upsampling network complexity. *StyleMelGAN* [35] modifies the Mel-GAN architecture by employing temporal adaptive normalization to style a low-dimensional noise vector containing the acoustic features of the target speech, aiming to speed up inference on high-end CPU-based architectures. Other studies concentrate on making Waveglow-based models more efficient and faster. *Efficient WaveGlow* [24] modifies the WaveGlow architecture by replacing the WaveNet-style transform network with an FFTNet-style [36] dilated convolution network. It results in significant parameter reduction and faster inference while maintaining similar Mean Opinion Scores. *SqueezeWave* [37] further adapts the WaveGlow model for mobile platforms, successfully running on devices like a MacBook Pro and a Raspberry Pi, albeit with a slow inference speed of 21k samples per second in the smallest model version, which can run on the single-board computer. These advancements have been tested across various hardware configurations, ranging from single-board computers to high-end CPUs. However, to our knowledge, our study is the first to target MCU platforms, pushing the boundaries of efficiency in Neural Vocoders even further.

III. Neural Architecture Design

In this Section, we present the main components of our vocoder pipeline. First, in Sec. III-A, we summarize the working mechanisms of Vocos [38]. Then, in Sec. III-B, we present the optimizations employed to improve the computational efficiency of Vocos and achieve real-time inference on MCUs.

A. Vocos Architecture

Vocos [38] is a recently proposed neural vocoder. It generates the waveform by first estimating the Fourier coefficients of the signal and then performing the Inverse Fourier Transform (ISTFT). This vocoder design avoids the sequence of costly upsampling operations needed to directly generate a waveform. In fact, Vocos substantially improves computational efficiency compared to prevailing time-domain neural vocoding approaches.

The neural architecture of Vocos, depicted in Fig. 1, comprises a sequence of convolutional blocks, $f_i(\cdot)$, to map the input spectrogram in a latent representation ($\mathbf{h}$). Then, the conversion head ($\mathcal{G}$) maps the latent representation ($\mathbf{h}$) into real and imaginary Fourier coefficients of dimensionality $n_{fft}/2+1$. Finally, the ISTFT maps the extracted coefficients to the time domain representation of the signal ($\mathbf{x}$).

Being $\mathcal{F}(\cdot)$ the sequence of convolutional blocks, $\mathcal{G}(\cdot)$ the projection from the latent representation ($\mathbf{h}$) to the Fourier coefficients, and $\mathbf{X}$ the input Mel-spectrogram, then $\mathbf{h} = \mathcal{F}(\mathbf{X})$ and

$$\begin{gathered}\mathbf{o} = \mathcal{G}(\mathbf{h}), \quad \mathbf{o} \in \mathbb{R}^{(n_{\text{fft}}/2+1)\times T}, \quad \mathbf{o}_i \in \mathbb{R}^T \\ \mathbf{m} = (o_1, ..., o_{n_{fft}/2+1}) \\ \mathbf{p} = (o_{n_{fft}/2+2}, ..., o_{n_{fft}+2})\end{gathered} \tag{1}$$

where $\mathbf{m}$ represents the log-scaled magnitude of the spectrogram and the phase ($\mathbf{P}$) can be extracted from the projections of $\mathbf{p}$ on the unit circle.

Finally, we can summarize the vocoder formulation in Vocos as

$$\mathbf{x} = \text{ISTFT}(\exp{(\mathbf{m})} e^{j\mathbf{P}}). \tag{2}$$

From this formulation, it is clear that there are two options to reduce the Vocos pipeline's complexity: (i) reduce the

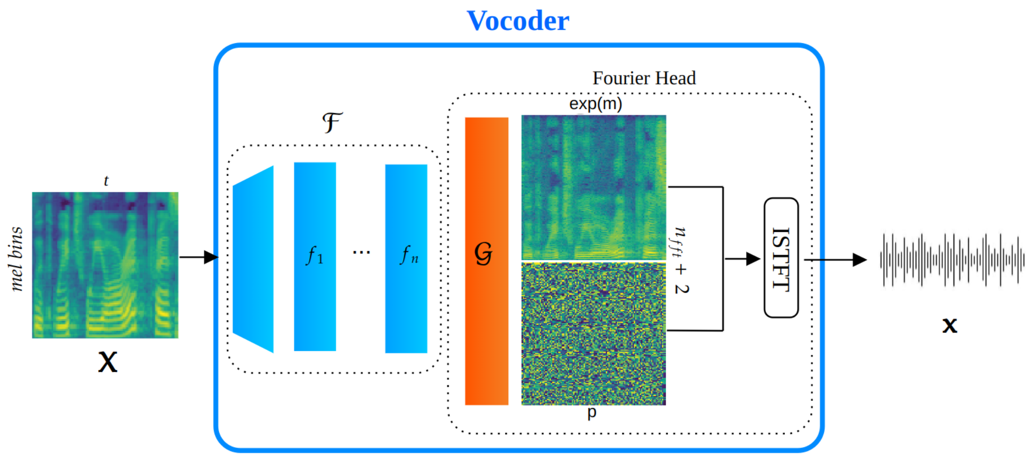

Fig. 1. The Vocos vocoder architecture projects Mel-spectrograms to the hidden representations. The hidden representation is afterwards fed to a conversion head that generates the Fourier coefficients and generates a waveform using ISTFT.

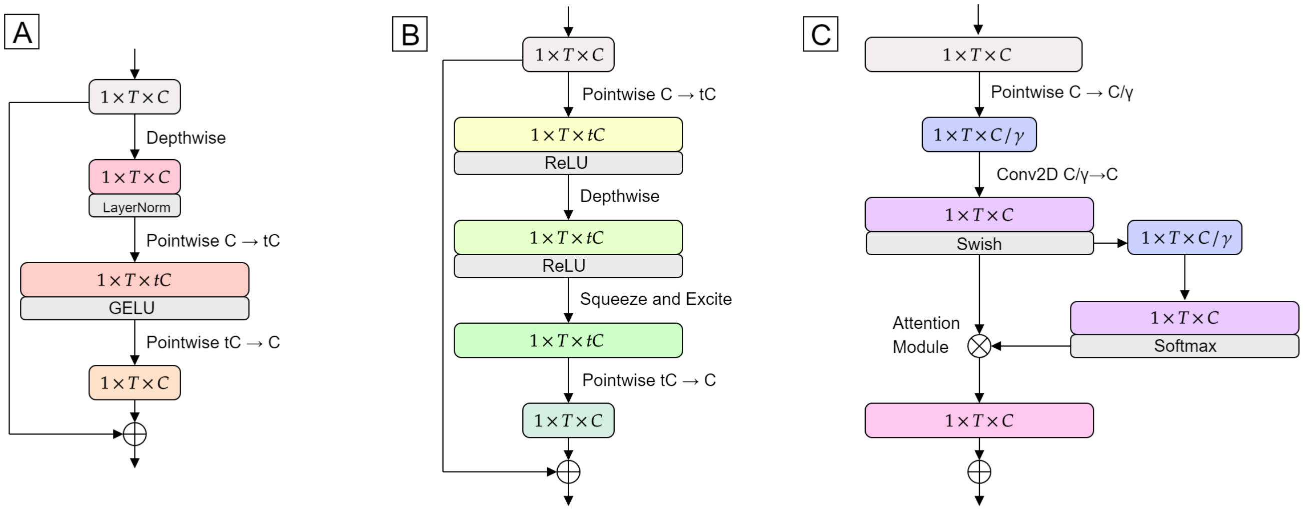

Fig. 2. Comparison of the functional form of the benchmarked convolutional blocks. From left to right, A: ConvNext; B: PhiNet; C: XiNet.

complexity of the neural encoding stage ($\mathcal{F}(\cdot)$) or (ii) reduce the latency of the ISTFT.

In this paper, we conduct an empirical study on the impact of using more efficient neural operators as Vocos convolutional blocks. Finally, to reduce the latency of the ISTFT, we reduce the number of frequency bins (n_{fft}). We note that the alternative would be to crop the audio signal, thus restricting the application domain of the vocoder.

B. Optimizing the Encoding

To analyze the tradeoffs in terms of performance *vs.* memory, performance *vs.* latency and performance *vs.* model size, we explore three convolutional blocks as alternatives to reduce the complexity of Vocos. First, we adapt the Vocos pipeline, restricting the network design to operations supported by common MCU toolchains. Then, we implement PhiNet [26] and XiNet [25], two edge-oriented architectures.

Vocos Adaptation. The ConvNext block, depicted in Fig. 2-A, is used as basic processing block ($f_i(\cdot)$) to encode the spectrograms in Vocos. It employs a depthwise convolution, followed by layer normalization and two pointwise convolutions. While this approach achieves promising performance, it presents two key limitations hindering its suitability for our desired application: (i) the availability of operators on MCU deployment toolchains and (ii) the memory footprint of the model. Regarding (i), Vocos employs the Gaussian Error Linear Unit (GELU) activation function in its convolutional blocks, which is currently not supported by many embedded runtimes (e.g. TFLite/STM32Cube.Ai/nncase). To address this, we replaced GeLU with Sigmoid Linear Units

TABLE I
COMPUTATIONAL CAPABILITIES OF THE EDGE PLATFORMS USED FOR ON-DEVICE BENCHMARKING.

	Clock frequency [MHz]	RAM [kB]	External RAM [MB]	FLASH [MB]	External FLASH [MB]
STM32H735G-DK	550	564	16	1.00	64
NUCLEO-H743ZI2	480	1024	N/A	2.00	N/A
STM32H7S78-DK	600	620	16	0.64	128
STM32H7A3ZIQ	280	1024	N/A	2.00	N/A

(SiLU), a functionally similar operation with documented TFLite support. In Section V-A, we quantify the impact of these approximations on the LibriTTS benchmark. To address the second constraint, we propose maintaining the hidden size fixed among pointwise convolutions, thus limiting the memory footprint of each ConvNext block.

PhiNet. PhiNets [26] is a family of neural architectures based on inverted residual blocks [39], depicted in Fig. 2-B. Its convolutional block is, therefore, a sequence of pointwise, depthwise, and again pointwise convolutions. The main advantage of PhiNets compared to functionally similar neural networks (e.g. MobileNet) lies in the scaling properties. PhiNets enable the disjoint optimization of RAM, FLASH, and the model's operation count. To exploit the effectiveness of this network for vocoders, we replaced the ConvNext blocks with the same inverted residual blocks used in PhiNets.

XiNet. Like PhiNets, XiNets [25] are convolutional networks designed to optimize energy consumption on resource-constrained devices. However, XiNets employs direct efficiency measurements to design the neural network. The convolutional block of XiNet is designed using only operations that are optimized on common inference toolchains. Specifically, the structure of the XiNet blocks - depicted in Fig. 2-C - comprises a pointwise convolution to bottleneck the number of channels, followed by convolution and a hybrid channel/spatial attention mechanism. Compared to PhiNets, XiNets generally exhibit faster execution speeds, improved energy efficiency and a smaller RAM footprint, albeit with a marginally larger parameter footprint when scaled for a speech synthesis task. For this, we benchmarked the performance of XiNet blocks in the Vocos pipeline.

IV. EXPERIMENTAL SETUP

To validate the effectiveness of the proposed efficient Vocoder, we benchmark the proposed approach at different computational budgets. We scale the three models as described in Sec. IV-A. We benchmark the models on two datasets, described in Sec. IV-B. We use quantitative and qualitative performance metrics, described in Sec. IV-C and Sec. IV-D, respectively. Finally, we present the target platforms in Sec. IV-E.

A. Model Implementation

To compare the backbones fairly, we conducted experiments targeting the computational complexity of the original Vocos implementation. We denote these models as L in the remainder of the manuscript. Then, we scaled down all architectures to computational budgets that enable on-device inference. Specifically, we targeted a medium computational budget that simulates the constraints typical of single-board computers (e.g. RaspberryPi). Resulting models are denoted as M. Finally, we further reduce the computational requirements of the models to achieve the least computationally intensive variants of the models presented in this manuscript - denoted as S. These models target MCU deployment and are tested and benchmarked on four target platforms.

As showcased in Table II, during model scaling, we followed the design principles of Vocos. The hidden size is fixed for all layers that compose the network, but we change it among different model variants. In fact, to reduce the computational cost of the models, we change both the number of layers and the hidden size. Additionally, we reduce the frequency bins from 1024 to 512, which linearly impacts the RAM usage and number of operations. For PhiNet-based models, we use the expansion factor of 1. For XiNet-based models, we use a compression factor of 4 in the bottleneck of the convolutional block. We use the default attention block proposed in the original model. However, we do not employ the broadcast skip connection mechanism presented in the original XiNet paper. For Vocos, we report the hidden size for the second pointwise convolution as "Hidden Dim 2" in Table II. For a reference implementation of PhiNet and XiNet, refer to the official repo[2].

To compare with a Vocoder that does not use deep learning, we report the results obtained using the Fast Griffin-Lim algorithm [19] coupled with Mel inversion.

TABLE II
MODEL HYPERPARAMETERS EMPLOYED FOR EMPIRICAL EVALUATION OF THE PIPELINE. VOCOS STD DENOTES THE ORIGINAL MODEL WITH THE GELU ACTIVATIONS.

Size	Model	n_{fft}	N	Hidden Dim	Hidden Dim 2
S	XiNet	512	3	128	N/A
	PhiNet	512	8	128	N/A
	Vocos	512	3	128	128
M	XiNet	512	4	512	N/A
	PhiNet	512	18	512	N/A
	Vocos	512	8	512	768
L	XiNet	1024	8	512	N/A
	PhiNet	1024	8	1024	N/A
	Vocos	1024	8	512	1536
	Vocos STD	1024	8	512	1536

[2] https://github.com/micromind-toolkit/micromind

B. Datasets

We evaluated the performance of the proposed architecture variants through experiments on the LibriTTS [40] and LJSpeech [41] benchmarks.

LibriTTS. To promote a fair comparison with Vocos, we trained the models on the LibriTTS dataset. LibriTTS is a multi-speaker English corpus of approximately 585 hours of read English speech. We use the entire training subset (both `train-clean` and `train-other`). The sampling rate is fixed to 24 kHz. We apply a random gain to the audio samples, resulting in a maximum level between -1 and -6 dBFS.

LJSpeech. A widely used benchmark for speech synthesis tasks is LJSpeech. LJSpeech contains 13,100 short audio clips of a single speaker reading passages from 7 non-fiction books. Following the standard practice, we adopted the train-validation split from the HiFi-GAN paper, consisting of 12,950 samples for training and 150 samples for validation, each corresponding to audio clips between 1.11 s and 10.10 s. We resample the data from 22 050 Hz to 24 kHz. For each audio sample, we compute Mel-scaled spectrograms using $n_{fft} = 1024$, $hop_n = 256$, and the number of Mel bins set to 80. We note that the bin count differs from the one used for the LibriTTS pre-processing. We do this to align with the Tacotron [17] pipeline for TTS evaluation. As in LibriTTS, we apply a random gain to the audio samples, resulting in a maximum level between -1 and -6 dBFS.

For both datasets, we crop all the waveforms to 16384 samples during the Vocoder training. This procedure reduces the training time while guaranteeing good variability among audio recordings. The implementation of the pre-processing pipeline is available in the official Vocos repository[3].

C. Quantitative Evaluation

We use five metrics to evaluate the reconstructed speech quality, each representing a specific property of the generated signal. We describe each metric in detail below.

PESQ. The Perceptual Evaluation of Speech Quality [42] is a metric originally developed to estimate the speech quality experienced by a telephony system user. PESQ uses the reference audio and analyzes the generated speech signal sample-by-sample after a temporal alignment. It analyzes features like distortion and noise in a degraded speech signal and maps them onto a predicted listener experience score. Validated against human listening tests, PESQ offers a standardized way to assess speech quality in research and development. As PESQ results principally model mean opinion scores (MOS), it is a scale ranging from 1 (bad) to 5 (excellent). For this metric, higher is better.

UTMOS. The UTokyo-SaruLab Mean Opinion Score [43] is a publicly available MOS prediction system developed by researchers from the University of Tokyo for the 2022 MOS challenge. The MOS estimation is based on Wav2Vec2 [44] features extracted from the input audio. This metric is between 1 (bad) and 5 (excellent). Higher scores correlate with better audio synthesis.

V/UV F1. Introduced by Morrison et al. [45], the voiced/unvoiced F1-score measures the quality of frame-level classification between voiced/unvoiced signals. As V/UV F1 indicates whether the frame exhibits the periodic structure of a pitched sound, low values correlate to samples exhibiting artifactual patterns. Higher is better.

Periodicity. Compares the frame-wise periodic structure of the signal. For this metric, lower is better as it implies that the generated audio closely resembles the periodic structure of the original waveform.

ViSQOL. Proposed by Chinen et al. [46], it is designed to measure the perceptual quality of audio and speech. Since our evaluation is performed on speech datasets, we utilize ViSQOL in speech mode (at 16 kHz) to compare the different Vocoders.

D. Qualitative Evaluation

Quantitative metrics provide a reproducible comparison between different Vocoders. However, user perception remains an essential aspect of generative modelling. We conduct two user studies to verify our model's perceived quality. First, we evaluate the naturalness and fidelity of the reproduced waveform with respect to the original audio. We use the same protocol employed in the Vocos evaluation. To assess the naturalness, we collect a MOS score (scale 1 to 5) for samples without presenting the reference to the users. Then, we measure the fidelity to the original waveform by computing a similarity-MOS (sMOS) score, in which the user provides a score from 1 (bad) to 5 (excellent), evaluating the fidelity to the input audio.

Finally, we evaluate the perceived quality of the model on a downstream, generative task. Using the Tacotron pipeline, we generate speech from text prompts. The only component of the pipeline that we changed is the Vocoder. During this user study, we provided the users with the input prompt for the model, and we asked them to rank the audio quality from 1 (bad) to 5 (excellent) using the Waveglow generated speech as a reference stimulus. This evaluation can be crucial in benchmarking our model for smart assistant applications, which are suitable target applications for the Internet of Sounds domain.

Each user study has eight samples and compares the PhiNet-based, XiNet-based and scaled Vocos pipelines. We collect results from 18 participants. We conducted both user studies using the WebMUSHRA [47] toolkit. All the generated audio is available for listening on a companion website[4].

E. On-device Benchmarking

To validate our results on MCUs, we deployed the S-variants of our model on four ARM Cortex-M7-powered boards. Precisely, we measure inference time, RAM usage, and model footprint on the `STM32H735G-DK`, `NUCLEO-H743`, `STM32H7S78-DK`, `NUCLEO-H7A3`. For

[3] https://github.com/gemelo-ai/vocos

[4] https://sciapponi.github.io/tinyvocos-companion

TABLE III

Quantitative comparison of model configurations on the LibriTTS Test-Clean-100 set.

Size	Model	F1 V/UV (↑)	Periodicity (↓)	PESQ (↑)	UTMOS (↑)	ViSQOL (↑)	Params [M]	MAC [M]
S	XiNet	**0.921**	**0.173**	2.09	**3.09**	**2.73**	0.39	39.5
	PhiNet	0.917	0.183	2.11	2.96	2.63	0.27	22.4
	Vocos	0.920	**0.173**	**2.37**	2.96	1.32	0.26	26.0
M	XiNet	0.928	0.156	2.52	3.46	**3.41**	6.76	677
	PhiNet	0.929	0.157	2.69	3.37	2.96	6.72	516
	Vocos	**0.947**	**0.120**	**3.33**	**3.67**	1.26	6.97	697
L	XiNet	0.925	0.158	2.39	3.54	3.03	12.3	1230
	PhiNet	0.952	0.130	3.28	3.49	3.49	12.4	968
	Vocos	0.951	0.112	3.51	3.46	4.07	13.5	1352
	Vocos STD	**0.958**	**0.101**	**3.70**	**3.73**	**4.08**	13.5	1352
	GL + Mel Inversion	0.517	0.482	1.12	1.27	1.95	N/A	N/A

TABLE IV

Quantitative Results for each model configuration on the LJSpeech Validation set. We used 80 mel bin spectrograms to make the models compatible with Tacotron.

Size	Model	F1 V/UV (↑)	Periodicity (↓)	PESQ (↑)	UTMOS (↑)	ViSQOL (↑)	Params [M]	MAC [M]
S	XiNet	0.9469	**0.1422**	2.452	**3.842**	**3.214**	0.39	39.40
	PhiNet	**0.9552**	0.1627	2.369	3.471	2.738	0.25	22.12
	Vocos	0.9469	0.1529	**2.619**	3.564	2.580	0.24	24.18
	GL + Mel Inversion	0.6715	0.4791	1.129	1.276	2.294	N/A	N/A

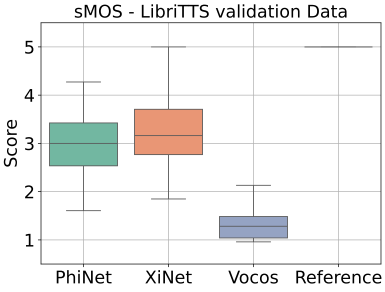

Fig. 3. sMOS for LibriTTS evaluation.

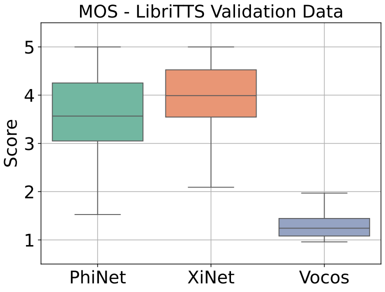

Fig. 4. MOS for LibriTTS evaluation.

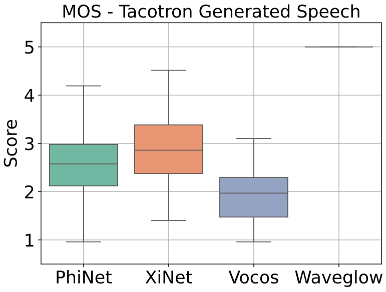

Fig. 5. MOS for TTS evaluation.

the NUCLEO-H743, we also report the power consumption of the proposed Vocoder. The computational constraints of the target platforms, reported in Table I, range from clock frequencies of 280 MHz to 600 MHz, from 564 kB to 1 MB of RAM and from 0.64 MB to 2 MB of FLASH. We used the ST Developer Cloud[5] to analyze the on-device performance of the models on all platforms except the NUCLEO-H7A3.

To measure the power consumption on the NUCLEO-H7A3, we used a $1\,\Omega$ shunt resistor. Then, we computed the average power consumption as:

$$P = \frac{1}{T}\sum_{t=t_0}^{T} I(t)V(t) \tag{3}$$

where $I(t)$ is the current measured over the shunt resistor at time t and $V(t) \approx 1.8\,\mathrm{V}$ is the MCU VIN. For all the measurements, we used a time window of $2\,\mathrm{s}$ to account for statistical fluctuations in measurements.

F. Training Strategy

For all the experiments reported in this paper, we used the same training loop presented in Vocos. The code is publicly available in the original Vocos GitHub[6].

V. Results

This section presents the results of the empirical analysis we conducted on the proposed vocoders. Sec. V-A summarizes the quantitative results, highlighting the best performance-complexity trade-off. In Sec. V-B, we present the outcome of the two user studies. Finally, Sec. V-C presents the power consumption analysis on the NUCLEO-H7A3 development board.

A. Quantitative Analysis

We report the results on the LibriTTS and LJSpeech benchmarks in Tables III and IV, respectively. On the LibriTTS benchmark, we note the comparable performance of

[5] https://stm32ai.st.com/st-edge-ai-developer-cloud/

[6] https://github.com/gemelo-ai/vocos

the original Vocos model with the GeLU and SiLU activations, validating this design choice. Furthermore, PhiNet and Vocos achieve comparable performance for L models, while XiNet performs marginally worse. Only for UTMOS, XiNet outperforms both Vocos and PhiNet, suggesting that the qualitative perception of the generated audio might be superior. For networks with a medium computational budget (M), we targeted models with half the computational requirements of the original Vocos model (around 6M params and 6B MAC). We observe a similar trend as per the L models. Vocos is the best-performing model for V/UV F1, Periodicity, PESQ, and UTMOS. For ViSQOL, however, we observe a considerable improvement when adopting the XiNet model. Among the S models - with computational requirements of 0.3M parameters and ∼25M MAC - all models achieve similar performance on the LibriTTS and the LJSpeech benchmarks. Vocos obtains the highest PESQ, but XiNet obtains the best UTMOS and ViSQOL. PhiNet achieves a comparable V/UV F1 and Periodicity to XiNet and Vocos, with minimal performance degradation in UTMOS and ViSQOL.

As expected, from L to S, we observe a decreasing trend in V/UV F1, Periodicity and PESQ. Interestingly, UTMOS and ViSQOL exhibit less sensitivity to model size reduction. This observation suggests that smaller models can generate intelligible speech despite slightly degrading objective reconstruction quality, highlighted by the decrease in PESQ, V/UV F1 and Periodicity. This characteristic could prove advantageous in text-to-speech applications, where maintaining intelligibility remains paramount even with potential trade-offs in objective quality metrics. Finally, we observe that all benchmarked models obtain superior performance with respect to the Fast Griffin-Lim algorithm.

B. *Qualitative Analysis*

The quantitative evaluation suggests that the perceived quality might be kept despite the considerable complexity reduction obtained using the proposed optimization. The two user studies validate this hypothesis. In Table V, VI we report the MOS scores with their respective confidence intervals at 0.95. Instead, in Fig. 3, Fig. 4, Fig. 5, we report the outcome of the two user studies as boxplots, which showcase median and first and third quartile of the gathered results.

On the LibriTTS samples - reported in Fig. 3, Fig. 4 and Table V - the user preference is towards the Xinet model for both Naturalness and Similarity to the reference stimulus. The PhiNet model generated more robotic utterances, as confirmed by the lower MOS scores than XiNet. Contrary to what we expected from the findings in Table IV, the waveforms generated by Vocos were rated as extremely unnatural and dissimilar from the reference waveform, showcasing multiple artifacts upon generation.

Table VI and Fig. 5 display the qualitative results obtained using the TTS pipeline. Also in this user study, Vocos is evaluated as the worst model at this computational scale, although with a lower margin than the results on the LibriTTS benchmark.

TABLE V
MOS and sMOS score on 8 LibriTTS audio samples.

Size	Model	sMOS	MOS
S	PhiNet	3.02 ± 0.12	3.60 ± 0.13
	XiNet	**3.24** ± **0.12**	**3.95** ± **0.11**
	Vocos	1.48 ± 0.14	1.52 ± 0.16
	Reference	4.87 ± 0.06	N/A

TABLE VI
MOS scores - Qualitative Test on the Tacotron pipeline.

Size	Model	MOS
S	PhiNet	2.55 ± 0.11
	XiNet	**2.89** ± **0.12**
	Vocos	1.90 ± 0.10
	Waveglow	4.73 ± 0.10

C. *On-device Benchmark*

We measure the required RAM and FLASH for each S-model, considering the optimizations employed by STM32Cube.AI. The inference time to process 1 s of audio, reported in Table VII, is mainly dependent on the differences in clock frequencies and whether a model needs to access external RAM or FLASH. As expected, XiNet uses the lowest amount of RAM among benchmarked models (517KiB) despite its marginally higher parameter count. The most memory-hungry network remains Vocos, with 737KiB of RAM.

On all benchmark platforms, the proposed vocoders obtain real-time performance, taking 630 ms to process 1 s audio in the worst case. With the lowest clock frequency of 280 MHz, without using external memory, the `NUCLEO-H7A3` exhibits the biggest gap in inference time among models (589 ms for Vocos and 631 ms for PhiNet). As expected, most inference time is dedicated to encoding the Mel-spectrograms in the latent representation, as showcased in Fig. V-C and Table VII. Conversely, on the `STM32H7S78-DK`, all models perform with a similar latency, with Vocos being marginally slower (+25 ms), due to the additional memory transfers needed to

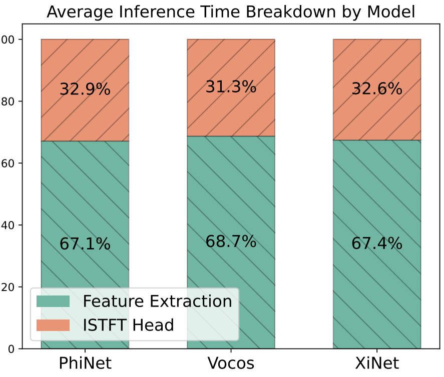

Fig. 6. Difference in percentage between Feature Extraction and ISTFT Head computation time for each model and S-sized model.

TABLE VII
RESULTS OF THE ON-DEVICE BENCHMARKING OF THE PROPOSED VOCODERS.

				Inference Time [ms]			
Board	**Model**	**FLASH [MiB]**	**RAM [KiB]**	**Total**	**Feature Extraction**	**Head**	**Power Usage [mW]**
STM32H735G-DK	XiNet	1.5	517	402	237	165	N/A
	PhiNet	1.1	559	**396**	235	161	N/A
	Vocos	1.3	737	531	357	174	N/A
NUCLEO-H743	XiNet	1.5	517	380	271	109	N/A
	PhiNet	1.1	559	393	281	112	N/A
	Vocos	1.3	737	**365**	255	110	N/A
STM32H7S78-DK	XiNet	1.5	517	**323**	215	109	N/A
	PhiNet	1.1	559	325	214	111	N/A
	Vocos	1.3	737	356	240	116	N/A
NUCLEO-H7A3	XiNet	1.5	517	613	448	165	**252**
	PhiNet	1.1	559	631	453	178	270
	Vocos	1.3	737	**589**	414	174	260

exploit the external RAM. We observe a similar trend on the STM32H735G-DK and NUCLEO-H743.

Finally, we observe that all vocoders have a power consumption around 250 mW, remarkably lower than single-board computers and workstations, which consume from 5 to several hundred Watts. Being low-power, these models enable a wide variety of generative applications in the IoS domain.

VI. CONCLUSION

In this work, we presented an empirical study on the effectiveness of spectral coefficient-based vocoders as scalable solutions for on-device inference. We compared the Vocos pipeline by replacing the convolutional blocks with more efficient variants from the tinyML literature (i.e. PhiNet and XiNet). Our quantitative and qualitative results demonstrate the feasibility of deploying neural vocoders on resource-constrained edge devices, potentially enabling new applications in IoS and Embedded Audio scenarios.

We observed some discrepancies between the results obtained using automated qualitative metrics (such as UTMOS and ViSQOL) and the user study. This is expected, especially considering the different nature of each metric. However, we think it is important to provide them as a reference for future works.

The proposed changes enable on-device audio synthesis while maintaining subjective speech quality suitable for text-to-speech (TTS) applications. This is evidenced by our best-performing model achieving a MOS of 3.95 out of 5 while utilizing only 1.5MiB of FLASH memory and 517KiB of RAM. Furthermore, this model consumes only 252 mW for a 1-second audio clip inference, confirming its suitability for energy-efficient deployments. These results open up new possibilities for IoT and edge computing scenarios, where resource constraints have traditionally limited the deployment of high-quality audio synthesis models.

REFERENCES

[1] L. Turchet, M. Lagrange, C. Rottondi, G. Fazekas, N. Peters, J. Østergaard, F. Font, T. Bäckström, and C. Fischione, "The Internet of Sounds: Convergent Trends, Insights, and Future Directions," *IEEE Internet of Things Journal*, vol. 10, no. 13, pp. 11 264–11 292, Jul. 2023, conference Name: IEEE Internet of Things Journal. [Online]. Available: https://ieeexplore.ieee.org/document/10061604/?arnumber=10061604

[2] J. Miquel, L. Latorre, and S. Chamaillé-Jammes, "Energy-Efficient Audio Processing at the Edge for Biologging Applications," *Journal of Low Power Electronics and Applications*, vol. 13, no. 2, p. 30, Jun. 2023, number: 2 Publisher: Multidisciplinary Digital Publishing Institute. [Online]. Available: https://www.mdpi.com/2079-9268/13/2/30

[3] L. Turchet and C. Fischione, "Elk Audio OS: An Open Source Operating System for the Internet of Musical Things," *ACM Transactions on Internet of Things*, vol. 2, no. 2, pp. 1–18, May 2021. [Online]. Available: https://dl.acm.org/doi/10.1145/3446393

[4] G. Cerutti, R. Andri, L. Cavigelli, E. Farella, M. Magno, and L. Benini, "Sound event detection with binary neural networks on tightly power-constrained IoT devices," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*. Boston Massachusetts: ACM, Aug. 2020, pp. 19–24. [Online]. Available: https://dl.acm.org/doi/10.1145/3370748.3406588

[5] G. Cerutti, R. Prasad, A. Brutti, and E. Farella, "Compact Recurrent Neural Networks for Acoustic Event Detection on Low-Energy Low-Complexity Platforms," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 4, pp. 654–664, May 2020, conference Name: IEEE Journal of Selected Topics in Signal Processing. [Online]. Available: https://ieeexplore.ieee.org/document/8970487/?arnumber=8970487

[6] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in DCASE 2022 Challenge," Jul. 2022, arXiv:2206.03835 [eess]. [Online]. Available: http://arxiv.org/abs/2206.03835

[7] D. Stefani, L. Turchet *et al.*, "On the challenges of embedded real-time music information retrieval," in *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, vol. 3, 2022, pp. 177–184.

[8] A. Ancilotto, F. Paissan, and E. Farella, "Ximswap: Many-to-many face swapping for tinyml," *ACM Transactions on Embedded Computing Systems*, vol. 23, pp. 1 – 16, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:258990773

[9] N. Dall'Asen, Y. Wang, H. Tang, L. Zanella, and E. Ricci, "Graph-based generative face anonymisation with pose preservation," *ArXiv*, vol. abs/2112.05496, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:245117660

[10] A. J. Obaid, "Assessment of smart home assistants as an iot," *International Journal of Computations, Information and Manufacturing (IJCIM)*, vol. 1, no. 1, 2021.

[11] A. Ancilotto, F. Paissan, and E. Farella, "Phinet-gan: Bringing real-time face swapping to embedded devices," *2023 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, pp. 677–682, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:259216938

[12] H. Liu, Z. Chen, Y. Yuan, X. Mei, X. Liu, D. P. Mandic, W. Wang, and M. . Plumbley, "Audioldm: Text-to-audio generation with latent diffusion models," in *International Conference on Machine Learning*, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:256390486

[13] H. Liu, Q. Tian, Y. Yuan, X. Liu, X. Mei, Q. Kong, Y. Wang, W. Wang, Y. Wang, and M. . Plumbley, "Audioldm 2: Learning holistic audio generation with self-supervised pretraining," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 32, pp. 2871–2883, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:260775781

[14] Y. Chen, R. Chen, J. Lei, Y. Zhang, and K. Jia, "Tango: Text-driven photorealistic and robust 3d stylization via lighting decomposition," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 30 923–30 936. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/c7b925e600ae4880f5c5d7557f70a72b-Paper-Conference.pdf

[15] R. Huang, J. Huang, D. Yang, Y. Ren, L. liu, M. Li, Z. Ye, J. Liu, X. Yin, and Z. Zhao, "Make-an-audio: text-to-audio generation with prompt-enhanced diffusion models," in *Proceedings of the 40th International Conference on Machine Learning*, ser. ICML'23. JMLR.org, 2023.

[16] Y. C. Subakan and P. Smaragdis, "Generative adversarial source separation," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 26–30.

[17] Y. Wang, R. J. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. V. Le, Y. Agiomyrgiannakis, R. Clark, and R. A. Saurous, "Tacotron: Towards end-to-end speech synthesis." in *INTERSPEECH*, F. Lacerda, Ed. ISCA, 2017, pp. 4006–4010. [Online]. Available: http://dblp.uni-trier.de/db/conf/interspeech/interspeech2017.html#WangSSWWJYXCBLA17

[18] F. Paissan, Z. Wang, M. Ravanelli, P. Smaragdis, and C. Subakan, "Audio editing with non-rigid text prompts," *Interspeech*, 2024. [Online]. Available: https://api.semanticscholar.org/CorpusID:264306038

[19] N. Perraudin, P. Balazs, and P. L. Søndergaard, "A fast griffin-lim algorithm," in *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2013, pp. 1–4.

[20] J. Kong, J. Kim, and J. Bae, "Hifi-gan: generative adversarial networks for efficient and high fidelity speech synthesis," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS'20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[21] J. Su, Z. Jin, and A. Finkelstein, "Hifi-gan: High-fidelity denoising and dereverberation based on speech deep features in adversarial networks," *ArXiv*, vol. abs/2006.05694, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:219558473

[22] T. Okamoto, T. Toda, and H. Kawai, "Multi-stream hifi-gan with data-driven waveform decomposition," in *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2021, pp. 610–617.

[23] D. Lim, S. Jung, and E. Kim, "JETS: jointly training fastspeech2 and hifi-gan for end to end text to speech," in *23rd Annual Conference of the International Speech Communication Association, Interspeech 2022, Incheon, Korea, September 18-22, 2022*, H. Ko and J. H. L. Hansen, Eds. ISCA, 2022, pp. 21–25. [Online]. Available: https://doi.org/10.21437/Interspeech.2022-10294

[24] W. Song, G. Xu, Z. Zhang, C. Zhang, X. He, and B. Zhou, "Efficient WaveGlow: An Improved WaveGlow Vocoder with Enhanced Speed," in *Interspeech 2020*. ISCA, Oct. 2020, pp. 225–229. [Online]. Available: https://www.isca-archive.org/interspeech_2020/song20_interspeech.html

[25] A. Ancilotto, F. Paissan, and E. Farella, "Xinet: Efficient neural networks for tinyml," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 16 968–16 977.

[26] F. Paissan, A. Ancilotto, and E. Farella, "Phinets: A scalable backbone for low-power ai at the edge," *ACM Trans. Embed. Comput. Syst.*, vol. 21, no. 5, dec 2022. [Online]. Available: https://doi.org/10.1145/3510832

[27] 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," *ArXiv*, vol. abs/1609.03499, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:6254678

[28] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SampleRNN: An unconditional end-to-end neural audio generation model," in *International Conference on Learning Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=SkxKPDv5xl

[29] R. J. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3617–3621, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:53145796

[30] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, "Improved variational inference with inverse autoregressive flow," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2016/file/ddeebdeefdb7e7e7a697e1c3e3d8ef54-Paper.pdf

[31] K. Kumar, R. Kumar, T. de Boissière, L. Gestin, W. Z. Teoh, J. M. R. Sotelo, A. de Brébisson, Y. Bengio, and A. C. Courville, "Melgan: Generative adversarial networks for conditional waveform synthesis," in *Neural Information Processing Systems*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:202777813

[32] J. Yuan, Z. Deng, S. Wang, and Z. Luo, "Multi receptive field network for semantic segmentation," *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1883–1892, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:215912060

[33] Z. Liu and Y. Qian, "Basis-MelGAN: Efficient Neural Vocoder Based on Audio Decomposition," in *Interspeech 2021*. ISCA, Aug. 2021, pp. 2222–2226. [Online]. Available: https://www.isca-archive.org/interspeech_2021/liu21h_interspeech.html

[34] Y. Luo and N. Mesgarani, "TaSNet: Time-Domain Audio Separation Network for Real-Time, Single-Channel Speech Separation," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2018, pp. 696–700, iSSN: 2379-190X. [Online]. Available: https://ieeexplore.ieee.org/document/8462116

[35] A. Mustafa, N. Pia, and G. Fuchs, "StyleMelGAN: An Efficient High-Fidelity Adversarial Vocoder with Temporal Adaptive Normalization," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Jun. 2021, pp. 6034–6038, iSSN: 2379-190X. [Online]. Available: https://ieeexplore.ieee.org/document/9413605/?arnumber=9413605

[36] Z. Jin, A. Finkelstein, G. J. Mysore, and J. Lu, "Fftnet: A real-time speaker-dependent neural vocoder," *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2251–2255, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:4081484

[37] 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*, vol. abs/2001.05685, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:210702343

[38] H. Siuzdak, "Vocos: Closing the gap between time-domain and fourier-based neural vocoders for high-quality audio synthesis," *arXiv preprint arXiv:2306.00814*, 2023.

[39] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[40] H. Zen, V. Dang, R. Clark, Y. Zhang, R. J. Weiss, Y. Jia, Z. Chen, and Y. Wu, "LibriTTS: A Corpus Derived from LibriSpeech for Text-to-Speech," in *Proc. Interspeech 2019*, 2019, pp. 1526–1530.

[41] K. Ito and L. Johnson, "The lj speech dataset," https://keithito.com/LJ-Speech-Dataset/, 2017.

[42] J. Beerends, A. Hekstra, A. Rix, and M. Hollier, "Perceptual evaluation of speech quality (pesq) - the new itu standard for end-to-end speech quality assessment - part ii - psychoacoustic model," *Journal of the Audio Engineering Society. Audio Engineering Society*, vol. 50, 10 2002.

[43] T. Saeki, D. Xin, W. Nakata, T. Koriyama, S. Takamichi, and H. Saruwatari, "UTMOS: utokyo-sarulab system for voicemos challenge 2022," in *Interspeech 2022, 23rd Annual Conference of the International Speech Communication Association, Incheon, Korea, 18-22 September 2022*, H. Ko and J. H. L. Hansen, Eds. ISCA, 2022, pp. 4521–4525. [Online]. Available: https://doi.org/10.21437/Interspeech.2022-439

[44] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and

H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 12 449–12 460. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf

[45] M. Morrison, C. Hsieh, N. Pruyne, and B. Pardo, "Cross-domain neural pitch and periodicity estimation," *ArXiv*, vol. abs/2301.12258, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:256390220

[46] M. Chinen, F. S. C. Lim, J. Skoglund, N. Gureev, F. O'Gorman, and A. Hines, "Visqol v3: An open source production ready objective speech and audio metric," *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, pp. 1–6, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:216036054

[47] M. Schoeffler, S. Bartoschek, F.-R. Stöter, M. Roess, S. Westphal, B. Edler, and J. Herre, "webmushra — a comprehensive framework for web-based listening tests," *Journal of Open Research Software*, Feb 2018.